3D Textured Surface Reconstruction from Endoscopic Video

Rui Wang

A dissertation submitted to the faculty of the University of North Carolina at Chapel Hill in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Computer Science.

Chapel Hill
2020

Approved by:

Stephen M. Pizer and Jan-Michael Frahm, Advisor

Ron Alterovitz, Committee Member

Sarah Mcgill, Committee Member

Junier Oliva, Committee Member

ABSTRACT

RUI WANG: 3D Textured Surface Reconstruction from Endoscopic Video.
(Under the direction of Stephen M. Pizer and Jan-Michael Frahm.)

Endoscopy enables high-resolution visualization of tissue texture and is a critical step in many clinical workflows, including diagnosis of infections, tumors or diseases and treatment planning for cancers. This includes my target problems of radiation treatment planning in the nasopharynx and pre-cancerous polyps screening and treatment in colonoscopy. However, an endoscopic video does not provide its information in 3D space, making it difficult to use for tumor localization, and it is inefficient to review. In addition, when there are incomplete camera observations of the organ surface, full surface coverage cannot be guaranteed in an endoscopic procedure, and unsurveyed regions can hardly be noticed in a continuous first-person perspective.

This dissertation introduces a new imaging approach that we call *endoscopography*: an endoscopic video is reconstructed into a full 3D textured surface, which we call an *endoscopogram*. In this dissertation, I present two endoscopography techniques.

One method is a combination of a frame-by-frame algorithmic 3D reconstruction method and a groupwise deformable surface registration method. My contribution is the innovative combination of the two methods that improves the temporal consistency of the frame-by-frame 3D reconstruction algorithm and eliminates the manual intervention that was needed in the deformable surface registration method. The combined method reconstructs an endoscopogram in an offline manner, and the information contained in the tissue texture in the endoscopogram can be transferred to a 3D image such as CT through a surface-to-surface registration. Then, through an interactive tool, the physician can draw directly on the endoscopogram surface to specify a tumor, which then can be automatically transferred to CT slices to aid tumor localization.

The second method is a novel deep-learning-driven dense SLAM (simultaneous localization and mapping) system, called RNN-SLAM, that in real time can produce an endoscopogram with display of the unsurveyed regions. In particular, my contribution is the deep learning system in the RNN-SLAM, called RNN-DP. RNN-DP is a novel multi-view dense depth map and odometry estimation method that uses Recurrent Neural Networks (RNN) and trains utilizing multi-view image reprojection and forward-backward flow-consistency losses.

ACKNOWLEDGMENTS

First of all, I would like to express my sincere gratitude to my principal advisors Prof. Stephen Pizer and Prof. Jan-Michael Frahm for their guidance and support of my PhD study over the past five years. I am really grateful to Prof. Stephen Pizer for giving me the opportunity to work on the endoscopic project and leading me into the medical image analysis field. He is truly a great mentor to me not only because of his guidance in my research field but also because of his help in my personal life. I am also very grateful to Prof. Jan-Michael Frahm for his motivation and guidance in computer vision field and for his help in my career.

I also thank my committee members Prof. Ron Alterovitz, Prof. Junier Oliva and Dr. Sarah McGill for their insightful advice and feedback on my research and dissertation. Without their efforts, I could not have made the current achievements. I thank Dr. Julian Rosenman whose optimism, enthusiasm and sense of humor really encouraged me on both my research and life, especially the moments when I am facing unknown challenges. I thank my colleagues Dr. Qingyu Zhao, Dr. True Price, Adam Aji, Ruibin Ma and many others in UNC Computer Science Department who have helped me with my research. I also thank Dr. Colette Shen and Dr. Tong Zhu for their collaborations.

Finally, special thanks to my beloved family. I thank my parents for their financial and mental support during all these years. I thank my wife for coming along with me to the U.S. and sacrificing her time with her parents to accompany me.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

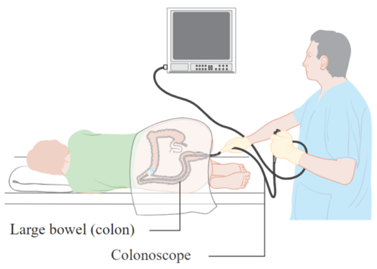
LoF	List of Figures
LoT	List of Tables
ToC	Table of Contents

Chapter 1

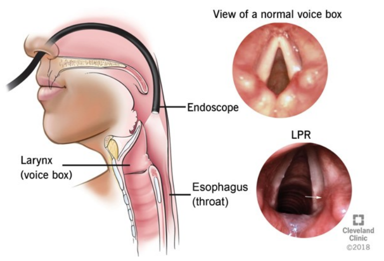
Introduction

Endoscopy is a common medical imaging procedure that provides direct view to the interior of a hollow organ or cavity of the body. Commonly, endoscopic devices consist of flexible tubing that contains a series of lighted mirror lenses and optic fibers. These instruments transmit light around corners, twists, and bends, allowing direct visualization of body systems not easily visualized by other means. An example of an endoscopic instrument and two common type of endoscopic procedures are shown in Figure 1.1. However, driving these flexible tubes is not easy because the orientations

(a) Endoscopic instrument



(b) Colonoscopy



(c) Pharyngoscopy

Figure 1.1: Example of endoscopic instruments and common endoscopic procedures.

of the lens can be arbitrary and, needless to say, the physicians must perform diagnosis simultaneously. Imagine that you are holding a camera and walking around to record a room while only staring at the screen on the camera. You may easily get lost or become unaware of which part of the room you didn't view. Also, if you take another video of the room and want to compare it with the video that you recorded before, it will be fairly difficult. The same problem exists in the endoscopic process; it is even worse due to the flexibility of the instrument and complexity of the environments in the human body.

We have seen that a reconstruction from the video into a 3D textured surface can be useful for resolving such problems. The 3D reconstructed surface not only can be helpful for the physicians to better localize themselves during the procedure but also can directly combine with other 3D imaging modalities to produce richer information for treatment planning. Additionally, it provides a more detailed, compact and natural representation for the patient organ than endoscopic videos and thus can be used for later examinations or even for training new endoscopists. The particular applications that we (the UNC endoscopography group) are working on are nasopharyngoscopy and colonoscopy.

1.1 3D reconstruction for nasopharyngoscopy

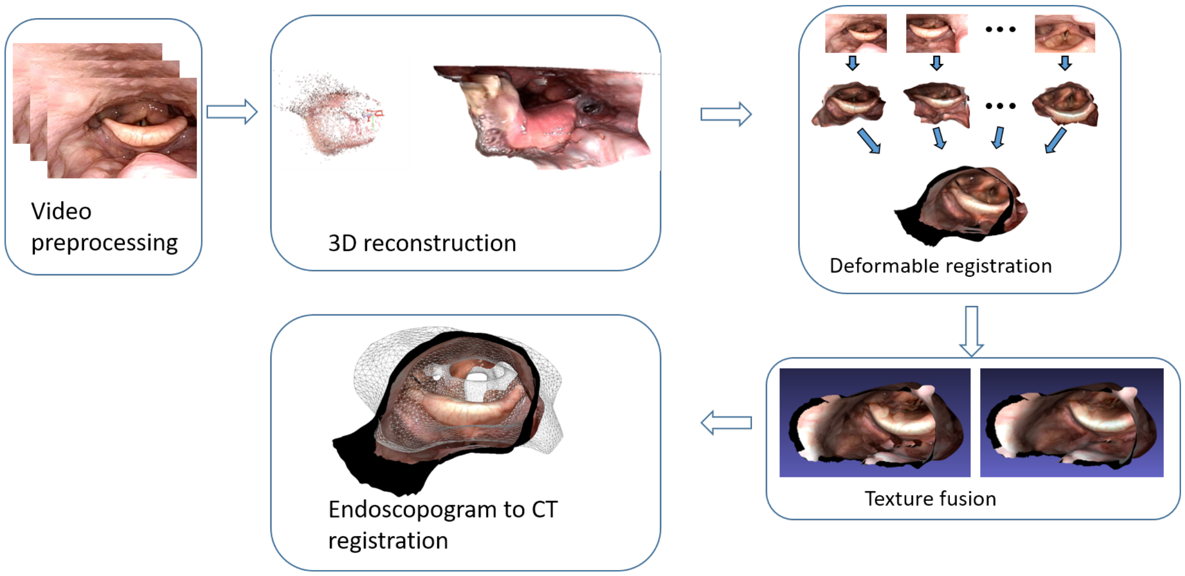


Figure 1.2: Overview of the system for 3D reconstruction in Nasopharyngoscopy.

Nasopharyngoscopy is a commonly used technique for nasopharyngeal cancer diagnosis and treatment planning. For radiotherapy, the planning requires tumor localization. Although nasopharyngoscopy can provide a direct, high-contrast, high-resolution visualization of a patient's interior tissue surface, it has a weakness for tumor localization in that it does not provide information below the tissue surface and in addition does not provide direct 3D spatial information. On the other hand, CT provides many critical sources of information on and below tissue surfaces, as needed in planning radiotherapy. However, it provides relatively low contrast and low resolution images for localization of the tumor, especially for tumors predominantly on the tissue surface, as is common in throat cancer. (The primary effect is on the surface; during development the tumor grows into the tissue). Therefore, if we can leverage the advantage of tissue information in nasopharyngoscopy together with the 3D geometry information in CT scan, the accuracy of tumor localization will be increased. In order to achieve that, we developed an automatic 3D reconstruction pipeline that can build a 3D surface model, which we call an *endoscopogram*, from the endoscopic video using many 2D video frames. Deformably registering the endoscopogram to a CT-extracted surface provides a means of fusing the information of these two images. Finally, we developed clinical software for physicians to draw a tumor contour on the endoscopogram and transfer the contour onto CT slices. Figure 1.2 shows the full pipeline of our method.

1.2 3D reconstruction for colonoscopy

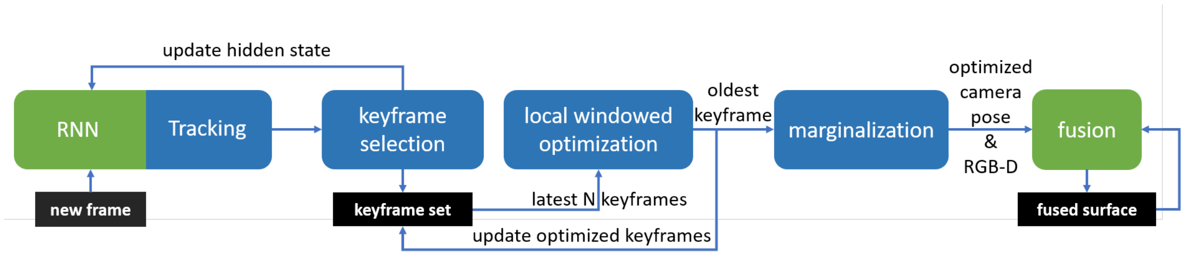


Figure 1.3: Overview of the system for 3D reconstruction in Colonoscopy.

Colon cancer is the second deadliest (Siegel et al., 2019) cancer in the U.S., and colonoscopy is currently the most effective way to reduce the chance of getting a colon cancer. These cancers are initiated by growths on the colon surface called polyps. However, clinical study shows that there is an average of twenty percent of polyps missed during current colonoscopic procedures. In a meta-analysis of six studies, using two immediate consecutive standard colonoscopies, on average 1 in every 5 adenomas was missed (pooled miss-rate 22%) (Van Rijn et al., 2006). Studies since then show this rate has not decreased (Lee et al., 2017).

In a general sense, polyps can only be missed at colonoscopy for two reasons. Either 1) the colonic mucosal surface was not entirely surveyed and thus some polyps were never seen, or 2) the polyps were indeed seen but not recognized as such. Currently much effort is being made to develop artificial intelligence systems that will detect polyps in real time during colonoscopy, with some success (Wang et al., 2019a; Shin et al., 2018), but studies of the extent of missed colonic surface and ways to prevent it have not yet been forthcoming.

Therefore, we have developed a real-time colon surface reconstruction algorithm that can be used for detecting the missing surface areas during the procedure. In contrast to pharygnoscopic 3D reconstruction, where the accuracy of the reconstructed surface has the highest priority, in missing-surface detection for colonoscopy, clinical practice requires the reconstruction to be accomplished within a few seconds. To do so, I have built a deep neural network that simultaneously estimates the visual odometry and depth maps from a video sequence taken by a monocular camera. In combination with dense visual SLAM, this framework can produce 3D reconstruction in real-time incrementally. Figure 1.3 shows the full pipeline of our real-time 3D reconstruction method for colonoscopic videos.

1.3 Challenges of 3D reconstruction from endoscopic videos

There already have been several attempts by researchers to accomplish 3D reconstruction from endoscopic images using computer vision (Kaufman and Wang, 2008), computer graphics (Hong et al., 2014) and machine learning techniques (Mahmood and Durr, 2018); I will elaborate in Chapter 2. However, the task is still considered unresolved due to the nature of difficulties in endoscopic videos. First, in these videos the light source is colocated with the moving camera, so the lighting consistently changes across frames. Also, for example in nasopharyngoscopy, the environment for 3D reconstruction is unknown because throat texture and shape can vary greatly from patient to patient, especially when tumors are present. Besides, due to the presence of the endoscope the throat constantly has sudden large deformations caused by the gag reflex and swallowing (Schwab et al., 1993; Kim et al., 2000). Moreover, the specularities of the saliva-coated throat tissue and the self-occlusions of different inner structures make the reconstruction even harder.

For colonoscopic videos, first, the colon surfaces are weakly textured. Second, the physician constantly cleans the colon mucosa by spraying and sucking water which causes deformation, blur and obscuration. Third, the colon surface consists of successive pockets and ridges, called haustra; the ridges occlude parts of the surface. Fourth, the colon is mostly contracted in its normal state, so in order to flatten the ridges and open the surface for better visualization, carbon dioxide is pumped into it to flatten the surface, a large deformation. In addition, the 3D reconstruction and analysis of colonoscopic images need to be in real time so that the endoscopist can be alerted to the unseen surface in a timely manner allowing the situation to be remedied.

1.4 A brief outline of the proposed methods

With all the aforementioned challenges in mind, in this dissertation I propose two frameworks for 3D reconstruction from endoscopic videos. The first, discussed in sections 1.4.1 and 1.4.2, focuses on the reconstruction quality. The second, discussed in sections 1.4.3 and 1.4.4, has the aim of real-time 3D reconstruction.

1.4.1 Fusion-guided structure-from-motion-and-shading

Structure-from-motion (SfM) and multi-view-stereo (MVS) form parts of the standard pipeline for estimating three-dimensional dense structures from two-dimensional image sequences. See Chapter 3, Section 3.1.1 for more details. However, the aforementioned challenges such as surface deformation and lighting change in endoscopic videos violates some of the fundamental assumptions in SfM and MVS such that the algorithms cannot directly produce a complete and consistent reconstruction.

Shape-from-shading (SfS) is a highly under-constrained single-view 3D reconstruction method that can only work when enough prior knowledge is given. Price et al. proposed to combine SfM with SfS that overcomes the challenges in endoscopic videos and thus produces dense surface reconstruction for every video frame. The method is named structure-from-motion-and-shading (SfMS). I further improved the temporal consistency and eliminated the manual intervention that was needed in the SfMS method by integrating a groupwise deformable registration into the iterative reconstruction pipeline. We call the improved pipeline fusion-guided SfMS. Fusion-guided SfMS can automatically reconstruct a complete 3D textured surface, an endoscopogram, from an endoscopic video in an offline manner.

1.4.2 Clinical software

The motivation for producing an endoscopogram from a pharyngoscopic video is to transfer the tumor detected in phyaryngoscopic video onto the CT for better treatment planning. To reach this goal, we developed clinical software that allows the physicians to draw contours on both the endoscopogram and the video frames and for the system to transfer the indicated ROI onto the CT slices. As shown in Figure 1.4, the software

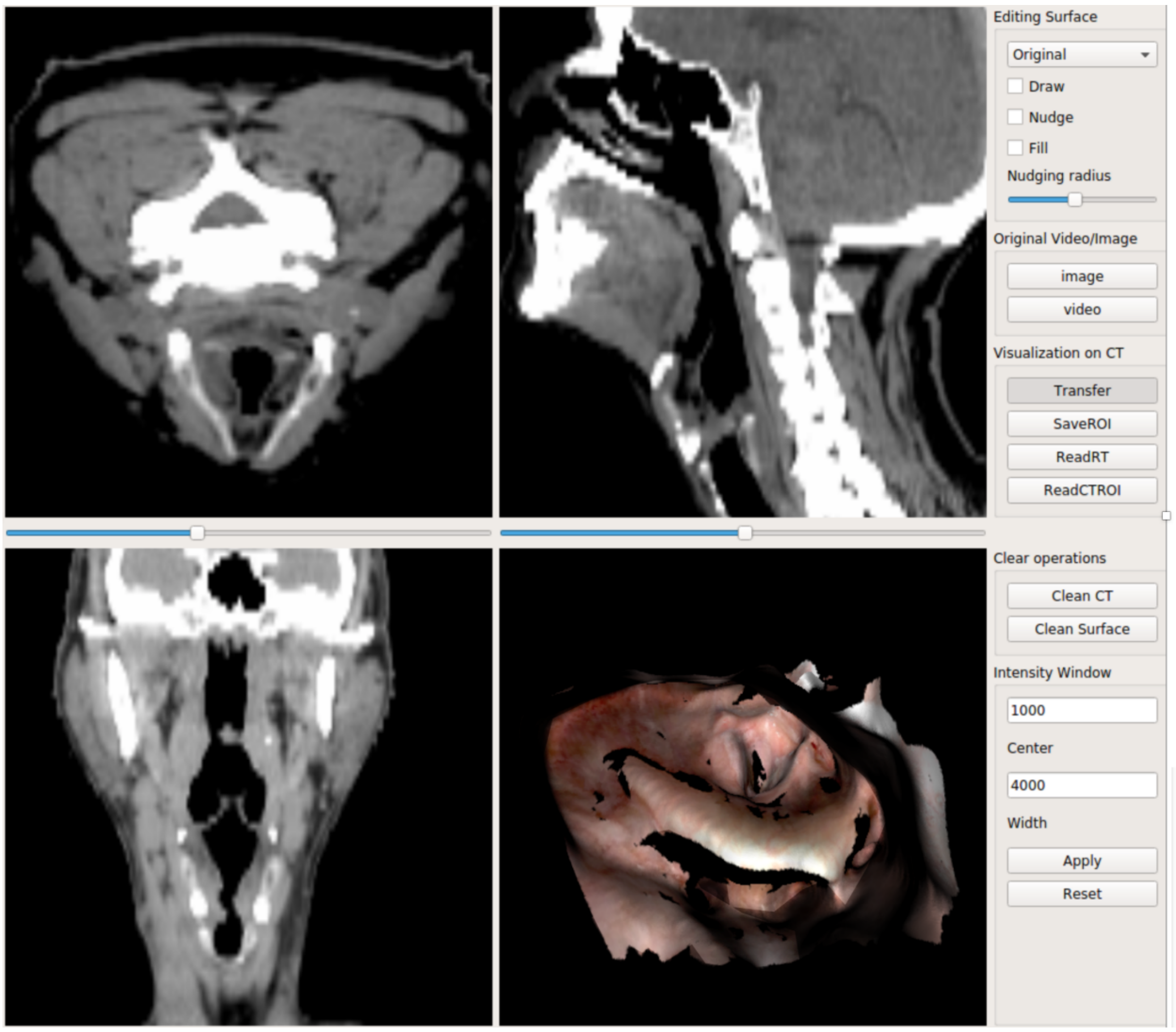


Figure 1.4: Clinical software that visualizes the three cardinal views of the CT and the reconstructed endoscopogram.

allows the physician to visualize both the CT slices and the 3D endoscopogram at the same time. In addition, it can import GTVs (Gross Tumor Volumes) that are drawn purely based on the CT to have a comparison with the ROI drawn based on the endoscopogram or endoscopic video frames.

1.4.3 Recurrent neural network for depth and visual odometry estimation

The aforementioned fusion-guided SfMS method can automatically generate a 3D textured surface from endoscopic videos, which is detailed in Chapter 3, Section 3.4. However, it only works in an offline manner. In colonoscopy, the motivation is to detect un-surveyed surfaces and give feedback to the physician in real time to reduce the missed polyp rate caused by missing surface. To achieve this, we turned our sights to deep neural networks that once trained can perform prediction in real time. The underlying rationale for deep-learning-based single view depth estimation methods is the possibility of human depth perception from a single image. A particular type of deep neural network that is designed to work on images is called a convolutional neural network (CNN). Commonly a CNN requires thousands or even millions of images with groundtruth labels for training. However, there is no adequately accurate groundtruth depth available for endoscopic videos. Therefore, in order to prove the concept of CNN-based real-time depth estimation, I used outdoor and indoor datasets with groundtruth depth measured by active sensors for experiments. I further leveraged a special type of deep neural network called a recurrent neural network (RNN) that can process temporal sequences to carry information from previous views into the current frame's depth and visual odometry estimation. Once trained, our RNN framework can simultaneously estimate the visual odometry and depth maps from a video sequence taken by a monocular camera. Finally, inspired by Zhou et al. (Zhou et al., 2017) and Godard et al. (Godard et al., 2017), I designed a multi-view image projection loss and a forward-backward consistency loss that allow unsupervised training of the RNN-based depth and odometry estimation network (RNN-DP).

1.4.4 Real-time 3D reconstruction for colonoscopic videos

The RNN-DP with multi-view image projection and forward-backward consistency losses introduced above works well for indoor and outdoor videos. However, due to surface deformation, large illumination changes and lack of texture in colonoscopic videos, the RNN-DP that without supervision trained on colonoscopic videos had unsatisfactory performance. To overcome the problem, we designed the following training strategy: 1) divide the colon videos into many small overlapping sub-sequences that each contains two hundred frames; 2) run SfM (Schonberger and Frahm, 2016) on all the sub-sequences to generate sparse depth maps for each frame; 3) use these sparse depth maps as ground-truth to train RNN-DP. Once trained, the RNN-DP can estimate depth and visual odometry for every informative incoming frame of a colonoscopic video in real-time. Informative frames are frames that have a clear view of the colon. We trained a CNN to automatically select informative frames (see chapter 6.1). However, in order to detect missing surfaces, we still need to fuse the estimated depth maps into a complete surface.

The RNN-DP estimated odometry has drifting problem due to lack of global pose optimization. A framework called visual SLAM (simultaneous localization and mapping) (Engel et al., 2014) is designed to fix this drifting problem and optimize both depth and pose in real-time. Therefore, we combined RNN-DP in a novel fashion with a SLAM pipeline to improve the stability and drift of successive frames' reconstructions. Based on these optimized camera poses, the depth maps of the keyframes are fused into a textured global mesh using a non-volumetric method.

1.5 Thesis and contributions

Thesis: *Endoscopography reconstructs a full 3D textured surface from an endoscopic video. We call this textured surface an endoscopogram. This opens the door for novel 3D visualizations of patient anatomy derived solely from endoscopic data.*

The contributions of this dissertation are as follows: I divide the achievements into two parts. The list of achievements for nasopharyngoscopic applications is

- An approach that integrates fusion into the iterative frame-by-frame 3D reconstruction that leads to more temporally consistent results.

- An optimization-based multi-view texture fusion algorithm that minimizes within-patch intensity gradient magnitude differences and inter-patch-boundary color differences.

The list of achievements for colonoscopic applications is

- A novel deep learning-based informative frame selection method that can automatically select frames that are suitable for 3D reconstruction.

- A novel recurrent neural network that can take advantage of temporal information for (un-)supervised learning of monocular video visual odometry and depth.

- An innovative combination of depth and pose estimation networks that allows the RNNs to be trained through two novel loss functions.
- A novel approach that interactively combines RNNs with visual SLAM that achieves real-time surface reconstruction from colonoscopic video. The prior knowledge learned by the RNNs provides a good initialization for the SLAM. The SLAM, on the other hand, performs optimization based regularization to the estimated depth and pose that resolves the drifting problem.

Besides the above methodological contributions, I have also accomplished the following engineering contributions:

- A full pipeline that integrates reconstruction, geometry fusion and texture fusion into an automatic process.
- Clinical evaluation software for tumor drawing on endoscopic video or endoscopogram and transfer to the CT space.
- A simulator for non-rigid 3D reconstruction evaluation.

With the above scientific and engineering contributions, the dissertation developed techniques to reconstruct a full 3D textured surfaces from endoscopic videos. In particular, they allow full 3D reconstruction from nasopharyngoscopies, thereby enabling physicians to efficiently review them and to visualize the endoscopic data directly in the CT space. They also allow reconstruction of colon surfaces into chunks that are then used to localize regions of inadequate surface covering during colonoscopy in real time.

1.6 Overview of chapters

The remainder of this dissertation is organized in the following chapters: Chapter 2 reviews mathematics and algorithm backgrounds for 3D vision and deep neural networks as well as image simulation methods. Chapter 3 describes the full pipeline of 3D reconstruction from pharyngoscopic videos and registration with CT. Chapter 4 gives a review of the state-of-the-art DNN-based 3D reconstruction methods. Chapter 5 describes my RNN-based depth and visual odometry estimation method and its application on outdoor and indoor datasets. Chapter 6 describes the framework for real-time 3D reconstruction from colonoscopic videos. Chapter 7 discusses the accomplishments and directions for future work.

Chapter 2

Background

In this chapter I present some background relevant to this dissertation. In section 2.1 I give an introduction to 3D vision that includes geometric camera models, camera calibration, and single-view and multi-view geometry. Section 2.2 gives a brief review of deep neural networks, including the back-propagation algorithm, convolutional neural networks and recurrent neural networks. Finally, section 2.3 discusses image simulation techniques for computer vision, in particular, those being used in deep learning and medical image analysis.

2.1 Introduction to 3D vision

2.1.1 Projective geometry

This dissertation deals with 3D reconstructions of a single 2D image either from a single 2D image or from a series of such images. In this subsection I will introduce the basic building blocks to describe the 2D and 3D world.

Euclidean geometry vs. projective geometry. Most people are familiar with Euclidean geometry that allows us to measure the length of an object and to determine the angle between two lines. These geometric properties within or between objects are invariant under Euclidean transformations (translation and rotation). However, in computer vision where the images are the projection from 3D space onto the camera plane, Euclidean geometry is no longer sufficient because the length of an object or the angle between lines is no longer preserved across depth. Projective geometry on the other hand includes many more transformations, including perspective projection, so it models the image formation well. Similar to the Cartesian coordinates that are used in Euclidean geometry, the coordinate system being used in projective geometry is homogeneous coordinates. Homogeneous coordinates have the advantage that the coordinates of points, including points at infinity, can be represented using finite coordinates.

Homogeneous coordinates. In Euclidean geometry a 2D-point can be represented as a pair of values $\boldsymbol{x} = (x, y) \in \mathbb{R}^2$, or in vector form,

$$\boldsymbol{x} = \begin{bmatrix} x \\ y \end{bmatrix} \tag{2.1}$$

In projective geometry the representation of a point uses homogeneous coordinates, $\tilde{\boldsymbol{x}} = (\tilde{x}, \tilde{y}, \tilde{\omega}) \in \mathbb{P}^2$, where $\mathbb{P}^2$ is the projective plane. Scale is unimportant in homogeneous coordinates; thus,

$$(\tilde{x}, \tilde{y}, \tilde{\omega}) = (\alpha\tilde{x}, \alpha\tilde{y}, \alpha\tilde{\omega}) \tag{2.2}$$

for any $(\tilde{x}, \tilde{y}, \tilde{\omega}) \neq (0, 0, 0)$ and $\alpha \in \mathbb{R}$. Points at infinity can be represented as $\tilde{\boldsymbol{x}} = (\tilde{x}, \tilde{y}, 0)$; these are also called ideal points.

Similarly, in Euclidean geometry a 2D line can be represented as

$$ax + by + c = 0 \tag{2.3}$$

If we replace the point (x, y) by homogeneous representation, we can get

$$a\tilde{x} + b\tilde{y} + c\tilde{\omega} = \tilde{\boldsymbol{l}} \cdot \tilde{\boldsymbol{x}} = 0 \tag{2.4}$$

where $\tilde{\boldsymbol{l}} = (a, b, c)$ is the homogeneous representation of a 2D line in projective geometry.

Projective geometry not only exists in the two dimensional space $\mathbb{P}^2$; it exists in any number of dimensions. Thus 3D points and planes in homogeneous coordinates can be represented as $\tilde{\boldsymbol{X}} = (\tilde{X}, \tilde{Y}, \tilde{Z}, \tilde{\Omega}) \in \mathbb{P}^3$ and $\tilde{\boldsymbol{L}} = (a, b, c, d)$

Duality. As we can see, in homogeneous coordinates 2D points and lines and respectively 3D points and planes have the same representation. This is due to the duality that exists between points and lines in the projective plane and between points and planes in the projective space. In a 2D plane a point can be represented as the intersection of two lines,

$$\tilde{\boldsymbol{x}} = \tilde{\boldsymbol{l_1}} \times \tilde{\boldsymbol{l_2}} \tag{2.5}$$

Similarly, a line going through two points on a projective plane can be represented as the cross product of two points,

$$\tilde{\boldsymbol{l}} = \tilde{\boldsymbol{x}}_{\boldsymbol{1}} \times \tilde{\boldsymbol{x}}_{\boldsymbol{2}} \tag{2.6}$$

Cross ratio. As mentioned, neither distances nor ratios of distances are preserved in projective space. However, the ratio of ratios of distances, the *cross ratio*, is preserved in projective geometry. Given four co-linear points $\tilde{\boldsymbol{p}}_{\boldsymbol{1}}, \tilde{\boldsymbol{p}}_{\boldsymbol{2}}, \tilde{\boldsymbol{p}}_{\boldsymbol{3}}, \tilde{\boldsymbol{p}}_{\boldsymbol{4}}$, the mathematical definition of cross ratio is

$$C_r(\tilde{\boldsymbol{p}}_{\boldsymbol{1}}, \tilde{\boldsymbol{p}}_{\boldsymbol{2}}, \tilde{\boldsymbol{p}}_{\boldsymbol{3}}, \tilde{\boldsymbol{p}}_{\boldsymbol{4}}) = \frac{|\tilde{\boldsymbol{p}}_{\boldsymbol{3}} - \tilde{\boldsymbol{p}}_{\boldsymbol{1}}||\tilde{\boldsymbol{p}}_{\boldsymbol{4}} - \tilde{\boldsymbol{p}}_{\boldsymbol{2}}|}{|\tilde{\boldsymbol{p}}_{\boldsymbol{3}} - \tilde{\boldsymbol{p}}_{\boldsymbol{2}}||\tilde{\boldsymbol{p}}_{\boldsymbol{4}} - \tilde{\boldsymbol{p}}_{\boldsymbol{1}}|} \tag{2.7}$$

where $C_r(\tilde{\boldsymbol{p}}_{\boldsymbol{1}}, \tilde{\boldsymbol{p}}_{\boldsymbol{2}}, \tilde{\boldsymbol{p}}_{\boldsymbol{3}}, \tilde{\boldsymbol{p}}_{\boldsymbol{4}})$ is invariant under projective transformations (illustrated in Figure 2.1).

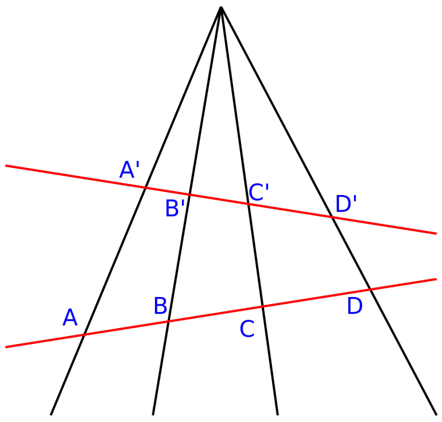


Figure 2.1: Example of the cross ratio under projective transformation. The cross ratio of (A, B, C, D) is equal to the cross ratio of (A', B', C', D').

2.1.2 Geometric camera model

The projective geometry introduced above is an important mathematical framework for most geometric computer vision problems, e.g., image formation through a geometric camera model. One of the simplest types of camera model is called the pinhole camera model.

As shown in Figure 2.2, the pinhole camera model assumes there is an infinitesimal pinhole on a barrier blocking most light. The light going through the pinhole will be projected onto an image plane. According to this pinhole camera model, a 3D point (X, Y, Z) in camera coordinates is projected onto the image plane point (x, y) via

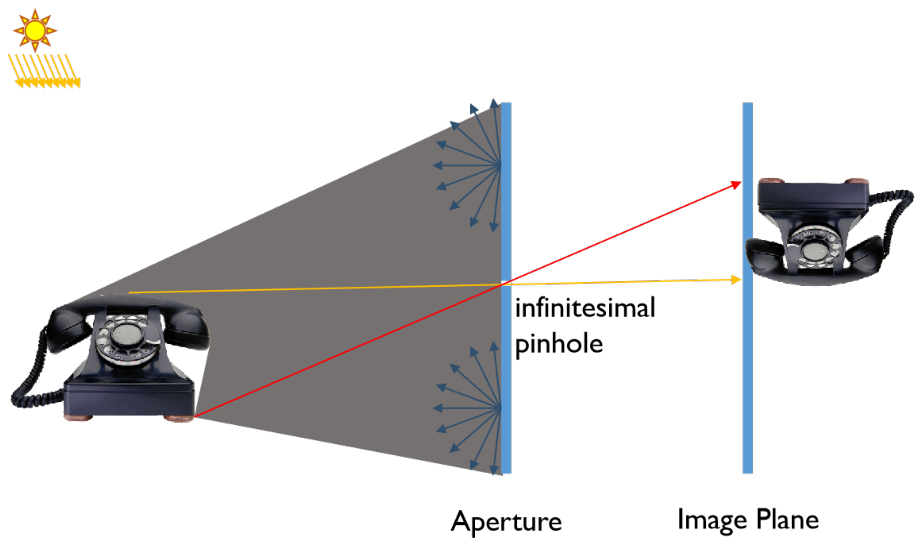


Figure 2.2: Example of a pinhole camera model.

perspective projection, which can be mathematically represented in Euclidean space as

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \frac{X}{Z} \\ \frac{Y}{Z} \end{bmatrix} \tag{2.8}$$

In homogeneous coordinates, perspective projection can be represented in linear form as

$$\tilde{\boldsymbol{x}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{2.9}$$

where $\tilde{\boldsymbol{x}}$ is the homogeneous version of a point in the image plane. The point $\tilde{\boldsymbol{x}}$ can then be converted from the image plane to the sensor plane, where the distance between points is measured as pixels:

$$\tilde{\boldsymbol{x}}_{\boldsymbol{s}} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \boldsymbol{K}\tilde{\boldsymbol{x}} \tag{2.10}$$

where (c_x, c_y) is an offset that accounts for different origins between the image plane coordinates and the sensor plane, and (f_x, f_y) is the focal length, which is the product of the physical focal length in millimeters with individual image pixel widths. The 3×3

matrix $\boldsymbol{K}$ is called the *calibration matrix* or *camera intrinsic matrix.*

While the camera intrinsic matrix converts points from image plane coordinates to sensor plane coordinates, there is also a *camera extrinsic matrix* that converts points from world coordinates to camera coordinates by

$$\tilde{\boldsymbol{x}}_{\boldsymbol{c}} = [\boldsymbol{R}|\boldsymbol{T}]\tilde{\boldsymbol{x}}_{\boldsymbol{w}} \tag{2.11}$$

$\tilde{\boldsymbol{x}}_{\boldsymbol{c}}$ and $\tilde{\boldsymbol{x}}_{\boldsymbol{w}}$ are homogeneous version of points in camera coordinates and world coordinates respectively, and $[\boldsymbol{R}|\boldsymbol{T}]$ is a 3×4 transformation matrix.

Since all the points are represented in homogeneous coordinates, Equations 2.9, 2.10 and 2.11 can be combined to form the projection from a point in world coordinates to a point in sensor plane coordinates as

$$\tilde{\boldsymbol{x}}_{\boldsymbol{s}} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{R}_{3\times 3} & \boldsymbol{T}_{3\times 1} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} = \boldsymbol{K}[\boldsymbol{R}|\boldsymbol{T}]\tilde{\boldsymbol{x}}_{\boldsymbol{w}} \tag{2.12}$$

$\boldsymbol{P} = \boldsymbol{K}[\boldsymbol{R}|\boldsymbol{T}]$ is called the *camera matrix* or *projection matrix.*

2.1.3 Camera calibration

The pinhole camera introduced above assumes an ideal pinhole, which is impossible in the real world. In the real world a lens is usually used to simulate the ideal pinhole. However, all lenses have different levels of distortion. Therefore, in order to perform 3D reconstruction, mapping between sensor coordinates and world coordinates, the lens distortion needs to be estimated.

The most common type of lens distortion is *radial distortion* wherein points are distorted along radial lines. Radial distortion occurs when light rays bend more near the edges of a lens than they do at its optical center, which can be mathematically represented as

$$\begin{aligned} x_d &= x_c(1 + k_1 r^2 + k_2 r^4) \\ y_d &= y_c(1 + k_1 r^2 + k_2 r^4) \\ r^2 &= x^2 + y^2 \end{aligned} \tag{2.13}$$

where (x_c, y_c) are undistorted (ideal) image plane coordinates and (x_d, y_d) are distorted image plane coordinates.

The camera intrinsics K and extrinsics $[R|T]$ that were introduced in previous section together with the lens distortion coefficients are called the *camera parameters*. The estimation of the camera parameters from a series of images is called *camera calibration*.

In order to estimate the camera parameters, 3D world points and their corresponding 2D image points are needed. The most common way to get these correspondences is using multiple images of a calibration pattern, such as a checkerboard. With at least 6 pairs of correspondences, the camera matrix P in equation 2.12 can be solved using Direct Linear Transformation method. Rearranging equation 2.12 we can get

$$\begin{pmatrix} X & Y & Z & 1 & 0 & 0 & 0 & 0 & -xX & -xY & -xZ & -x \\ 0 & 0 & 0 & 0 & X & Y & Z & 1 & -yX & -yY & -yZ & -y \end{pmatrix} p = 0 \tag{2.14}$$

where p is a 12×1 vector containing the elements of the camera matrix P.

By concatenating the above equation for $n \geq 6$ correspondences we can get $Ap = 0$, where A is a $2n \times 12$ matrix. The solution of p can be obtained from the eigenvector with least eigenvalue of $A^T A$. The linear solution of P can then be used as the initialization for the non-linear optimization of $\sum_i d^2(x_i, PX_i)$ to obtain a more accurate solution.

After the camera matrix P is obtained, the camera intrinsics and extrinsics can be computed through decomposition of P. The left 3×3 sub-matrix of P is the product of camera intrinsics K and rotation matrix R:

$$M = KR \tag{2.15}$$

Through QR matrix decomposition, M can be factored into K and R. The translation T can then be obtained as

$$T = K^{-1}(P_{14}, P_{24}, P_{34})^T \tag{2.16}$$

Finally, the lens distortion coefficients can be calculated by minimizing a non-linear function:

$$\min_{(K,k,R,T)} \sum_i ||x_i - \hat{x}(K, k, R, T; X_i)|| \tag{2.17}$$

where $\hat{x}$ is reprojection of the 3D point to its corresponding 2D image point through the combination of equations 2.12and 2.13.

In this dissertation most images dealt with are endoscopic images. Endoscopic images are taken from fisheye lenses, which have a large field of view. Fisheye lenses cannot be described using the pinhole model introduced before due to the very large distortion. The common model used for fisheye lenses is omnidirectional camera model. Scaramuzza *et al.* (Scaramuzza et al., 2006) proposed a model that relates the viewing direction to the scene point:

$$\tilde{\boldsymbol{x}}_{\boldsymbol{c}} = \lambda \begin{bmatrix} \tilde{x} \\ \tilde{y} \\ a_0 + a_2\rho^2 + a_3\rho^3 + a_4\rho^4 \end{bmatrix} \tag{2.18}$$

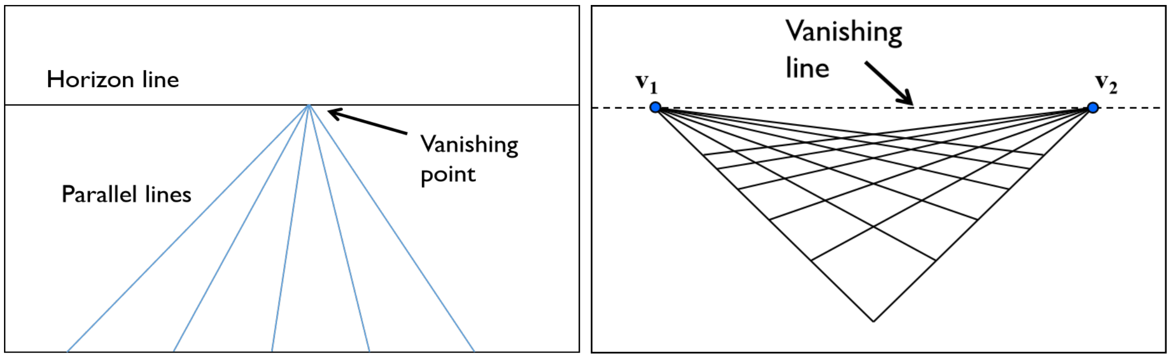
The calibration of the fisheye cameras is similar to that for calibrating standard perspective cameras.

2.1.4 Single-view geometry

Estimating depth from a single image has been a longstanding task in the computer vision field. The reason is that in many cases people do not have access to multiple images whereas we humans have the capability of recovering 3D information from a single image. The target objects to reconstruct in this dissertation are human anatomies. The organs in the human body are constantly deforming, which violates the fundamental assumptions in multi-view 3D reconstruction. Furthermore, the lighting condition is constantly changing due to the moving light source in endoscopic videos that also makes feature matching and tracking across different views very difficult. Therefore, single-view depth estimation becomes a natural solution to the problem of 3D reconstruction from endoscopic images.

Vanishing points and lines. The projective geometry introduced above is the earliest clue that people used to depict 3D information in a single image. As early as the 12th century, people started to use projective geometry in oil paintings to convey the 3D information. An obvious effect to most people is that when we drive along a straight road, the road appears to converge to a point. This is a fact in projective geometry that parallel lines in 3D will converge to a single point, called the *vanishing point* when projected onto the projective plane.

Figure 2.3a shows an example of the vanishing point. The steps for detecting vanishing points are to 1) detect line segments, 2) cluster the lines into groups with the assumption that a cluster will share a common vanishing point and 3) find the three dominant pair-wise orthogonal vanishing points. Different groups of parallel lines (lines



(a) Vanishing point (b) Vanishing line

Figure 2.3: Example of vanishing point and vanishing line.

not in the same group are non-parallel) on the same plane in 3D will form different vanishing points. These vanshing points will form a line on the projective plane, called the *vanishing line*, shown in Figure 2.3b.

A common usage of vanishing points and lines is measuring the height of an object from a single image. If two objects are sitting on the same plane and the vanishing line of the plane is known, then the relative height of the two objects can be measured using the cross ratio (equation 2.7). Vanishing points and lines can also be used for 3D reconstruction. We live in a man-made world that full of orthogonal planes. Therefore, by detecting pairwise orthogonal vanishing points and lines, the scene can be modeled as a set of planes. Anjyo *et al.* (Anjyo, 1997) proposed a method that can construct a simple 3D model from a single image or painting using vanishing points and perspective projected lines drawn from the vanishing points.

Shape-from-shading. Vanishing points and lines are useful when there are enough lines and planes in the image, which is usually not the case in endoscopic images. Another type of single-view modeling technique that leverages the shading and surface reflectivity property is called shape-from-shading. Shape-from-shading (SfS), first introduced by Horn (Horn, 1970), tries to solve the inverse problem of given an image, a model of the illumination and a model of the surface reflectivity, recovering the surface geometry. More background about SfS is presented in Chapter 3, Section 3.1.2.

Learning-based methods. Both perspective-geometry-based and shading-based single-view 3D reconstruction methods are ill-posed. They all require some prior assumptions in order to derive a reasonable result. These assumptions are based on our knowledge or experience and thus can be based on statistics rather than geometry. Hoiem *et al.* (Hoiem et al., 2005) proposed a statistical-learning-based 3D reconstruction method: they convert the 3D reconstruction problem into a recognition problem. For example, in training data they explicitly label every pixel into ground, vertical or sky. Once trained, their model is able to assign a geometric label to every pixel; then they reconstruct the 3D scene using those geometric labels. Saxena *et al.* (Saxena et al., 2008) also proposed a learning-based single-view 3D reconstruction method. They use a Markov Random Field (MRF) together with a set of learnable parameters to infer a set of "plane parameters". The image is divided into small homogeneous regions, called "Superpixels," and the learnable parameters map image features into the plane parameters for those superpixels. Other than local planarity, they made no explicit assumptions about the structure of the scene and thus get much better results than the method proposed by Hoiem *et al.*. However, modeling the mapping between image features and plane parameters using only a few hundred or thousand parameters is far from enough. Recently, by leveraging a deep neural network that contains millions or trillions of learnable parameters, learning-based single-view 3D reconstruction methods (Eigen et al., 2014) have achieved significantly better results. More background about deep-learning-based depth estimation is presented in Chapter 4, Section 4.1.

2.1.5 Multi-view geometry

I have introduced the geometric camera model in Section 2.1.2. The motion and calibration of multiple geometric camera models as well as the scene structure can be explicitly related using projective geometry.

Epipolar constraint. For a pair of cameras (c_1, c_2) that are viewing a common 3D point P, the projection of the 3D point in both cameras (x_1, x_2) can be related using a matrix; this relation is called the epipolar constraint,

$$x_1 M x_2 = 0 \tag{2.19}$$

As shown in Figure 2.4, without loss of generality we assume camera c_1 to be the world origin. Then camera c_2 has a relative rotation R and translation t to c_1. According to Section 2.1.2, a point in sensor coordinates can be converted to normalized camera coordinates using the inverse intrinsic matrix. Therefore, we can get

$$\begin{aligned} \tilde{x}_1 &= K_1^{-1} x_1 \\ \tilde{x}_2 &= K_2^{-1} x_2 \end{aligned} \tag{2.20}$$

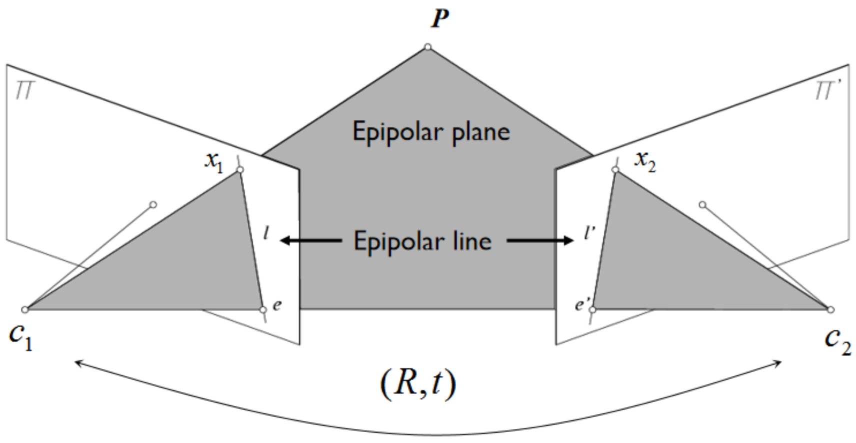


Figure 2.4: Points x_1 and x_2 that are viewing the same 3D point P in cameras c_1 and c_2 are related by the epipolar constraint.

And a point in normalized camera coordinates can be converted to camera coordinates by multiplying the depth of the 3D point, $P_1 = z_1\tilde{x}_1$, $P_2 = z_2\tilde{x}_2$. $P = P_1$ since we assume c_1 to be the world origin. Furthermore, a point in camera coordinates can be converted to world coordinates using the extrinsic matrix, $P = z_2R\tilde{x}_2 + t$. This relates $\tilde{x}_1$ and $\tilde{x}_2$ as

$$z_1\tilde{x}_1 = z_2R\tilde{x}_2 + t \tag{2.21}$$

Taking the cross product with t on both sides, we can get

$$z_1[t_\times]\tilde{x}_1 = z_2[t_\times]R\tilde{x}_2 \tag{2.22}$$

Then taking the dot product with $\tilde{x}_1$ on both sides, we can get

$$z_1\tilde{x}_1^T[t_\times]\tilde{x}_1 = z_2\tilde{x}_1^T([t_\times]R)\tilde{x}_2 = 0 \tag{2.23}$$

We therefore derive

$$\tilde{x}_1^T E\tilde{x}_2 = 0 \tag{2.24}$$

where

$$E = [t_\times]R \tag{2.25}$$

is called the *essential matrix*. By combining equation 2.20 with 2.24 we derive the epipolar constraint:

$$x_1^T K_1^{-T}EK_2^{-1}x_2 = \tilde{x}_1^T F\tilde{x}_2 = 0 \tag{2.26}$$

and

$$F = K_1^{-T} E K_2^{-1} = [e_\times]\tilde{H} \tag{2.27}$$

is called the *fundamental matrix*. The most common way to solve for the essential or fundamental matrix is known as the *normalized eight-point algorithm* (Hartley, 1997).

Once the fundamental matrix F or the essential matrix E is solved for, we can then compute the camera matrices. Through the derivation of fundamental matrix we can see that a pair of camera matrices determines a unique fundamental matrix. However, this relationship is not one-to-one: a fundamental matrix may correspond to pairs of camera matrices differing by projective transformations. Assume P_1 and P_2 are a pair of camera matrices that correspond to the fundamental matrix F. Then given a 4×4 projective transformation H, P_1H and P_2H also correspond to the fundamental matrix F. This is because $P_1X_1 = P_1H(H^{-1}X_1) = x_1$: both X_1 and $H^{-1}X_1$ are projected to the same scene point x_1, and similarly for x_2.

Given such ambiguity, a simple choice for a pair of camera matrices is to assume one of the camera in canonical form $P_1 = [I|0]$ and $P_2 = [e'_\times F|e']$, where e' is the epipole. A more general formula is

$$P_1 = [I|0], P_2 = [e'_\times F + e'v^T|\lambda e'] \tag{2.28}$$

where v is any 3-vector, and λ is a non-zero scalar. Interested readers are referred to (Hartley and Zisserman, 2003) for the detailed derivation of equation 2.28.

As shown in equations 2.20 and 2.24, the essential matrix is applied on normalized camera coordinates. Therefore, the camera matrix $K^{-1}P = [R|T]$ derived from the essential matrix is called the normalized camera matrix. Retrieving normalized camera matrices from essential matrices is up to scale and has a four-fold ambiguity. The SVD of the essential matrix E is

$$E = U\Sigma V^T \tag{2.29}$$

According to equation 2.25 $E = [t_\times]R = SR$, S is skew-symmetric. Therefore, E can be factorized as

$$S = UZU^T, R = UWV^T \; or \; UW^TV^T \tag{2.30}$$

where $Z = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$ and $W = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$. Since $S = [t_\times]$, we can get $t = U(0,0,1)^T = u_3$. The sign of t cannot be determined, so the four possible solutions of normalized camera matrix P_2, given $P_1 = [I|0]$, are

$$P_2 = [UWV^T|u_3] \; or \; P_2 = [UWV^T|-u_3] \; or \; P_2 = [UW^TV^T|u_3] \; or \; P_2 = [UW^TV^T|-u_3] \tag{2.31}$$

Triangulation. Once the camera matrices are determined, the pairs of matches across different cameras can be reconstructed into 3D through the process called *triangulation.* Giving a set of matches $(\boldsymbol{x_1}, \boldsymbol{x_2}, \boldsymbol{x_3}, ...)$ and the corresponding camera matrices $(\boldsymbol{P_1}, \boldsymbol{P_2}, \boldsymbol{P_3}, ...)$, the most straightforward solution to triangulation is to minimize the distance between the reprojected 3D points and image points, which mathematically is

$$\arg\min_{\boldsymbol{X}} E = \min \sum_i ||x_i - P_i X||_2^2 \tag{2.32}$$

The optimal value for $\boldsymbol{X}$, which minimizes the sum of squared reprojection errors, can be computed as a regular least squares problem.

Bundle adjustment. The epipolar constraint and triangulation introduced above solves for camera pose and 3D points seperately using the direct linear transform (DLT) algorithm. However, there is a more accurate algorithm, called *bundle adjustment*, that can solve for camera poses and 3D points simutaneously as non-linear least squares. The bundle adjustment formulation is

$$\arg \min_{\boldsymbol{X},\boldsymbol{R},\boldsymbol{t},\boldsymbol{K}} = min \sum_i ||\boldsymbol{x_i} - f(\boldsymbol{X}, \boldsymbol{R}, \boldsymbol{t}, \boldsymbol{K})||_2^2 \tag{2.33}$$

This is a huge optimization problem, but the special structure of the formulation can be utilized to solve the problem much more efficiently (Triggs et al., 1999).

2.2 Introduction to deep neural networks

A major fraction of this dissertation (Chapters 4, 5 and 6) involves using deep neural networks (DNNs) to solve for computer vision and medical image analysis tasks. Therefore, in this section I present background knowledge to understand deep neural networks and their application in vision tasks.

2.2.1 Basic neuron in deep neural networks

The concepts and development of neural networks are mainly inspired by the goal of modeling the human brain. As with the human brain, the basic module in a neural network is called a *neuron* or *artificial neuron*. An early artificial neuron, called a *perceptron*, was developed by Rosenblatt et al. (Rosenblatt, 1958). A neuron usually takes multiple inputs; and those inputs are combined by different *weights*. These weights control the importance of different inputs and usually are learnable. Afterwards, the combined value will go through a function called an *activation function*. The activation function determines whether a neuron is "fired" or not; it also adds non-linearity to the neuron. One of the most common types of activation function is a sigmoid function. Figure 2.5 shows an example of the inputs, weights, activation function and outputs of a basic neuron. Mathematically the full process is

$$o = f(\boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{v} + b) \tag{2.34}$$

where $\boldsymbol{w}$ is a vector of the weights, $\boldsymbol{v}$ is a vector of the inputs, b is the bias and $f(\cdot)$ is the activation function.

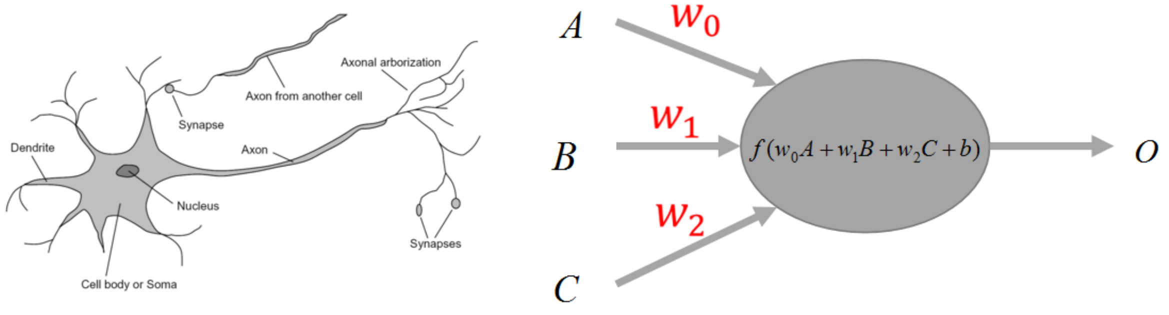


Figure 2.5: Example of a real and an artificial neuron.

2.2.2 Multi-layer neural networks

A basic neuron introduced in the previous section can only make a very simple decision. In order to make more complex decisions, a common way is to construct a graph using many neurons. To avoid infinite loops, the graph is acyclic and thus forms a layer-wise graph in which the outputs from one layer of neurons are the inputs to the next layer of neurons; we call such a graph a *multi-layer neural network*. Figure 2.6 shows a three-layer neural network. The first layer of neurons take inputs and make simple decisions;

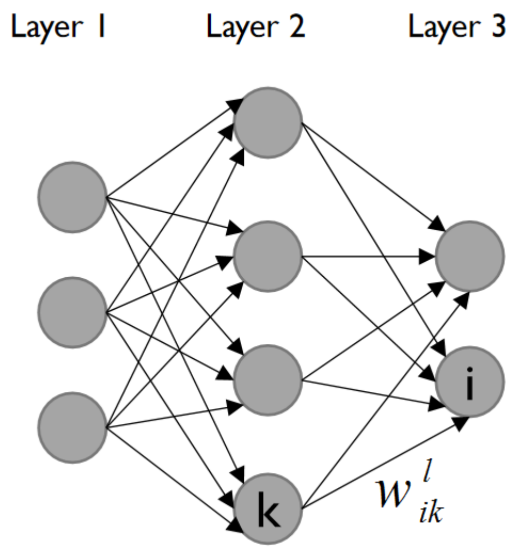


Figure 2.6: w^l_{ik} is the weight from the k^{th} neuron in $(l-1)^{th}$ layer to the i^{th} neuron in the l^{th} layer.

then the neurons in second layer take the results from the first layer and make decisions at a more complex and more abstract level. The level of complexity and abstraction increases as the network goes deeper; we also call a neural network with many layers a *deep neural network*.

2.2.3 Backpropagation

I have introduced the basic neuron and multi-layer neural networks. The operation at a single neuron i at layer l is

$$o^l_i = f(\sum_k w^l_{ik} v^{l-1}_k + b^l_i) \tag{2.35}$$

where the sum is across all the neurons k in the previous layer $l-1$. When there is such a connection with all the neurons in previous layer, the layer is called *fully connected*. The weights w and bias b are the learnable parameters. For a common learning algorithm, we usually need to construct a cost function, and the parameters can be learned by minimizing the cost function. *Backpropagation* is the algorithm that computes the partial derivative of the cost function C with respect to every weight and bias in the whole network. In a deep neural network that has many layers, backpropagation is basically a recursive application of the chain rule.

To demonstrate the basic steps in the backpropagation algorithm, let us assume that the cost function C is a simple quadratic function:

$$C = \sum_i (y_i - o_i^l)^2 \tag{2.36}$$

where y are the desired outputs and o^l (see equation 2.35) are the outputs of all the neurons from the final layer. For example, y might be the groundtruth labels for a classification task.

Let us forget the variables w and b for now, and let $q_i^l = \sum_k w_{ik}^l v_k^{l-1} + b_i^l$ be the only variable. Equation 2.35 simplifies to $o_i^l = f(q_i^l)$. Then the error backpropagated from the cost function to the neurons of the final layer of the network with respect to the variable q is then

$$\delta_i^l = \frac{\partial C}{\partial q_i^l} = \frac{\partial C}{\partial o_i^l} f'(q_i^l) \tag{2.37}$$

where f' is the derivative of the activation function.

Before applying the chain rule to w and b, let us first derive the error δ_i^{l-1}, that to be backpropagated to the neuron in the layer before the final layer:

$$\delta_i^{l-1} = \frac{\partial C}{\partial q_i^{l-1}} \tag{2.38}$$

By applying the chain rule we can get

$$\frac{\partial C}{\partial q_i^{l-1}} = \sum_j \frac{\partial C}{\partial q_j^l} \frac{\partial q_j^l}{\partial q_i^{l-1}} \tag{2.39}$$

where j denotes all the neurons in the final layer that take the output from the neuron i in the previous layer as input. Notice that $\delta_j^l = \frac{\partial C}{\partial q_j^l}$, so

$$\delta_i^{l-1} = \sum_j \delta_j^l \frac{\partial q_j^l}{\partial q_i^{l-1}} \tag{2.40}$$

Since

$$q_j^l = \sum_i w_{ji}^l v_i^{l-1} + b_j^l = \sum_i w_{ji}^l f(q_i^{l-1}) + b_j^l \tag{2.41}$$

we can derive

$$\frac{\partial q_j^l}{\partial q_i^{l-1}} = w_{ji}^l f'(q_i^{l-1}) \tag{2.42}$$

Putting this back into Equation 2.40, we can get the error to be backpropagated to the previous layer given the error in the current layer as

$$\delta_i^{l-1} = \sum_j \delta_j^l w_{ji}^l f'(q_i^{l-1}) \tag{2.43}$$

With equation 2.43 we know how to compute the error with respect to variable q for the previous layer given the error in the current layer. This can be recursively applied from the layer before the final layer all the way to the first layer in a deep neural network.

Now let us derive the error to be backpropagated with respect to w and b giving the cost function C:

$$\frac{\partial C}{\partial b_i^l} = \frac{\partial C}{\partial o_i^l} f'(q_i^l) \frac{\partial q_i^l}{\partial b_i^l} \tag{2.44}$$

According to equation 2.41, $\frac{\partial q_i^l}{\partial b_i^l} = 1$, so

$$\frac{\partial C}{\partial b_i^l} = \frac{\partial C}{\partial o_i^l} f'(q_i^l) = \delta_i^l \tag{2.45}$$

Similarly for w we have

$$\frac{\partial C}{\partial w_{ik}^l} = \frac{\partial C}{\partial o_i^l} f'(q_i^l) \frac{\partial q_i^l}{\partial w_{ik}^l} = \frac{\partial C}{\partial o_i^l} f'(q_i^l) f(q_k^{l-1}) = \delta_i^l o_k^{l-1} \tag{2.46}$$

With the derived equation for $\frac{\partial C}{\partial w_{ik}^l}$, $\frac{\partial C}{\partial b_i^l}$, δ_i^{l-1} and δ_i^l, the backpropagation algorithm recursively updates every weight and bias by subtracting the derivatives of error in the neural network layer by layer from the last layer to the first layer.

2.2.4 Convolutional neural networks

There is a special type of deep neural network, called a *convolutional neural network*, that is designated for images as inputs. As introduced above, the basic operation at a neuron is a weighted sum of the inputs followed by an activation function. Convolution, as one of the most commonly used operations in computer vision and image processing, is nothing but a weighted sum of neighboring pixels. The weights are called the kernel, and this kernel is shift-invariant. Therefore, when we have an image as the input to a neural network and let the input to a neuron be only a local neighbor of the output of the neurons from the previous layer, the neural network becomes a convolutional neural network. Furthermore, similar to convolution, each kernel is convolved across the whole image in a convolutional neural network. This not only makes the basic operation in a layer of the neural network truly mimic the convolutional operation in image processing but also allows more efficient implementation and vastly reduces the amount of parameters in the network. A layer that usually follows the convolutional layer to reduce the spatial dimension of the output is called a *pooling layer*.

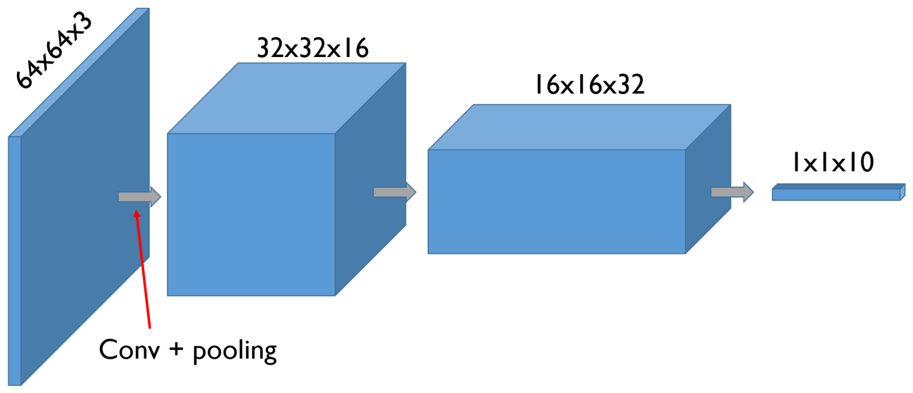


Figure 2.7: An example of a 3-layer convolutional neural network.

Figure 2.7 shows an example of a simple 3-layer convolutional neural network. The number of neurons are determined by the number of kernels. In this example the input image is $(64 \times 64 \times 3)$, and the first layer has 16 $(3 \times 3 \times 3)$ kernels, which results in $32 \times 32 \times 16$ neurons.

2.2.5 Recurrent neural networks

So far, the neural networks that I have introduced only work for spatial data such as an image. However, in order to make the neural networks also work for temporal sequences such as video or language, people have designed a special type of network called *recurrent neural networks* (RNN).

A key difference between a vanilla neural network and an RNN is that the RNN can work on input data with varying sizes whereas the vanilla neural network can only work on input data with fixed size.

As shown in Figure 2.5 and equation 2.35, a neuron in a vanilla neural network takes its input from only the outputs of neurons in the previous layer. A neuron in an RNN, on the other hand, takes its input from both the outputs of the neurons in the previous layer and the output of the same neuron at the previous time step. In addition, a neuron in an RNN usually has an internal memory in order to process arbitrary length sequences.

Figure 2.8 shows an example of a simple RNN. The connection from a neuron to itself demonstrates the temporal connection from the previous time step to the current time step.

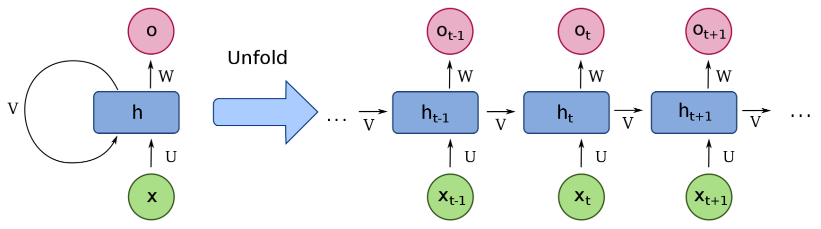


Figure 2.8: Example of an RNN and its unfolded version.

There are different designs for the internal memory in RNNs. Two of the most common types are *long short-term memory* (LSTM) and gated recurrent units (GRU). The operation at a single neuron with LSTM is

$$\begin{aligned}
i_t &= f(\boldsymbol{w_{vi}^T v_t} + \boldsymbol{w_{hi}^T h_{t-1}} + \boldsymbol{w_{ci}^T c_{t-1}} + b_i) \\
g_t &= f(\boldsymbol{w_{vg}^T v_t} + \boldsymbol{w_{hg}^T h_{t-1}} + \boldsymbol{w_{cg}^T c_{t-1}} + b_g) \\
c_t &= g_t \circ c_{t-1} + i_t \circ tanh(\boldsymbol{w_{vc}^T v_t} + \boldsymbol{w_{hc}^T h_{t-1}} + b_c) \\
o_t &= f(\boldsymbol{w_{vo}^T v_t} + \boldsymbol{w_{ho}^T h_{t-1}} + \boldsymbol{w_{co}^T c_t} + b_o) \\
h_t &= o_t \circ tanh(c_t)
\end{aligned} \tag{2.47}$$

where $\circ$ denotes the Hadamard product and where i_t is the "input gate", g_t is the "forget gate", c_t is the "cell state", o_t is the "output gate" and h_t is the "final state". In short, the activation function in i_t and g_t make them have a value close to either 0 or 1, i.e., act like gates. Then the current cell state C_t is a gated combination of the previous cell state and the current candidate value. Again, the final output h_t is further controlled by the output gate o_t.

2.3 Image simulation for computer vision

Photo-realistic simulation is very important for many computer vision tasks. For example, self-driving cars require very many driving tests on all kinds of road and weather conditions; it is almost impossible to accomplish this in real world scenarios. In this case, computer graphics based simulation can create all different environments so that the self-driving algorithms can be tested thoroughly in these synthesized environments. Besides self-driving cars, image simulation has also been widely used for training deep neural networks, evaluating 3D reconstruction algorithms and more. Theoretically, using synthetic data has the advantage of 1) full control of the data generation pipeline; 2) lower costs; 3) greater flexibility and variability; and 4) limitless quantity. Of course, adequate realism is a challenge for these simulators.

2.3.1 Image simulation for deep learning

It is well known that deep neural networks require millions of varied and annotated data for training. However, in many cases, such as medical imaging, a large amount of labeled data is impossible to acquire. Therefore, synthetic data becomes an increasingly popular tool for training deep learning models, especially in computer vision. One of the fields in computer vision that is hard to obtain accurate groundtruth data is optical flow estimation. Therefore, Dosovitsky et al. (Dosovitskiy et al., 2015) produced a large synthetic dataset called Flying Chairs from a public database of 3D chair models, adding them on top of real backgrounds to train a CNN-based optical flow estimation model. Another field is semantic segmentation, for which synthetic images and semantic labels can be easily generated from virtual 3D environments. Virtual KITTI (Gaidon et al., 2016) and SYNTHIA (Ros et al., 2016) are two commonly used synthetic datasets for training semantic segmentation networks. The Virtual KITTI dataset is generated using a commercial computer graphics engine Unity. Unity has a public Assets Store, where realistic 3D models and materials of objects are available. The Virtual KITTI dataset was created as a synthetic clone of a real world dataset KITTI (Geiger et al., 2013), so any deep neural network trained on KITTI can be tested on Virtual KITTI with slightly modified weather conditions to evaluate their generalizabilty. Figure 2.9 shows an example of the Virtual KITTI dataset compared with the KITTI dataset. The SYNTHIA dataset is similar to the Virtual KITTI dataset in that it is generated using the Unity development platform and mainly contains urban scenes.

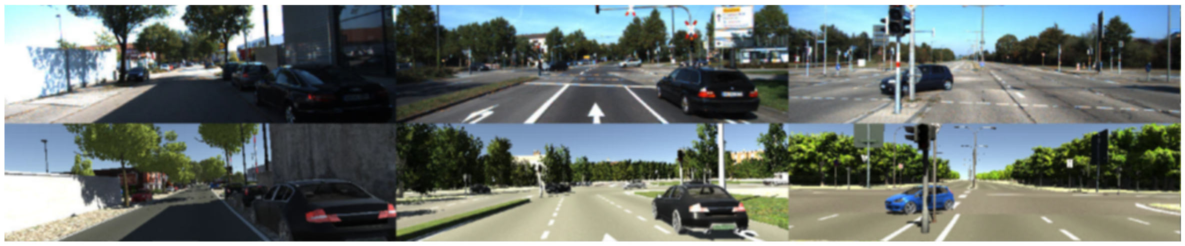


Figure 2.9: Examples from the virtual KITTI dataset compared with counterparts in the KITTI dataset.

2.3.2 Image simulation for medical image analysis

Large, accurately annotated medical imaging datasets are hard to acquire due lack of experts available for annotations, limited data for rare cases, lack of standardization, and privacy issues. Therefore, the medical image community has been fascinated by creating simulated or synthesized datasets to validate image analysis and reconstruction algorithms. With the fast development of deep learning based medical imaging methods, such demands further increased. Pfaehler (Pfaehler et al., 2018) developed a Positron-emission tomography (PET) simulator, called SMART-PET, for development and performance evaluation of segmentation methods. SMART-PET is a standalone program written in Interactive Data Language (IDL). SMART-PET can produce PET images that are comparable to actual phantom data. It requires a 3D PET image representing the 'true' activity distribution and a 3D attenuation map or a CT image of the same object with corresponding image dimensions.

Besides the analytic simulation techniques, there are also machine learning based simulators. Zhao et al. (Zhao et al., 2018) proposed a generative adversarial network (GAN) that can generate synthesised retinal and neuronal images. Their generative network takes a tubular structured annotation as input and produces a raw RGB image; their discriminator in turn tries to separate the real images from the synthetic ones. Advanced computer graphics tools are also used for simulating medical images. Freedman et al. (Freedman et al., 2020) generated a synthetic colon model using the colon simulator developed by 3D Systems. This model was then rendered into a synthetic colonoscopic video using Blender (Community, 2018). Mahmood et al. (Mahmood and Durr, 2018) also generated a synthetic colon phantom using Blender. The virtual colon had anatomically realistic diameters, bending angles and polyps.

Chapter 3

Fusion-guided Structure-from-motion-and-shading for 3D Reconstruction from Pharyngoscopic Videos

Endoscopy enables high-resolution visualization of tissue texture and is a critical step in many clinical workflows, including diagnosis and treatment planning for cancers in the nasopharynx. However, an endoscopic video does not provide its information in 3D space, making it difficult to use in tumor localization, and it is inefficient to review. Therefore, we developed a novel 3D reconstruction method that given an input pharyngoscopic video sequence, reconstructs the throat surface as a textured 3D mesh named an *endoscopogram* (Zhao et al., 2016). The endoscopogram is generated by first reconstructing a textured 3D partial surface for each frame. Then these multiple partial surfaces are fused into an endoscopogram using a groupwise surface registration algorithm and a seamless texture fusion from the partial surfaces. Finally, the endoscopogram geometry is registered with the surface extracted from CT, thereby enabling the desired tumor transfer process. In this chapter I will present the details of the aforementioned steps for producing the endoscopogram. The frame-by-frame partial surface reconstruction algorithm that was developed by Price et al. is introduced in Section 3.2. The groupwise surface registration algorithm developed by Zhao et al. (Zhao et al., 2016) is introduced in Section 3.3. My contributions are 1) introduced in Section 3.4, a novel combination of these two methods that improves the temporal consistency of the frame-by-frame 3D reconstruction algorithm and eliminates the manual intervention that was needed in the deformable surface registration method; 2) introduced in Section 3.5, a seamless texture fusion algorithm that produces complete and high quality texture for an endoscopogram; 3) introduced in Section 3.6, a simulation-based evaluation method; and 4) introduced in Section 3.7, an interactive tool for tumor drawing and transferring between endoscopogram and CT.

3.1 Background

To date, most work on combining motion-based reconstruction with shading information has utilized shading to augment an existing shape template or model priors (Salzmann and Fua, 2010). Wu *et al.* (Wu et al., 2011) proposed first building coarse-scale dynamic models from multi-view video and then leveraging shading appearance to estimate fine-scale, temporally varying geometry. Fine-scale shading correction has also been used to refine dense surfaces obtained via depth sensor (Han et al., 2013; Zollhöfer et al., 2015). In endoscopic applications, a related method by Tokgozoglu *et al.* (Tokgozoglu et al., 2012) used multi-view stereo to derive a low-frequency model of the upper airway and then applied Lambertian shape-from-shading (SfS) on albedo-normalized images to endow the existing surface with higher-resolution shape. For monocular reconstruction of deforming environments, several efforts have been made to extend the Shape-from-Template problem (Bartoli et al., 2015) to utilize shading information. In (Malti et al., 2011; Malti et al., 2012; Malti and Bartoli, 2014), Malti, Bartoli, and Collins proposed a two-stage approach for surgery of the uterus: Pre-surgery an initial 3D template is recovered under rigid scene assumptions, and reflectance parameters are estimated for the surface. In surgery the deforming surface is recovered via conformal deformations of the template surface, and subsequent shading refinement is performed using the estimated reflectance model. But we address the problem of dense reconstruction in conditions where dense shape templates are unavailable or difficult to derive. Laryngoscopy is a good example of this (Figure 3.1) because the anatomic shapes in this region are highly patient-specific and as compared to surfaces extracted from endoscopy, those extracted from CT scans are typically low-resolution and have a notably different shape. Multi-view stereo also tends to fail in this scenario, as the combination of strong illumination changes and limited non-deforming image sequences is problematic. Motivated by our observation that structure-from-motion (SfM) works over short temporal sequences for these data, we have developed a method for dense single-view surface estimation that leverages sparse 3D geometry obtained from SfM. Section 3.1.1

discusses the background for SfM, and Section 3.1.2 discusses the background for SfS.

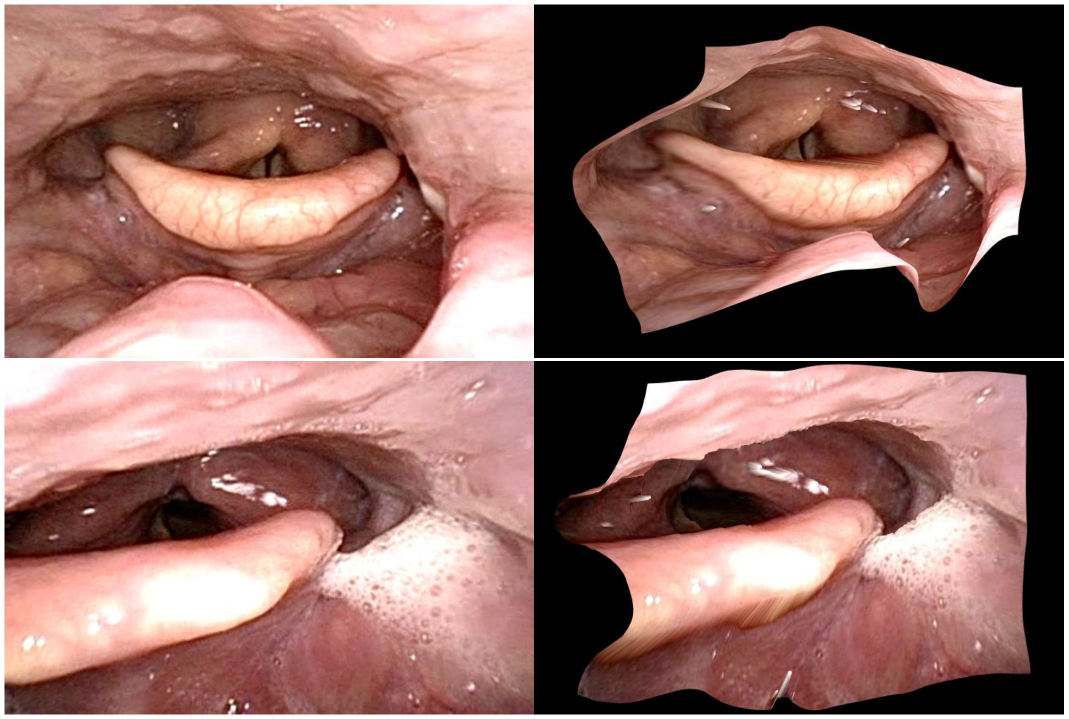


Figure 3.1: Example results of SfMS on live endoscopy from two different patients. Left: Original image. Right: Surface estimated from the image using our algorithm.

Both the fusion of partially reconstructed surfaces and bringing the fused surface into CT coordinates requires non-rigid surface-to-surface registration. Non-rigid 3D registration has been a common topic in medical image analysis. Surface embedding is one class of surface-to-surface registration methods. (Elad and Kimmel, 2003; Bronstein et al., 2006; Beardsley et al., 1996; Dellaert et al., 2000) proposed a multidimensional scaling embedding method that can place the two surface vertices in a low-dimensional Euclidean space, where a nearest-neighbor matching method can be performed to yield the correspondences. Gu (Gu et al., 2004) proposed to use conformal mapping, which is angle-preserving, to embed the surfaces into a common disc or sphere domain. However, such methods require the surfaces to have the same intrinsic geometry, so it cannot handle surface topology change or missing patches. Matching-based methods (Sun et al., 2009; Gatzke et al., 2005; Zaharescu et al., 2009) use hand-crafted feature descriptors to perform feature matching, which produce a set of corresponding points. However, without any regularization the outliers produced in the feature matching will lead to non-smooth or even incorrect deformations. Zeng (Zeng et al., 2013) proposed to use an MRF to regularize the deformation field. (Bauer and Bruveris, 2011) have provided an elegant mathematical framework (called LDDMM) that produces diffeomorphic deformations between surfaces by comparing their normal fields. Thirion (Thirion, 1998) proposed a Demons algorithm which optimize a per pixel displacement field. The forces that apply on each pixel were inspired from the optical flow equations. The idea of the Demons algorithm is appealing because it has no assumptions about the surface properties. (Both LDDMM and Demons are developed for image-to-image registration, then adapt to surface). Section 3.3 discusses the novel surface-to-surface registration method named thin-shell-demons, invented by my colleagues, Zhao et al. (Zhao et al., 2016).

3.1.1 Structure-from-motion

SfM (Schönberger and Frahm, 2016; Mohr et al., 1995; Dellaert et al., 2000; Pollefeys et al., 2004) is the simultaneous estimation of camera motion and 3D scene structure from multiple images taken from different viewpoints. Typical SfM methods produce a sparse scene representation by first detecting and matching local features in a series of input images, which are the individual frames of the endoscope video in our application. Then, starting from an initial two view the essential matrix or fundamental matrix is computed, camera matrices are retrieved and 3D points are triangulated, which are detailed in Chapter 2, Section 2.1.5. Afterwards, new images are registered to the existing model to incrementally estimate both camera poses (orientation and position for each image) and scene structure. The scene structure is parameterized by a set of 3D points projecting to corresponding 2D image features.

Our motivation for using SfM is that it provides a prior on depth, albeit at sparse locations, that provides constraints for surface geometry and reflectance model estimation. Figure 3.2 shows an example SfM reconstruction of endoscopic data using several segments from the overall video. One limitation to the generality of our method is that sparse non-rigid reconstruction in medical settings is an unsolved problem (Gotardo and Martinez, 2011; Kong and Lucey, 2019). However, the approach we propose can handle any sparse data as input, so the method could easily be integrated with non-rigid SfM in future work.

3.1.2 Shape-from-shading

SfS, first introduced in the 1970 thesis of Horn (Horn, 1970), is a monocular method of depth estimation that, given a single image viewing a scene, recreates the three-

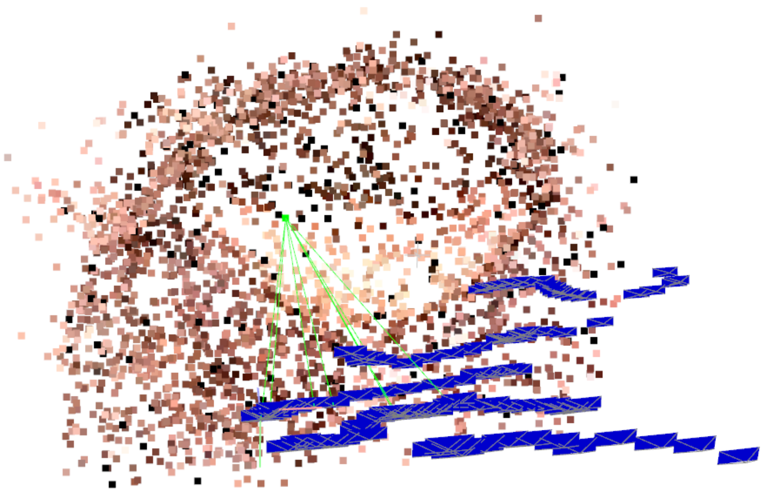


Figure 3.2: Structure-from-Motion results for endoscopic video. Individual 3D surface points (colored dots) and camera poses (blue) are jointly recovered.

dimensional shape of the scene under given assumptions about the lighting conditions and surface reflectance properties (Zhang et al., 1999; Prados and Faugeras, 2006; Durou et al., 2008). A number of different formulations have been proposed to solve the SfS problem, including energy minimization, recovery of depth from estimated gradient, local shape estimation, and modeling as a partial differential equation (PDE) (Zhang et al., 1999; Durou et al., 2008). Over the last decade, the PDE formulation of SfS has received the most attention, starting with Prados and Faugeras (Prados and Faugeras, 2005), who introduced a novel, provably convergent approach for solving the problem as a PDE.

Our primary motivation for using SfS is that many of its simplifying assumptions are well suited for general endoscopic devices. In particular, use of an endoscope allows us to assume a co-located camera and light source, which greatly simplifies the modeling of surface reflectance in the scene. We next describe what this simplification entails, which sets the stage for introducing our proposed reflectance model.

3.2 Structure-from-motion-and-shading

In this section I will introduce our frame-by-frame reconstruction method developed by Price et al., which is based on a new Shape-from-Shading formulation that utilizes the sparse, but accurate, 3D point data obtained via Structure-from-Motion. First, we introduce a regularized formulation of SfS that allows for a trade-off between predicted image intensity and similarity to an existing estimated surface. We also suggest a way to account for errors along occlusion boundaries in the image using intensity-weighted finite differences. Second, we propose a general reflectance model for use in our SfS framework that can more accurately capture real-world illumination conditions. Finally, we develop an iterative update scheme that at each iteration (1) warps an estimated surface to the SfM point cloud, (2) estimates a reflectance model using this warped surface and the given image, and (3) produces a new estimated surface using the regularized SfS method.

Reflectance Models. The amount of light reflecting from a surface can be modeled by a wavelength-dependent Bidirectional Reflectance Distribution Function (BRDF) that describes the ratio of the radiance $I_{\lambda r}$ of light reaching the observer to the irradiance $E_{\lambda r}$ of the light hitting the surface. Generally, a BRDF is given as a function of four variables: the angles (θ_i, ϕ_i) between the incident light beam and the normal, and the reflected light angles (θ_r, ϕ_r) with the normal; that is,

$$\mathrm{BRDF}_{\lambda}(\theta_i, \phi_i, \theta_r, \phi_r) = \frac{I_{\lambda r}}{E_{\lambda i}}, \tag{3.1}$$

where λ represents light wavelength. In the following we implicitly assume the wavelength dependence of the BRDF.

The irradiance for an incoming beam of light is itself a function of θ_i and the distance r to the light source:

$$E_i = I_i \frac{A}{r^2} \cos\theta_i, \tag{3.2}$$

where I_i is the light source intensity and A is the projected area of the light source.

We make two simplifying assumptions about the BRDF that help the overall modeling of the problem. First, we assume surface isotropy of the BRDF, which constrains it to only depend on the relative azimuth, $\Delta\phi = |\phi_i - \phi_r|$, rather than the angles themselves (Koenderink et al., 1996).

Second, we assume that the light source is approximately located at the camera center relative to the scene, which is a reasonable model for many endoscopic devices. In this case, the incident and reflected light angles are the same, $(\theta_i, \phi_i) = (\theta_r, \phi_r)$. Under these assumptions, the observed radiance simplifies to

$$I_r(r, \theta_i) = I_i \frac{A}{r^2} \cos(\theta_i)\mathrm{BRDF}(\theta_i). \tag{3.3}$$

The reflectance model we propose is based on the set of BRDF basis functions introduced by Koenderink *et al.* (Koenderink et al., 1996). These functions form a complete, orthonormal basis on the hemisphere derived via a mapping from the Zernike polynomials, which are defined on the unit disk.

We adapt the BRDF basis of Koenderink *et al.* to produce a multi-lobe reflectance model for camera-centric SfS. First, taking the light source to be at the camera center, we have $\theta_i = \theta_r$ and $\Delta\phi_{ir} = 0$; this gives

$$\mathrm{BRDF}(\theta_i) = \sum_{k=0}^{K-1} \left(\alpha_k + \beta_k \sin\left(\frac{\theta_i}{2}\right)\right) \cos^k \theta_i, \tag{3.4}$$

where α_k and β_k are coefficients that specify the BRDF.

Surface Model. Let $(x, y) \in \Omega$ represent image coordinates after normalization by the intrinsic camera parameters (centering around the principal point and dividing by the focal length). For a given camera pose, the surface function $f : \Omega \to \mathbb{R}^3$ maps points in the image plane to 3D locations on a surface viewed by the camera. Under perspective projection,

$$f(x, y) = z(x, y) \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}, \tag{3.5}$$

where $z(x, y) > 0$ is a mapping from the image plane to depth along the camera's viewing axis. The distance r from the surface to the camera center is

$$r(x, y) = \|f(x, y)\| = z(x, y)\sqrt{x^2 + y^2 + 1}, \tag{3.6}$$

and the normal to the surface is defined by the cross product between the x and y derivatives of f:

$$\mathbf{n}(x, y) = f_x \times f_y = z \begin{pmatrix} -z_x \\ -z_y \\ xz_x + yz_y + z \end{pmatrix}. \tag{3.7}$$

Given a co-located light source, the light direction vector for a point in the image is the unit vector $\hat{\mathbf{l}}(x, y) = \frac{1}{\sqrt{x^2+y^2+1}}(x, y, 1)$. The cosine of the angle between the normal and light direction vectors, i.e., the term $\cos\theta_i$ in Eq. 3.3, is then equal to their dot product:

$$\cos\theta_i = \hat{\mathbf{n}} \cdot \hat{\mathbf{l}} = \frac{z}{\sqrt{(x^2+y^2+1)\left(z_x^2+z_y^2+(xz_x+yz_y+z)^2\right)}}, \tag{3.8}$$

where "ˆ" indicates normalization to unit length.

Prados and Faugeras (Prados and Faugeras, 2005) note that Eq. (3.8) can be simplified using the change of variables $v(x, y) = \ln z(x, y)$:

$$\hat{\mathbf{n}} \cdot \hat{\mathbf{l}} = \frac{1}{\sqrt{(x^2+y^2+1)\left(v_x^2+v_y^2+(xv_x+yv_y+1)^2\right)}}; \tag{3.9}$$

this expression involves only derivatives of v. In our shading model, more specifically in the $\cos\theta_i$ factor in Equation 3.3, this transformation allows us to separate factors involving v from those involving its derivatives.

Adapted PDE framework

In the following, we modify the traditional SfS PDE to include regularization against a pre-existing estimated surface. Then, we address an implementation for solving this regularized SfS equation. Finally, we propose the use of weighted finite differences to mitigate the effect of in the implementation's smoothness assumptions that cause inaccurate depth measurements along surface occlusion boundaries.

Original PDE. Eq. (3.3) models observed intensity for a generic, isotropic BRDF with the assumption that the light source is co-located with the camera. Joining this with Eqs. (3.6) and (3.9) and multiplying by r^2, we have

$$(x^2+y^2+1)I_r e^{2v} - I_i A \cos(\theta_i)\mathrm{BRDF}(\theta_i) = 0 \tag{3.10}$$

(note $e^{2v} = z^2$). This is a static Hamilton-Jacobi equation of the form

$$\begin{cases} Le^{2v} - H(v_x, v_y) = 0, & (x, y) \in \Omega \\ v(x, y) = \psi(x, y), & (x, y) \in \partial\Omega, \end{cases} \tag{3.11}$$

where the dependence of H and L on x and y is implied. $\psi(x, y)$ specifies boundary conditions for the PDE.

Regularized Equation. The PDE introduced above is dependent on the accuracy of the BRDF modeling the scene. To prevent surface mis-estimations arising from an inaccurate BRDF, we use the 3D points obtained from SfM as an additional set of constraints for our estimated log-depths, v.

To the SfS PDE (Eq. (3.11)) we add a simple regularization that constrains the solution to be similar to a warped surface generated from the 3D SfM points. Specifically, instead of a proper PDE, we consider the following energy function:

$$E(v) = \frac{1}{2}\left(e^{2v} - \frac{1}{L}H\left(v_x, v_y\right)\right)^2 + \frac{\lambda}{2}\left(e^{2v} - z_{\text{warp}}^2\right)^2, \tag{3.12}$$

where $z_{\text{warp}}(x, y)$ is the depth of the warped surface at a given image coordinate and the parameter $\lambda(x, y) \geq 0$ controls the influence of the right term, which regularizes on depths. We show how to choose the value of λ below when we introduce our iterative algorithm. Minimizing $E(v)$ w.r.t v, we obtain

$$\frac{\partial E}{\partial v} = \left[e^{2v} - \frac{1}{1+\lambda}\left(\frac{1}{L}H(v_x, v_y) + \lambda z_{\text{est}}^2\right)\right]2e^{2v} = 0. \tag{3.13}$$

Incorporating boundary conditions, we have the following optimization problem:

$$\begin{cases} e^{2v} - \frac{1}{1+\lambda}\left(\frac{1}{L}H(v_x, v_y) + \lambda z_{\text{est}}^2\right) = 0 & (x, y) \in \Omega \\ v(x, y) = \psi(x, y). & (x, y) \in \partial\Omega. \end{cases} \tag{3.14}$$

Solving the Regularized SfS Equation. To solve our regularized SfS equation, we employ the fast-sweeping method proposed for SfS by Ahmed and Farag (Ahmed and Farag, 2006), itself based on a method by Kao *et al.* (Kao et al., 2004). This approach uses the Lax-Friedrichs (LF) Hamiltonian, which provides an artificial viscosity approximation for solving static Hamiltonian-Jacobi equations.

At a high level, the algorithm presented in (Ahmed and Farag, 2006) initializes the log-depth values $v(x, y)$ to a large positive constant and proceeds to iteratively update these values to progressively closer depths. We refer the reader to (Ahmed and Farag, 2006) for the full algorithm of the fast-sweeping scheme as the order of sweeping directions, treatment of boundary conditions, and convergence criterion presented in (Ahmed and Farag, 2006) are the same as for our method.

Iterative Update Scheme

We now describe our iterative updating scheme. Our method has an EM flavor in the sense that it iterates a step optimizing a set of parameters (the reflectance model) based on the existing surface followed by a step computing expected depths using these parameters.

```
Algorithm 1: Shape-from-Motion-and-Shading
  Input: An endoscopic image $F_i$ and the associated 3D SfM points $C_i$
  1. Warping $S^n_{warp}(x,y) = \rho(x,y) S^n_{warp}(x,y)$
  2. Reflectance model estimation $E(\Theta) = \sum_\Omega (I_r(x,y) - I_{\text{est}}(x,y;\Theta))^2$
  3. Solve the SfS PDE using the estimated reflectance model parameters $\Theta$ and
     the warped surface $S^n_{warp}$ to generate a newly estimated surface $f_{n+1}$
  4. Re-warp $f_{n+1}$ and repeat steps 1-3
```

The proposed algorithm takes as input an observed image and the 3D SfM points associated with that image. It outputs a dense surface using depth-correcting warpings and the computed reflectance model.

Warping. We denote the warped surface at iteration n of our scheme as S^n_{warp}. For initialization, we define an estimated surface S^0_{warp} having $r(x,y) = 1$, where r is defined in Eq. (3.6). First, we perform an image-space warp of S^n_{warp} using the 3D SfM points with known distance $\hat{r}_i(x_i, y_i)$ as control points. For each SfM point, we estimate the ratio $\rho_i = \hat{r}_i / r_i$, where r_i is the point's (bilinearly interpolated) distance on S_n. To minimize the effect of outlier points from SfM, we adopt a nearest-neighbor approach to define our warping function: For each pixel (x,y) in the image, we compute the N closest SfM points in the image plane. In our experiments we use $N = 10$. Then, we define the warp function at that pixel as $\rho(x,y) = \sum w_i \rho_i / \sum w_i$, where the sums are over the neighboring SfM points. We set $w_i = \exp(-d_i)$, where d_i is the distance in the image plane between (x,y) and the SfM point (x_i, y_i). The new surface is calculated as $S^n_{warp}(x,y) = \rho(x,y) S^n_{warp}(x,y)$.

Reflectance Model Estimation. From this warped surface, we optimize the reflectance model parameters Θ for the specified BRDF (where the parameters depend on what BRDF we choose). This optimization is done by minimizing the least-squares error

$$E(\Theta) = \sum_\Omega \left(I_r(x,y) - I_{\text{est}}(x,y;\Theta)\right)^2, \tag{3.15}$$

where $I_{\text{est}}(x, y; \Theta)$ is the estimated image intensity (see Eq. (3.3)) determined by S^n_{warp} and the estimated BRDF.

SfS. Following reflectance model estimation, we apply the PDE framework introduced above (Eq. (3.14)) using the warped surface S^n_{warp} for values of z_{est} and using the current estimated reflectance model in computing H.

Concerning values of $\lambda(x, y)$ in our PDE, $\lambda > 1$ will give greater weight to S^n_{warp}, while $\lambda < 1$ will favor a purely SfS solution. We decide the weighting based on agreement between the SfM points and S^n_{warp}. Let Δr_i be the distance between a 3D SfM point with distance $\hat{r}_i$ and its corresponding point on S^n_{warp}. We compute the agreement between the warped surface and the SfM point as $\lambda_i = \log_{10} \hat{r}_i / 2\Delta r_i$. This equally weights SfM and SfS ($\lambda_i = 1$) when Δr_i is 5% of $\hat{r}_i$. The log term serves to increase λ_i by 1 for every order-of-magnitude decrease in $\Delta r_i / \hat{r}_i$. Just as for $\rho(x, y)$ above, we use the same nearest-neighbor weighting scheme to define $\lambda(x, y)$ based on the λ_i values at the SfM control points.

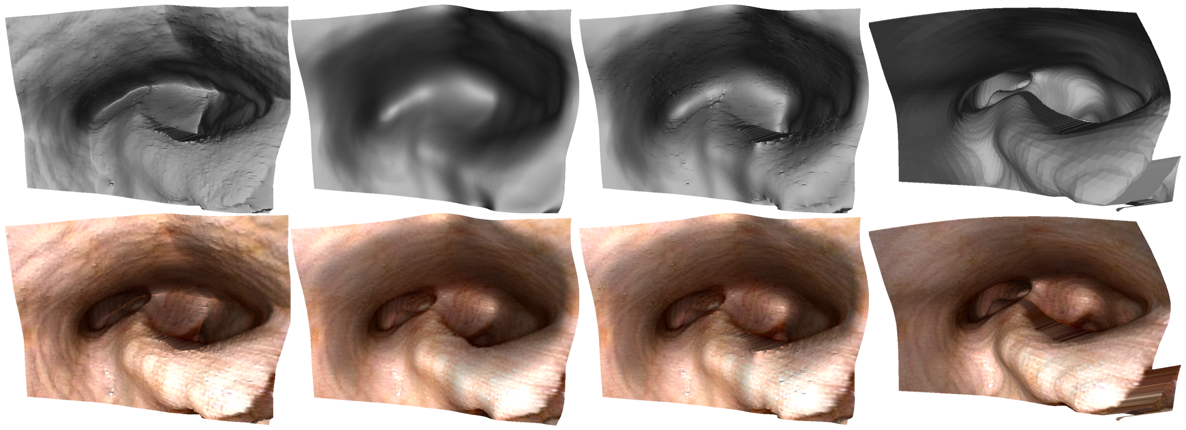


Figure 3.3: Visual comparison of surfaces generated by our approach for an image from our ground truth dataset. Top/bottom rows: Visualizations of the surface without/with texture from the original image. Columns from left to right: (1) using a Lambertian BRDF, (2) using our proposed BRDF ($K = 2$) without image-weighted derivatives, (3) using our proposed BRDF ($K = 2$) with image-weighted derivatives, and (4) the ground-truth surface. Note the oversmoothing along occlusion boundaries in column 2 versus column 3.

Iteration. Once SfS has been performed, we have a newly estimated surface S^{n+1}_{est}. Then, we simply re-warp the surface, re-estimate the reflectance model, and re-run regularized SfS. This iterative process is repeated for a maximum number of iterations or until convergence.

Results. Results of the SfMS method are illustrated in Figure 3.3.

3.3 Deformable surface registration

To fuse multiple frame-by-frame 3D reconstructions from SfMS into an endoscopogram, we use a novel groupwise surface registration algorithm involving N-body interaction. This algorithm is described in (Zhao et al., 2016) and is based on Zhao *et al.* (Zhao et al., 2015)'s pairwise surface registration algorithm, Thin Shell Demons (TSD). Here we only give an overview.

Thin Shell Demons

Thin Shell Demons is a physics-motivated method that uses geometric virtual forces and a regularizing thin shell model to estimate surface deformation. The geometric virtual forces $\{f\}$ are defined as vectors connecting vertex pairs $\{u^k, v^k\}$ between two surfaces $\{S_1, S_2\}$ (we use k here to index correspondences). The correspondences are automatically computed using geometric and texture features. The thin shell model is a physical model that regularizes the non-parametric deformation vector field $\phi : S_1 \rightarrow S_2$. Combining these two, the algorithm is defined as an iterative energy minimization function

$$E(\phi) = \sum_{k=1}^{M} c(v^k)(\phi(v^k) - f(v^k))^2 + E_{shell}(\phi), \tag{3.16}$$

where $c(v^k)$ is the confidence score based on the feature distance and E_{shell} is the thin shell deformation energy.

N-body Surface Registration

The endoscopogram requires registration of multiple partial surfaces. As an extension to the pairwise Thin Shell Demons, Zhao (Zhao et al., 2016) proposed a groupwise deformation scenario in which: N surfaces are deformed under the influence of their mutual forces. As shown in Figure 3.4, mutual forces are defined as virtual forces that attract one surface by all the other surfaces. In other words, the deformation of a single surface is determined by the overall forces exerted on it. Such groupwise attractions bypass the need of a target mean.

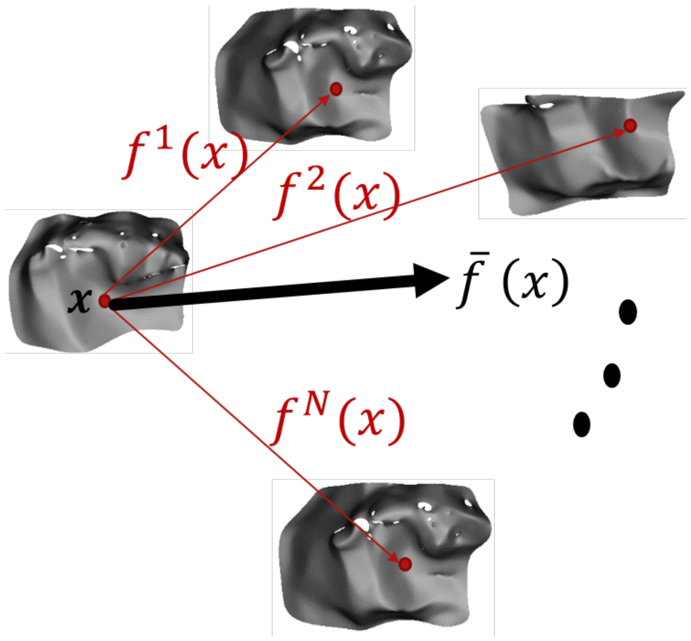


Figure 3.4: Each surface is attracted by virtual forces from all the other surfaces.

3.4 Fusion-guided SfMS

3.4.1 Improving shape reconstruction and reflectance model estimation by fusion

The aforementioned SfMS method together with the groupwise TSD can reconstruct a textured interior tissue surface in 3D that provides (1) complete 3D anatomical geometry, thereby facilitating tumor localization; (2) efficient visualization, thereby providing a full overview of the scoped area and providing comparison within and between patients; and (3) the opportunity to register endoscopy data with other modalities, such as CT, thereby enabling transfer of the tumor information into CT spaces for treatment planning. However, the combined method is still far from perfect: (1) Since the reconstruction method of Price et al. (Section 3.2) is frame-by-frame, there are no temporal constraints between successive images, leading to inconsistent reconstructions and even failure to reconstruct some frames. (2) Due to such inconsistency, very few partial surface reconstructions can be selected for fusion, and therefore, careful manual selections are needed for the groupwise TSD fusion.

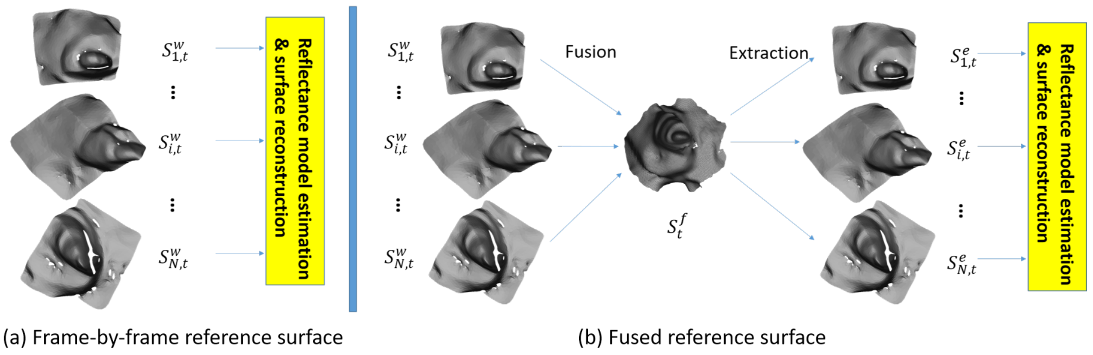
We assume that even if the tissues in endoscopic video are deformable, in adjacent frames they should still be quite close to each other. Therefore, we expect the reconstructed surfaces from adjacent frames to have small deformations from each other. In other words, we want the reconstruction to be consistent across time. In the original SfMS method each frame uses its own SfM points to generate a warped surface, which is used as a prior shape for reflectance model estimation. During the iterative reconstruction the method makes no interconnection between different frames. Therefore, initialization errors easily lead to different reconstruction results. I solve this problem by introducing a fused reference surface. This fused reference surface can be seen as a summary estimation from multiple frames, thereby being more robust than a single frame estimation. In addition, by leveraging the deformable registration and outlier geometry trimming in the geometry fusion, this fused reference surface is much more reliable than a simple average. Finally, all the frames use this fused reference surface to estimate their reflectance models and guide the SfS reconstruction as well. Our experimental results show that this fused reference surface provides not only more consistent but also more accurate geometry for each frame.

The modified algorithm is as follows. Lines in boldface indicate the new contributions in addition to SfMS.

Algorithm 2: Fusion-guided SFMS

```
Input: A sequence of endoscopic video frames $\{F_i | i = 1...N\}$
1. Generate a sparse 3D point cloud P and camera positions $C_{i,t}$ from the
   input frames using SfM
2. Initialize estimated surface of each frame with constant depth
3. Warp the estimated surface $S_(i,t)^w$ using its corresponding SFM 3D points
   $P_i$
4. Fuse the warped surfaces into a fused reference surface $S_t^f$
5. For each frame $F_i$
6.      Extract a reference surface $S_{i,t}^e$ from the fused reference
        surface $S_t^f$
7.      Warp the reference surface $S_(i,t)^e$ using its corresponding SfM 3D
        points $P_i$
8.      Remove saturated and under-illuminated pixels $F_i^{'}$
9.      Estimate the reflectance model BRDF using the extracted reference
        surface $S_(i,t)^e$ and the preprocessed image $F_i^{'}$
10.     Perform SFS to generate a better estimate surface $S_(i,t+1)^w$
11. Repeat steps 3-10 until convergence
```

In this algorithm the subscript i indicates the frame index and t is the iteration index. The superscript f indicates the fused reference surface, w is the warped surface, and e is the extracted surface. A sequence of endoscopic video frames $\{F_i | i = 1...N\}$ is the only input to our system. At step 1, a sparse 3D point cloud P is generated using



$S^w_{1,t}$
$S^w_{i,t}$
$S^w_{N,t}$
Reflectance model estimation & surface reconstruction
(a) Frame-by-frame reference surface
Fusion
Extraction
S^f_t
$S^e_{1,t}$
$S^e_{i,t}$
$S^e_{N,t}$
(b) Fused reference surface

Figure 3.5: In the original pipeline, the reference surface is generated for each reconstruction separately. In fusion-guided SfMS, the fused reference surface is generated using the deformable registration shared by all of the reconstructions. Then for each reconstruction a surface visible to the respective camera pose is extracted from that fused reference surface.

a program named Colmap (Schonberger and Frahm, 2016). Colmap implements a structure-from-motion (SfM) algorithm that simultaneous estimates both camera pose and 3D scene structure from multiple frames. In our system, the point cloud P is used as a prior for reflectance model estimation and surface reconstruction.

In comparison to the original SfMS, where $S^w_{i,t}$ (step 3) is directly used for reflectance model estimation (step 9) and surface reconstruction (step 10), our fusion-guided SfMS uses a single fused reference surface S^f_t. That surface is generated by fusing all warped surfaces $\{S^w_{i,t}|i = 1...N\}$ at iteration t using Zhao's (Zhao et al., 2016) registration method (step 4). Since each endoscopic image is taken at a different time, such a fusion provides temporal regularity across all the frames. Figure 3.5 shows an example of $S_(i,t)^w$ and S^f_t. We could directly incorporate temporal regularity into the SfS equation by computing optical flow between successive fames, but that would result in an extremely complex optimization system. Separating the temporal regularity and SfS makes the overall problem more solvable and stable.

Step 6 involves viewing the fused reference surface. Given camera position $C_{i,t}$, obtained from SfM (step 1), the corresponding surface $S^e_{i,t}$ that is visible to $C_{i,t}$ is extracted from S^f_t as the initial guidance surface for the subsequent reflectance model estimation and surface reconstruction. Since SfM points are treated as ground-truth, a warping is performed in step 7 to ensure that the reference surface $S^e_{i,t}$ won't deviate too much from those points.

3.4.2 Improving reflectance model estimation by outlier removal and approximation of multiple light reflections

Many assumptions and constraints are needed for SfS to be solvable. Among those assumptions, Lambertian surface reflection is one of the most popular. As described in Section 3.2, Price et al. proposed a more flexible reflectance model for modeling the surface in endoscopic environment, which is suitable for any kind of surface property. Furthermore, the reflectance model estimation process is simplified by utilizing SfM points as prior information and by the co-location of the light and camera. Price's reflectance model estimation uses a linear regression yielding the BRDF coefficients ω given the reflectance model X and image y. This regression is sensitive to noise. In the original SfMS the whole frame is used to estimate the reflectance model. However, saturated and under-illuminated pixels do not provide much useful information on surface depth. Such pixels can easily be filtered out using a predefined threshold. Doing so prevents corruption of the reflectance model by these outliers. In addition, because large BRDF coefficients are unrealistic, we also introduce a term preferring small coefficients ω in estimation of the reflectance model, thus improving its robustness against noisy data:

$$\underset{\omega}{min} ||X\omega - y||_2^2 + \alpha||\omega||_2^2 \tag{3.17}$$

Furthermore, the use of the fused surface instead of the reference surface from each single reconstruction induces further consistency of the reflectance model across different frames.

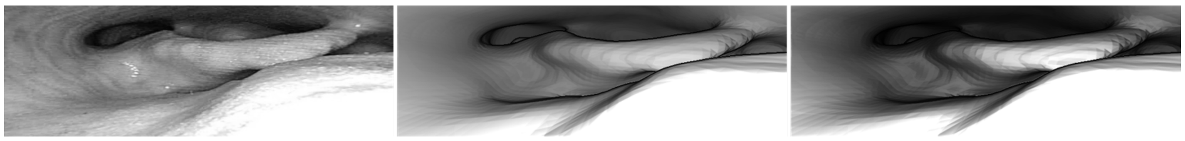


Figure 3.6: Estimated image from the refined and original reflectance models. From left to right: original image, estimation according to the refined reflectance model, and estimation according to the original reflectance model.

I noticed that the original SfMS formulation tends to underestimate surface depth for points farther away from the camera compared to the average depth of the scene. We suspect this is because the single-reflection assumption inherent in the original SfMS does not hold in endoscopic video. Points farther away from the camera are additionally illuminated by light reflecting off nearer points. Figure 3.6 shows that the original reflectance model (on the right) expects the far surface to be very dark, while it is much brighter in the actual image (on the left). We solve this problem by reducing the falloff speed of illumination in the SfS model and thus roughly approximate the multiple light bouncing effect where the overall environment is brighter. Equation 3.18 is the new reflectance model, where m controls the rate of light attenuation:

$$I_r = I_i \frac{A}{r^m} cos(\theta) BRDF(\theta) \tag{3.18}$$

I_r is the observed radiance, I_i is the light source intensity, A is related to the projected area of the light source, and θ is the angle between incident light and surface normal. Table 3.1 shows the total squared error in intensity, averaged over 12 images, using a variable falloff term versus using a fixed falloff of $m = 2$. It is apparent that intensity over the entire image is much better modeled when a variable falloff is used.

	Variable Intensity Falloff (proposed)	Fixed Intensity Falloff
Mean squared error in intensity over 12 images	3261.293	7983.519

Table 3.1: The mean squared error in intensity between the original input intensity image and a rendered version of that image using a reflectance model fit to that image with the underlying ground-truth surface. Error is averaged over 12 images of the phantom model.

3.4.3 Results

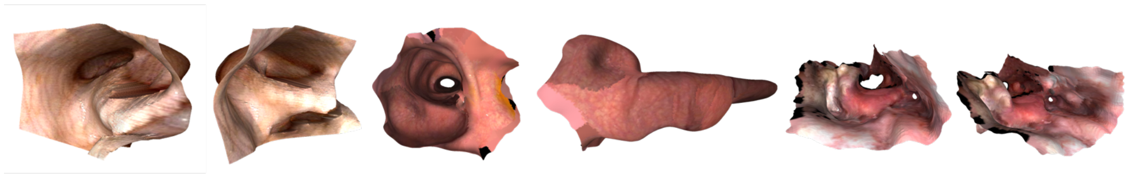


Figure 3.7: Example results from our improved SfMS method. Left pair: phantom. Middle pair: colonoscopy video. Right pair: throat.

To evaluate SfMS, Price et al. used endoscopy of a 3D-printed phantom model. A CT image of that model provided a ground-truth 3D mesh of the throat (on the left of Figure 3.7). We use the same data and evaluation scheme to show the superiority of our fusion-guided SfMS. The closest distance of SfMS estimation to the phantom surface is used to measure the reconstruction accuracy. We uniformly picked 50 frames from a sequence of 100 frames as testing data. Table 3.2 shows the percentage of average distance of each pixel to the ground-truth surface that falls within 0.5, 1.0, 1.5, 2.0, and 2.5 mm. These show improvements due to our modifications.

Methods	**Mean (Std. Dev.): Proportion of Pixels within D mm of Ground Truth**				
	D=0.5mm	D=1.0mm	D=1.5mm	D=2.0mm	D=2.5mm
SfMS	0.148 (0.066)	0.273 (0.093)	0.386 (0.100)	0.485 (0.108)	0.573 (0.115)
SfMS with improved refl. model	**0.169 (0.044)**	0.314 (0.071)	0.427 (0.100)	0.519 (0.116)	0.593 (0.123)
SfMS with fusion and improved refl. model	0.158 (0.024)	**0.319 (0.054)**	**0.453 (0.090)**	**0.560 (0.112)**	**0.637 (0.110)**

Table 3.2: Comparison results between original and improved SFMS methods using ground-truth endoscopic data.

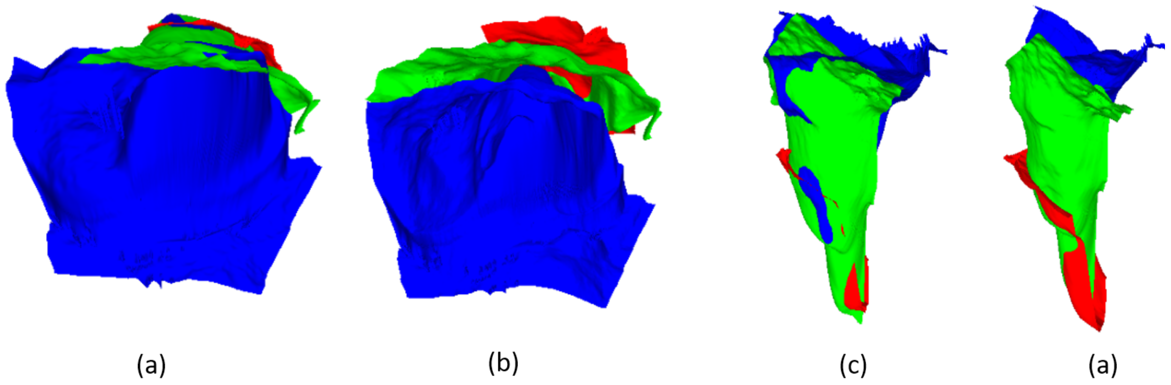


Figure 3.8: Demonstrating increased reconstruction consistency of the improved SfMS method. Single-frame reconstructions of three successive colonoscopic video frames, shown respectively in blue, green, and red are superimposed. (a) Top view of improved SfMS results. (b) Top view of original SfMS results. (c) Side view of improved SfMS results. (d) Side view of original SfMS results.

Since the phantom is rigid, the SfM algorithm already produces a fairly dense point cloud, which leads to rather consistent surface reconstructions between adjacent frames. However, in real endoscopic video, SfM produces only a sparse and sometimes inaccurate point cloud due to tissue deformation. Therefore, being without temporal regularities, the original SfMS generated reconstruction results that have larger deformations than pure tissue deformations between adjacent frames, i.e., inconsistent reconstructions. We have used real patient data to visually compare the reconstruction consistency between the original and the fusion-guided SfMS methods. Besides the pharyngeal dataset, we also applied the improved SfMS on colonoscopic video as a new application. Figure 3.8 shows the comparison result on a colonoscopic video sequence. Those three surfaces (in red, green, and blue) are reconstructed from three adjacent frames. As we can see, fusion-guided SfMS (a and c) produces a more consistent reconstruction (surfaces are closer to each other) than the original SfMS (b and d).

3.5 Texture fusion

The endoscopogram is generated by fusing both the geometry and texture from the multiple partial reconstructions. Here we present the method for fusion of the texture maps acquired from different views. Dramatically changing illumination (light binding with camera), reflection and surface deformation in endoscopic video make this problem non-trivial. The illumination changes in endoscopic images are huge even for subtle camera motions. Therefore, we need to derive a texture map from the various frames but avoid the dramatic color differences caused by the challenges we just mentioned.

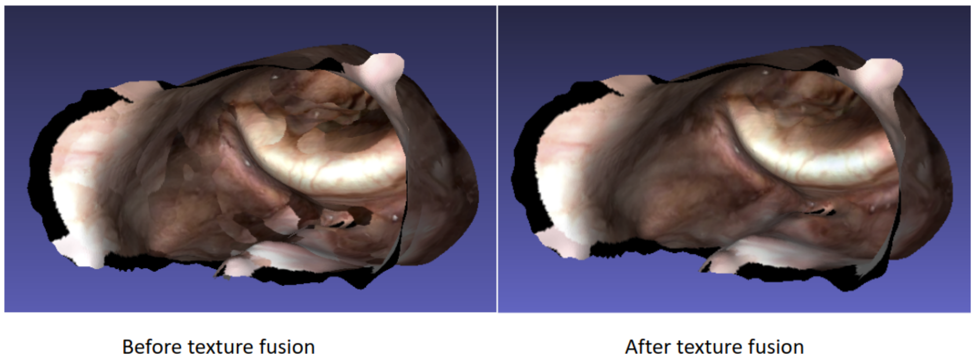


Figure 3.9: Example of our seamless texture fusion. Left: Initial pixel selection result. Right: Seamless texture fusion result.

Our approach has two stages. In the first stage an initial texture is created: to color each voxel on the endoscopogram surface we select the image whose reconstruction has the closest distance to that voxel. As detailed in the following, a Markov Random Field (MRF) based regularization is used to make the pixel selection more spatially consistent, resulting in a texture map that has multiple patches separated by clear seams, as shown in Figure 3.9.

Then in the second stage, to generate a seamless texture, we minimize within-patch intensity gradient magnitude differences and inter-patch-boundary color differences. This is also detailed in the following.

Initial pixel selection and seam placement

In the fusion process used to form the endoscopogram each frame has been registered onto it. At each endoscopogram vertex $S(i)$ one of these registered frame-based surfaces $S_k^{'}$ is closest. To begin the initialization, the color from this frame is transferred to form the initial texture map for the endoscopogram. However, the irregularity of such selection results in extreme patchiness. Thus, we add a regularity energy term that depends on the labels in the local neighborhood. Then for each pixel on the endoscopogram the scheme selects the frame index k providing the color as follows:

$$\begin{aligned} D_k(i) &= \min_{j \in S_k^{'}} d(S(i), S_k^{'}(j)) \\ M(k) &= \arg\min_{k \in L} \sum_{i \in S} (D_k(i) + \lambda N_{k,i}) \end{aligned}$$

where $D_k(i)$ is the minimum distance from the surface $S_k^{'}$ to the i^{th} point on the surface S, where $N_{k,i}$ is the number of voxels in the neighboring voxel $S(i)$ that have different labels from the label k, where $k \in 1...N$ indicates the frame indices and where M is the initial fused texture map. Such a setup is often called a Markov Random Field.

Texture fusion by minimizing within-patch and inter-patch differences

In this subsection I explain how the texture map M resulting from step 1 is modified through an iterative optimization to produce a seamless texture.

Let F be the set of images used to create the fused texture map. Let I_k be a single image in F. Let ω_k be all the pixels in image k that are selected to color M. We create a list ϕ that is composed of pairs of adjacent pixels in M that come from a different lighting condition, i.e., are members of different sets ω_k.

The fused texture should have low within-patch intensity gradient magnitude difference. The intuition is that the fused image should have the same details as the original images. The fused texture should also have low inter-patch-boundary color differences. Thus we wish to minimize

$$L_A = f + \lambda g + \mu ||g||^2 \tag{3.19}$$

where f sums the within-ω_k intensity gradient magnitudes squared (across each RGB channel) and g sums the color difference magnitudes squared of pixel pairs in ϕ. That is,

$$f = \sum_{k \in F} \sum_{i \in \omega_k} || \bigtriangledown M(C(I_k(i))) - \bigtriangledown I_k(i)||_2^2 \tag{3.20}$$

where $I_k(i)$ is the i^{th} pixel in frame k that used to form texture map M. $C(I_k(i))$ is the coordinate in M corresponding to pixel $I_k(i)$; and

$$g = \sum_{(i,j) \in \phi} ||M(i) - M(j)||_2^2 \tag{3.21}$$

I use an augmented Lagrangian method to iteratively solve the optimization problem in equation 3.19.

3.6 Simulation-based evaluation

There is no groundtruth 3D geometry available for endoscopic videos, making it difficult to quantitatively evaluate our SfMS method. Therefore, in this section I introduce an endoscopic video simulator that can produce a synthesized video with known geometry and that is realistic enough, in terms of texture, geometry, surface deformation and lighting, to be used as groundtruth data for quantitative evaluation.

3.6.1 Endoscopic Video Simulator

In order to create a realistically synthesised endoscopic video, I take four aspects into consideration: geometry, texture, deformation and lighting. For realistic geometry I directly use the CT extracted surface as the base model for the synthesized video. In order to obtain realistic texture, I register the reconstructed endoscopogram to the CT extracted surface, and then I transfer the texture from the endoscopogram to the CT extracted surface. Since both the CT and the texture on endoscopogram are from a real patient, the generated textured CT model has both realistic texture and geometry. I deform the textured CT model back into the endoscopic space to more faithfully represent the geometry occurring during endoscopy. An example of the textured CT model and the deformed textured CT model are shown in Figure 3.10.

In a real endoscopic video, the pharyngeal region is constantly deforming; thus, I also need to simulate such deformations to maintain the realism of our synthesized video. To do so, I use software named Blender (Community, 2018). The pharyngeal region can be roughly divided into four regions: epiglottis, glottis (vocal cord), base of

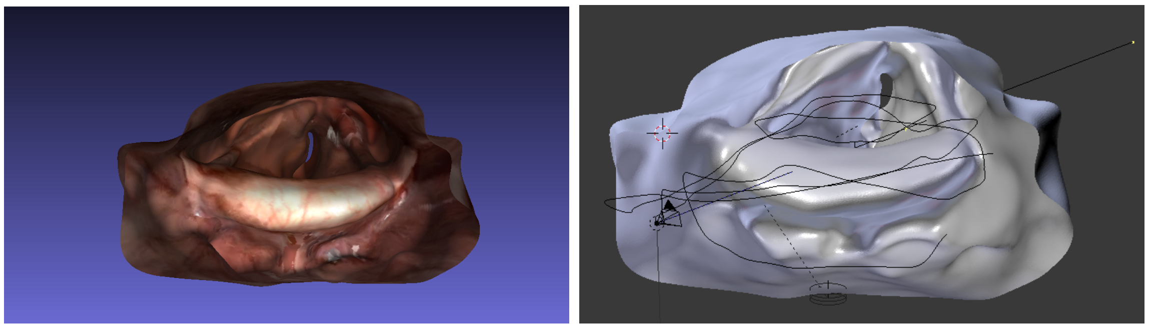
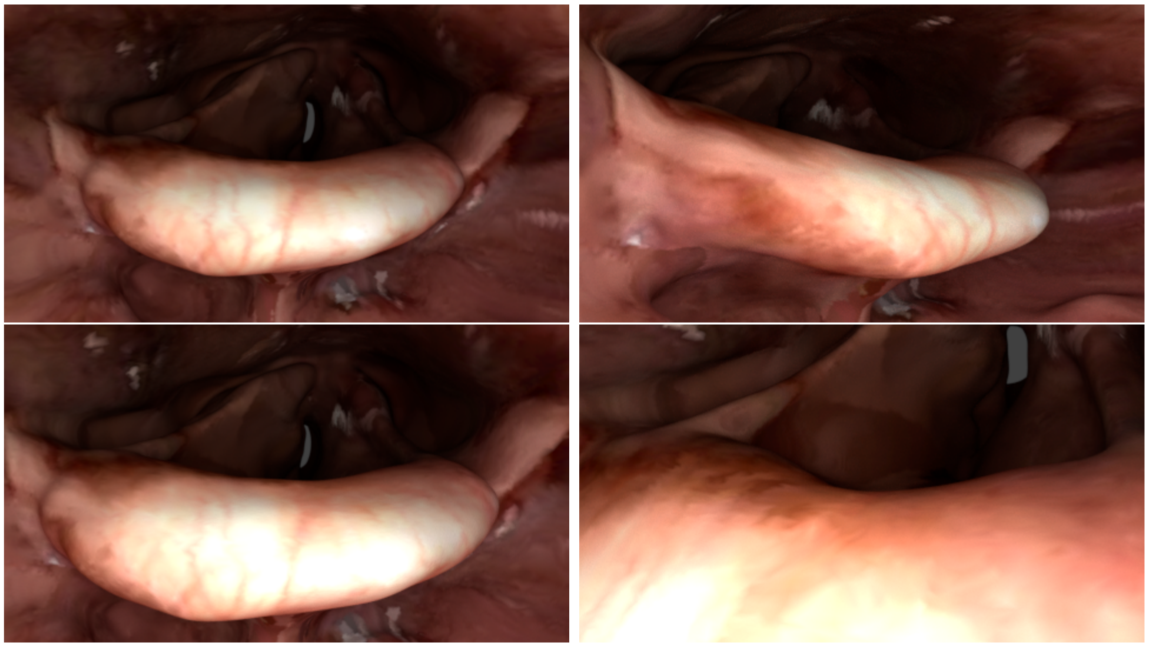


Figure 3.10: Example of deformed textured CT (left) and manually created camera path (right).

tongue, and all surrounding tissues. The vocal cord can also be further divided into cartilage and muscles. I assign different elasticity properties to these five regions to mimic the tissue properties in pharyngeal region. Forces are applied using Blender on different surfaces to produce realistic deformations. For the lighting, I use a point light source with quadratic decay to mimic the lighting from the endoscopic instrument.



(a) Input (b) Consecutive reproj.

Figure 3.11: Example images from the synthetic endoscopic video.

The Synthetic Endoscopic video is created by a virtual camera following a manually created path. Figure 3.10 shows an example of the virtual camera and the camera path. Figure 3.11 shows example images of the synthetic endoscopic video. For each rendered individual frame I have the groundtruth depth map, which is used to evaluate our frame-by-frame surface reconstruction method. I also have the complete geometry, the textured CT extracted surface, which is used to evaluate the TSD registration as well as the whole SfMS pipeline.

3.6.2 Frame-by-frame reconstruction evaluation

As discussed in Section 1.4.1, given an endoscopic video, our SfMS method first performs a frame-by-frame reconstruction; then the partial surfaces are registered using group-wise TSD to produce a fused surface. In this section I evaluate the accuracy of our frame-by-frame resconstruction using the synthetic endoscopic video. I compare each SfMS-reconstructed depth map to the corresponding groundtruth depth map. Table 3.3 shows the quantitative evaluation results. I synthesised surface deformation at different levels. It can be seen that with the increasing amounts of deformation, our method outperforms the state-of-the-art multi-view-stereo (MVS) methods. The MVS method relies on projecting 3D points across multiple images. Thus, as the deformation increases, a 3D point can only correspond to a very limited number of frames, resulting in larger errors in depth estimation. Our fusion-guided SfMS uses a fused reference surface to introduce temporal consistency across multiple frames. Moreover, it is a frame-by-frame reconstruction method that does not heavily rely on temporal consistency across multiple frames. Therefore, with the increasing amount of deformation, the performance of our SfMS is not affected as much as MVS, thereby demonstrating the superiority of our method for 3D reconstruction from endoscopic videos.

		Mean (Std. Dev.): Proportion of Pixels within X mm of GT			
Deformation	Methods	4mm	6mm	8mm	10mm
Small	MVS	0.500 (0.219)	0.789 (0.208)	0.882 (0.166)	0.945 (0.098)
	Our SfMS	0.669 (0.168)	0.910 (0.109)	0.975 (0.060)	0.995 (0.019)
Large	MVS	0.139 (0.155)	0.318 (0.285)	0.428 (0.356)	0.501 (0.382)
	Our SfMS	0.153 (0.130)	0.456 (0.178)	0.657 (0.150)	0.820 (0.105)

Table 3.3: Accuracy of our approach on a synthetic endoscopic dataset.

3.7 Clinical study software

After a complete endoscopogram is generated using our fusion-guided SfMS method, we can now register it to CT to achieve the fusion between endoscopic video and CT. To

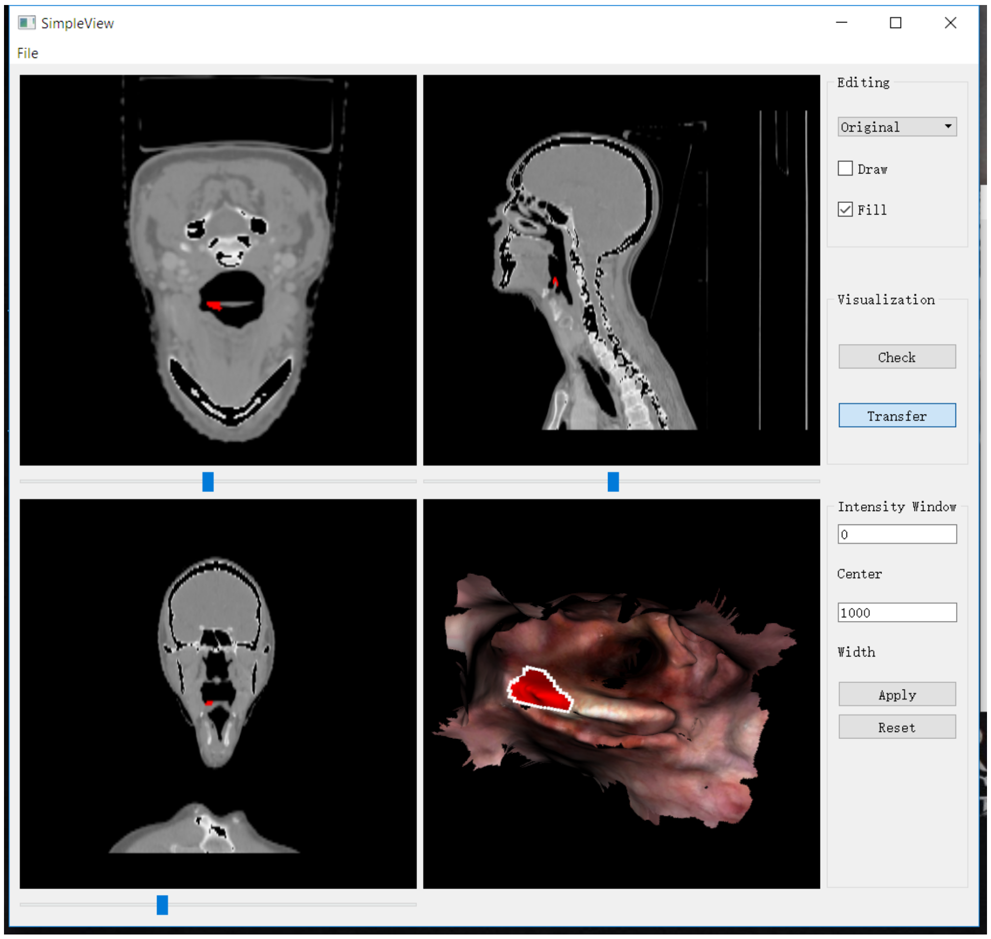


Figure 3.12: Example of an ROI being drawn on the endoscopogram surface and transferred to CT image. The user-drawn ROI is shown as a red region surrounded by a white contour in the lower right window.

allow a good initialization of the registration, we first extract the tissue-gas surface from the CT and then do a surface-to-surface registration between the endoscopogram and the surface derived from the CT. The surface extraction from CT is done using software named 3D Slicer (Pieper et al., 2004). The registration is done via Thin-Shell-Demons.

We have created a tool, named Endo2CT for the physicians to directly draw on the endoscopogram surface. Having the endoscopogram surface being registered to the CT extracted surface, the highlighted region can then be displayed on the CT image as well as each individual endoscopic frames. Figure 3.12 shows a screen shot of the UI of the Endo2CT tool and an example of an ROI being drawn on the endoscopogram surface and transferred to CT image.

The UI of Endo2CT is created using the Qt C++ framework. The visualizations are developed using VTK and ITK libraries. The basic functions of Endo2CT tool are

1. *Display of the CT images in axial, coronal and sagittal planes.*

2. *Visualization of the endoscopogram in 3D.* For visualization, the user can alternate between the original endoscopogram and the endoscopogram that is already registered into the CT space.

3. *Automatic synchronization between the endoscopogram and all the CT views.* In either the CT views or the endoscopogram view, the user can select a point and hit the 's' key on the keyboard to cause all the CT images to be switched to the slices that include that point.

4. *Interactive drawing on the endoscopogram surface or endoscopic video frames for tumor localization.* The contour that is being drawn on the video frame will be automatically mapped onto the endoscopogram surface and vice versa. In addition, the contour will be tracked across all the video frames so that the user does not need to redraw for each frame.

5. *Tumor transfer to CT.* The tumor location that being marked on the endoscopogram surface or the endoscopic video frames can be automatically extended in depth by a few millimeters and transferred onto the CT slices via the registration between the CT extracted surface and the endoscopogram.

6. *Saving the marked tumor.* Endo2CT can export the user marked tumor location into a file that can be reloaded by Endo2CT tool or loaded by any other software that supports the file format.

3.8 Preliminary clinical study

In Section 3.2 I showed the quantitative evaluation results of our fusion-guided SfMS method using a synthetic dataset. In this section I further demonstrate the usefulness of our method through a preliminary clinical study in radiation treatment planning for treatment of head and neck cancer. To our knowledge, this is the first attempt to fully integrate endoscopy video into the radiation treatment process.

In this study we analyzed 12 retrospective cases of patients with oropharyngeal or laryngeal tumors from whom endoscopic videos and planning CT scans were available for this study. The key steps in this clinical study are

1. A portion of the video is reconstructed into an endoscopogram using fusion-guided SfMS.

2. The endoscopogram is then registered with a surface-reconstructed CT scan using TSD.

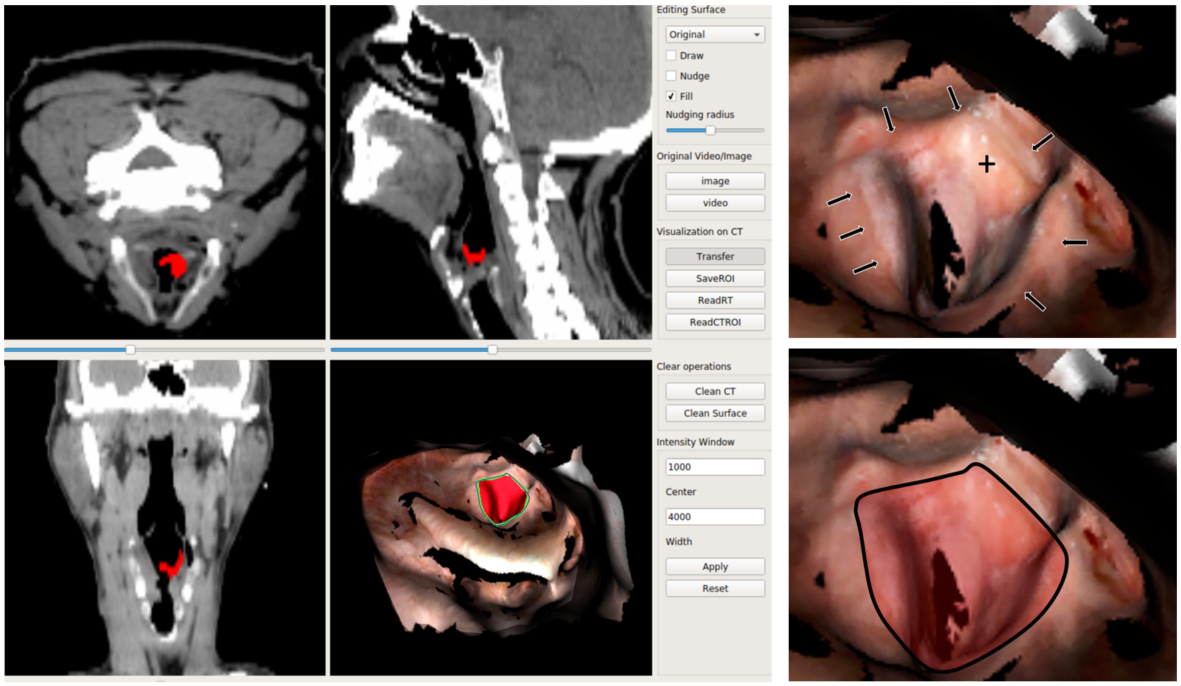
3. The clinician draws a tumor region boundary on one of the keyframes that is used for reconstructing the endoscopogram or directly on the endoscopogram via the Endo2CT software.

4. The registration transform is applied to the drawn region, and that is extended in depth by a few mm. The resulting 3D region is superimposed on the CT so as to fuse the endoscopic tumor region into the CT. This step allows us to complete our goal of bringing endoscopic video-generated data into the treatment planning process.

5. The transformed tumor region is displayed on the CT slices.

In this experiment two expert in head and neck radiation oncologists and I examined the reasonability of the registration and the relation of the transformed tumor region to the tumor region seen on the original CT. Figure 3.13 shows an example of the gross tumor volume outlined on the endoscopogram. Through the registration the tumor appeared on the patient's CT scan.

Of the 11 cases in which the tumor appears in either the CT or the endoscopic video, in 4 (2, 6, 8, 9) the tumor from the endoscopic frames corroborated what was seen on the CT but added little or no further information on the tumor location or extent. In 5 of the 11 cases (4, 10, 13, 15, 23) the tumor from the endoscopic frames provided clinically meaningful additional information either in terms of confirming the location of a poorly seen tumor or in suggesting that the tumor extent was greater than was appreciated on CT alone. In case 16 the tumor was not discernible on the CT at all, so the endoscopic information provided its location. In case 22 the registration quality appears to be inadequate. Table 1 provides details. With our limited study, our results suggest that there could be notable clinical benefit of fusing endoscopic video with CT for radiation treatment planning.



✓ Fill

Intensity Window: 1000

Center: 4000

Figure 3.13: The left and middle panels in the top row and the leftmost panel on the bottom row show the three cardinal views of the CT of a patient with a tumor of the left false cord. The user indicated tumor region, transferred to the CT, is shown in red. The reconstructed endoscopogram of the video is shown in the middle of the seccond row. Top right is a close-up view of the tumor region on the endoscopogram. Lower right is the outlined tumor region on the endoscopogram by a physician. The tumor is not well seen on CT but is clearly visible on endoscopy.

Cases	Tumor site	Registration quality*	Information added from video*
case002	Laryngeal surface of the epiglottis	Translated 2-3 mm.	Shows only tumor already seen on CT
case004	Laryngeal surface of the epiglottis	Registration is correct.	Shows unsuspected tumor extension
case005			Failure: Tumor not convincingly seen on either modality
case006	Base tongue	Registration is correct	Tumor placed properly but shows only a small portion of what is seen on CT
case008	Right base of tongue	Registration correct	The tumor is deep to the surface so red patch can only confirm general location
case009	Base of tongue	Registration correct	Red patch only confirms tumor location as already seen on CT
case010	Larynx	Registration correct	Adds greatly to knowledge of tumor location and extent (demonstration case)
case013	Left base of tongue	Small registration error	Helps confirm a questionable tumor location.
case015	Epiglottis	Registration is correct	Confirms location of poorly seen tumor
case016	Epiglottis	Registration is correct	Tumor not seen on CT at all
case022		Registration failure	
case023	Larynx, aryepiglottic fold	Registration correct	Shows tumor more extensive than on CT

Table 3.4: Clinical study results on 12 cases. The results are from the subjective judgements of the two clinicians.

Chapter 4

Background for DNN-based 3D Reconstruction

Traditionally, 3D reconstruction and localization are mostly solved by pure geometric reasoning. SfM and SLAM are the two most prevalent frameworks for sparse 3D reconstruction of rigid geometry from images. SfM is typically used for offline 3D reconstruction from unordered image collections, while visual SLAM aims for a real-time solution using a single camera (Davison et al., 2007; Newcombe et al., 2011). More recent developments of SLAM systems include ORB-SLAM (Mur-Artal et al., 2015) and DSO (Engel et al., 2018a). In (Schönberger and Frahm, 2016), Schönberger and Frahm review the state-of-the-art in SfM and propose an improved incremental SfM method.

Recently, DNNs are increasingly being applied to 3D reconstruction, in particular, to the problem of 3D reconstruction of dense monocular depth, which is similar to the segmentation problem so the structure of the DNNs can be easily adapted to the task of depth estimation (Long et al., 2015). In this chapter I give a review of the DNN-based 3D reconstruction methods.

4.1 CNN-based single view depth estimation

Convolutional neural networks were first developed for image classification tasks. Later on, with the invention of the fully convolutional neural network (Long et al., 2015) and the U-net (Ronneberger et al., 2015), dense prediction tasks such as pixel-level semantic segmentation became doable for CNNs. Dense depth mapping, with each pixel representing the distance of a point to the camera, is a very suitable and widely applied dense prediction task for CNNs. Therefore, CNN-based dense depth map prediction quickly gained a lot of interest among computer vision researchers. The most common architecture for dense depth prediction is an encoder-decoder network architecture, as shown in Figure 4.1. The encoder contains multiple convolutional layers, similar

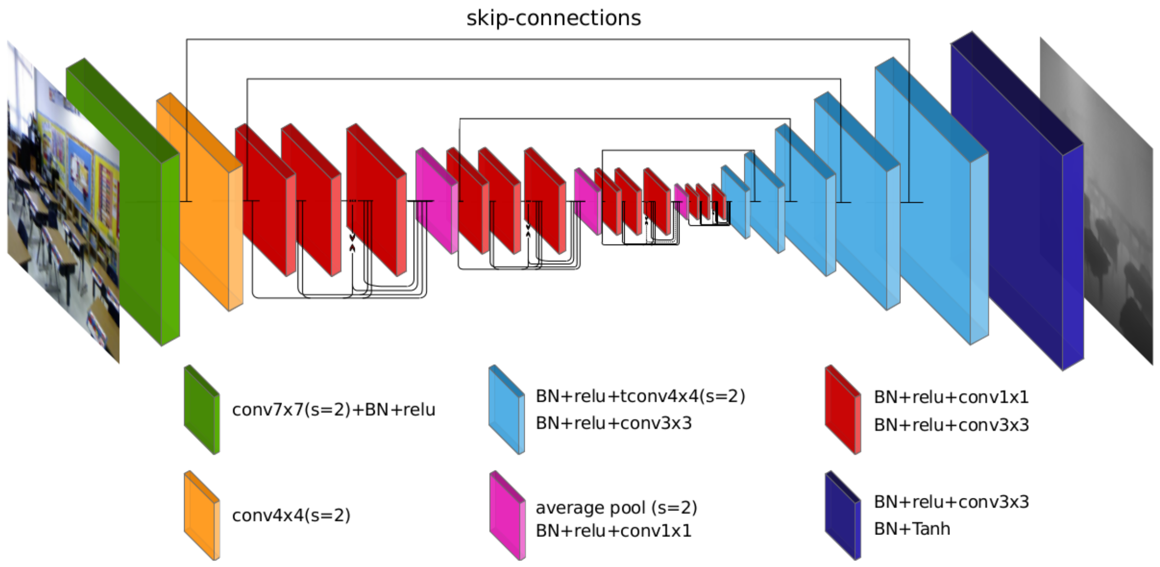


Figure 4.1: A common encoder-decoder network architecture for single image depth estimation.

to the simple CNN shown in Figure 2.7, that successively performs convolution and pooling operations to extract deep features from the input image. The decoder takes those deep features as input and successively performs deconvolution and upsampling to bring the feature map back to the desired spatial dimension. Similar to the supervised classification tasks, if the groundtruth labels (depth maps) are available, cost functions can be created to measure the differences between the prediction and the groundtruth. Then the network can be trained by minimizing the combination of the cost functions using backprogagation, as introduced in Chapter 2, Section 2.2.3. The commonly used cost function for dense depth map prediction is the L_1 loss on the inverse depth or the log depth ξ:

$$L_{depth} = \sum_{\Omega} |\xi - \hat{\xi}| \tag{4.1}$$

The inverse or log depth allows representation of points at infinity and accounts for the growing localization uncertainty of points with increasing distance. Another commonly used loss function is the scale-invariant loss proposed by Eigen et al. ():

$$L_{depth} = \frac{1}{N}\sum_{\Omega} d^2 - \frac{1}{N^2}(\sum_{\Omega} d)^2 \tag{4.2}$$

where $d = log(z) - log(\hat{z})$ and z and $\hat{z}$ are the predicted and groundtruth depths respectively.

Supervised methods. Eigen *et al.* (Eigen and Fergus, 2015) and Liu *et al.* (Liu et al., 2015) proposed end-to-end networks for single-view depth estimation, thereby opening the gate for deep-learning-based supervised single-view depth estimation. Following their work, Laina *et al.* (Laina et al., 2016) proposed a deeper residual network for the same task. Qi *et al.* (Qi et al., 2018) jointly predicted depth and surface normal maps from a single image. Fu *et al.* (Fu et al., 2018a) further improved the network accuracy and convergence rate by treating the problem as ordinal regression. Li *et al.* (Li and Snavely, 2018) used modern structure-from-motion and multi-view stereo (MVS) methods together with multi-view Internet photo collections to create the large-scale MegaDepth dataset, providing improved depth estimation accuracy via bigger training dataset size.

Unsupervised methods. Recently, by incorporating elements of view synthesis (Zhou et al., 2016) and Spatial Transformer Networks (Jaderberg et al., 2015), monocular depth estimation has been trained in an unsupervised fashion. This was done by transforming the depth estimation problem into an image reprojection problem where the depth is the intermediate product that integrates into the image reconstruction loss. Formally, the image reprojection loss is defined as

$$L_{reproj} = \sum_{\Omega} |I_t(x) - \hat{I}_s(x)| \tag{4.3}$$

where $\hat{I}_s(x)$ is the projection of the source image I_s onto the target image I_t's coordinates using the depth of the target image and the relative camera pose between the two views. The reprojection of a point from the source image to the target image can be done via Equation 2.12:

$$\tilde{x}_s = K[R|T]\tilde{X}_t = K[R|T]D(x_t)K^{-1}\tilde{x}_t \tag{4.4}$$

where K is the camera intrinsic matrix, $\tilde{x}_s$ and $\tilde{x}_t$ are the homogeneous coordinates of pixels in the source and the target image respectively, $[R|T]$ is the relative pose between the two views, and $D(x_t)$ is the predicted depth at pixel x_t in the target image.

Godard *et al.*(Godard et al., 2017), and Garg *et al.*(Garg et al., 2016) used stereo pairs to train CNNs to estimate disparity maps from single views. Luo *et al.* (Luo et al., 2018) leverage both stereo and temporal constraints to generate improved depth at known scale. Zhou *et al.* (Zhou et al., 2017) further relaxed the needs of stereo images to monocular video by combining a single view depth estimation network with a multi-view odometry estimation network. Following Zhou *et al.*'s work, Mahjourian *et al.* (Mahjourian et al., 2018) further enforced consistency of the estimated 3D point clouds and ego-motion across consecutive frames. In addition to depth and ego-motion, Yin *et al.* (Yin and Shi, 2018) also jointly learned optical flow in an end-to-end manner, imposing additional geometric constraints. Casser *et al.* (Casser et al., 2019) took advantage of both structure and semantics for unsupervised depth and ego-motion estimation. They modeled the 3D motion of each individual objects separately, so more accurate results were obtained for dynamic scenes.

4.2 RNN-based depth and visual odometry estimation

Two-view or multi-view stereo methods have traditionally been the most common techniques for dense depth estimation. For the interested reader, Scharstein and Szeliski (Scharstein and Szeliski, 2002) give a comprehensive review on two-view stereo methods. However, in contrast to traditional methods, most CNN methods introduced above treat depth estimation as a single-view task and thus ignore the important temporal information in monocular or stereo videos. Recently, Ummenhofer *et al.* (Ummenhofer et al., 2017) formulated two-view stereo as a learning problem. They showed that by explicitly incorporating dense correspondences estimated from optical flow into the two-view depth estimation, the network can be forced to utilize stereo information on top of the single-view priors.

As introduced in Chapter 2, Section 2.2.5, there is a special type of DNN, called a recurrent neural network (RNN), that is designated for sequential data. RNNs were originally designed and used for language processing, but later they have also been applied to videos. Wang *et al.* (Wang et al., 2017b) proposed an end-to-end framework for visual odometry estimation using recurrent convolutional neural networks. The authors use CNNs to first extract features from the input images, and then the features are given as inputs to the RNNs to predict camera poses. Choy *et al.* (Choy et al., 2016) used an RNN to reconstruct the object in the form of a 3D occupancy grid from multiple viewpoints. Yao *et al.* (Yao et al., 2018) proposed an end-to-end deep learning framework for depth estimation from multiple views. They use differentiable homography warping to build a 3D cost volume from one reference image and several source images. Kumar *et al.* (Kumar et al., 2018) proposed an RNN architecture that can learn depth prediction from monocular videos.

4.3 CNN+SLAM for real-time 3D reconstruction

The CNN- and RNN-based depth and visual odometry estimation methods that have been introduced above can only perform forward prediction based on learned knowledge and do not optimize on a specific case. Therefore, the errors in each step of prediction accumulate quickly, making it very difficult to fuse the depth maps into a complete reconstruction.

In computer vision an essential algorithm for online depth and camera pose estimation is called simultaneous localization and mapping (SLAM). Two major components of a typical SLAM system are tracking and local mapping (Mur-Artal et al., 2015). The tracking module uses visual clues to predict camera poses for each incoming frame and to create keyframes. The local mapping module manages the keyframes and jointly optimizes their poses and visible 3D point positions (bundle adjustment). Optionally, there can be a loop closure module with global bundle adjustment. Depending on whether the error functions in these modules directly use photometric error (intensity difference) or not, SLAM methods can be divided into "direct" methods (Engel et al., 2018b; Engel et al., 2014) and "indirect" methods such as ORB-SLAM (Mur-Artal et al., 2015; Mur-Artal and Tardós, 2017). ORB-SLAM is based on ORB feature points and reprojection error.

There have been developments combining a CNN and a traditional SLAM system to utilize both the learned prior and online optimization to achieve better 3D reconstruction. CNN-SLAM (Tateno et al., 2017) incorporates CNN-predicted depth maps into the LSD-SLAM framework. Depth maps provide a denser and more accurate uncertainty estimation. DVSO (Yang et al., 2018b) proposed replacing the stereo measurements in Stereo DSO (Wang et al., 2017a) by depth values predicted by a CNN. The effectiveness of using a CNN comes from a robust depth prior and gives reasonable depth prediction to assist the SLAM system. The designers of the BA-NET (Tang and

Tan, 2018) proposed incorporating bundle adjustment into the CNN training pipeline. They introduced a differentiable feature-metric bundle adjustment layer that takes multiple feature-pyramids of a depth estimation network as input and then predicts dense depth and camera poses. Similar to "indirect" SLAM methods, the feature-metric bundle adjustment layer uses features to measure the differences of aligned pixels, but their features are learned via back-propagation instead of using hand-crafted (SIFT, ORB) or pre-trained CNN features.

Chapter 5

RNN-based Depth and Visual Odometry Estimation from Monocular Video

The SfMS method introduced in Chapter 3 can produce high quality 3D reconstructions from endoscopic videos. However, it can take up to few hours to produce one single reconstruction. This is fine for nasopharyngoscopy, where the endoscopogram reconstruction is used for tumor localization and treatment planning and thus does not require being real-time. However, in the colonoscopic cases polyps need to be found and excised during the procedure. As mentioned in Chapter 1, one of the reasons that polyps can be missed is that the colonic mucosal surface is not entirely surveyed. Therefore, the 3D reconstruction, requiring depth and odometry estimation, needs to be done in real-time to detect missing surface areas and to help guiding the colonoscopist back to the un-surveryed areas.

Earlier results on depth and odometry estimation are evaluated on indoor and outdoor scenes. I will discuss how I have adapted the techniques to colonoscopic videos in the next chapter.

In this chapter I introduce my method for real-time multi-view depth and visual odometry estimation by leveraging RNNs. I first describe my recurrent neural network architecture and then my multi-view reprojection and forward-backward flow-consistency constraints for the network training. Finally, I demonstrate the superiority of my method by comparing it to the state-of-the-art approaches on a benchmark dataset.

5.1 Introduction

The tasks of depth and odometry (also called ego-motion) estimation are longstanding tasks in computer vision, providing valuable information for a wide variety of tasks, e.g., autonomous driving, AR/VR applications, and virtual tourism.

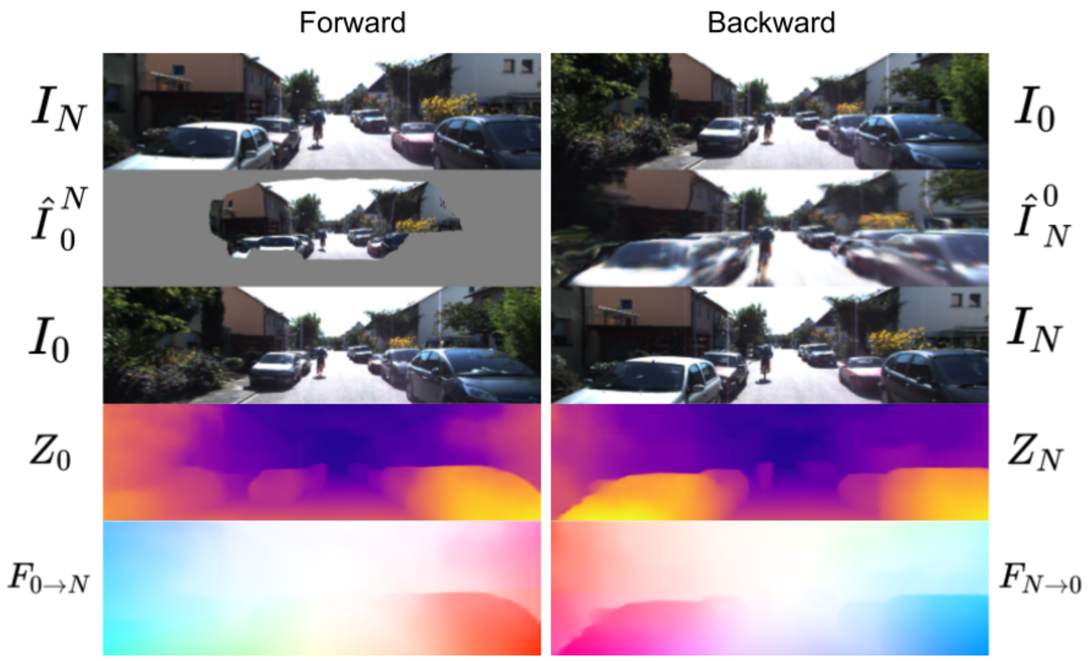


Figure 5.1: Example results from my method on the KITTI self-driving dataset. The first row shows the source image. The second row illustrates the projection of the source image into the target image. The third row shows the target image. The fourth row illustrates the estimated depth map, and the last row illustrates the estimated optical flow.

Recently convolutional neural networks (CNN) (Liu et al., 2015; Eigen and Fergus, 2015; Garg et al., 2016; Zhou et al., 2017; Ummenhofer et al., 2017) have begun to produce results of comparable quality to traditional geometric computer vision methods for depth estimation in measurable areas. They achieve significantly more complete results for ambiguous areas through the learned priors. However, in contrast to traditional methods, most CNN methods treat depth estimation as a single-view task and thus ignore the important temporal information in monocular or stereo videos. The underlying rationale of these single-view depth-estimation methods is the capability of humans to perceive depth from a single image. However, the rationale neglects the fact that motion is actually more important to the human for inferring distances (Rogers and Graham, 1979). We are constantly exposed to moving scenes, and the speed of things moving in the image is related to the combination of their relative speed and their depth.

In this work I propose a framework that simultaneously estimates the visual odometry and depth maps from a video sequence taken by a monocular camera. To be more specific, I use convolutional Long Short-Term Memory (ConvLSTM) (Xingjian et al., 2015) units to carry temporal information from previous views into the current frame's depth and visual odometry estimation. I have improved upon existing deep one- and two-view stereo depth estimation methods by interleaving ConvLSTM units between the convolutional layers to effectively utilize multiple previous frames in each estimated depth map. Since I utilize multiple views, the image reprojection constraint between multiple views can be incorporated into the loss, leading to significant improvements for both supervised and unsupervised depth and camera pose estimation.

In addition to the image reprojection constraint, I further utilize a forward-backward flow-consistency constraint (Yin and Shi, 2018). Such a constraint provides additional supervision to image areas where the image reprojection is ambiguous. Moreover, it improves the robustness and generalizability of the model. Together, these two constraints can even allow satisfactory models to be produced when groundtruth is unavailable at training time. Figure 5.1 shows an example of forward-backward image reprojection and optical flow as well as the resulting predicted depth maps.

I summarize my innovations as follows: 1) An RNN architecture for monocular depth and odometry estimation that uses multiple consecutive views. It does so by incorporating LSTM units into depth and visual odometry estimation networks.
2) A multi-view image reprojection constraint and a forward-backward flow-consistency constraint that allow the depth and camera motion estimation to benefit from the richer constraints of a multi-view process. 3) An ability for the model to be trained in both supervised and unsupervised fashion; and an ability for it to be continuously run on arbitrary length sequences delivering a consistent scene scale.

I demonstrate on the KITTI (Geiger et al., 2012) benchmark dataset that my method can produce superior results over the state-of-the-art for both supervised and unsupervised training.

5.2 Network architecture

My architecture, shown in Figure 5.2, is made up of two networks, one for depth and one for visual odometry.

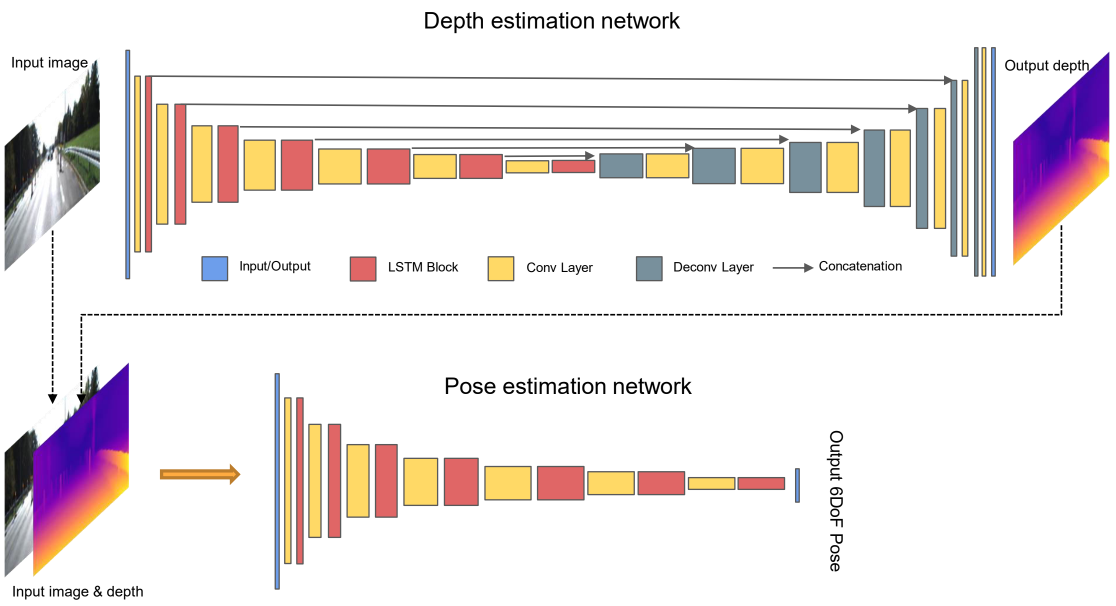


Figure 5.2: Overall network architecture of my RNN-based depth and visual odometry estimation framework. The height of each rectangle represents the size of its feature maps, where each smaller feature map is half the size of the preceding feature map.

The depth estimation network uses a U-shaped network architecture similar to DispNet (Mayer et al., 2016). My main innovation is to interleave recurrent units into the encoder, allowing the network to leverage not only spatial but also temporal information in the depth estimation. The spatial-temporal features computed by the encoder are then fed into the decoder for accurate depth map reconstruction. The ablation study in Secion 5.4.5 confirms my choice for the placements of the ConvLSTM (Xingjian et al., 2015) units. Table 5.1 details the network architecture. The input to the depth estimation network is a single RGB frame I_t, and the hidden states h^d_{t-1} from the previous time-step (h^d_{t-1} are initialized to be all zero for the first time-step). The hidden states are transmitted internally through the ConvLSTM units. The output of my depth estimation network are the depth map Z_t and the hidden states h^d_t for the current time-step .

The visual odometry network uses a VGG16 (Simonyan and Zisserman, 2014) architecture with recurrent units interleaved. Table 5.2 details the network architecture. The input to my visual odometry network is the concatenation of I_t and Z_t together with the hidden states h^p_{t-1} from the previous time-step. The output is the relative 6DoF camera pose $P_{t\to t-1}$ between the current view and the immediately preceeding view. The main differences between my visual odometry network and most current deep learning-based visual odometry methods are the following: 1) At each time-step, instead of a stack of frames, my visual odometry network only takes the current image as input; the knowledge about previous frames is in the hidden layers. 2) My visual odometry network also takes the current depth estimation as input, as a result ensuring a consistent scene scale between depth and camera pose (important for unsupervised depth estimation, where the scale is ambiguous). 3) My visual odometry network can run on a full video sequence while maintaining a single scene scale.

Type	Filters	Output size
Input		128 ×416×3
Conv+ConvLSTM	32@3×3×3	64×208×32
Conv+ConvLSTM	64@3×3×32	32×104×64
Conv+ConvLSTM	128@3×3×64	16×52×128
Conv+ConvLSTM	256@3×3×128	8×26×256
Conv+ConvLSTM	256@3×3×256	4×13×256
Conv+ConvLSTM	256@3×3×256	2×7×256
Conv+ConvLSTM	512@3×3×256	1×4×512
Deconv+Concat+Conv	256@3×3×512	2×7×256
Deconv+Concat+Conv	128@3×3×256	4×13×128
Deconv+Concat+Conv	128@3×3×128	8×26×128
Deconv+Concat+Conv	128@3×3×128	16×52×128
Deconv+Concat+Conv	64@3×3×128	32×104×64
Deconv+Concat+Conv	32@3×3×64	64×208×32
Deconv	16@3×3×32	128×416×16
Conv (output)	1@3×3×16	128×416×1

Table 5.1: Details of the depth estimation network architecture. Every convolution in the encoder uses stride 2 for downsampling. Before the output a sigmoid activation function is used to ensure the output is in range [0, 1]; All the other convolutions and decovolutions are followed by batch norm computation and LeakyRELU activation.

Type	Filters	Output size
Input		128×416×4
Conv+ConvLSTM	32@3×3×3	64×208×32
Conv+ConvLSTM	64@3×3×32	32×104×64
Conv+ConvLSTM	128@3×3×64	16×52×128
Conv+ConvLSTM	256@3×3×128	8×26×256
Conv+ConvLSTM	256@3×3×256	4×13×256
Conv+ConvLSTM	256@3×3×256	2×7×256
Conv+ConvLSTM	512@3×3×256	1×4×512
Conv (output)	6@1×1× 512	1× 1×6

Table 5.2: Details of the visual odometry network architecture. Every convolution (except for output layer) is followed by batch normalization and RELU activation.

5.3 Losses and constraints

Multi-view Reprojection Loss

Zhou *et al.* (Zhou et al., 2017) showed that the learning of depth and visual odometry estimation can be formulated as an image reconstruction problem using a differentiable geometric module (DGM). Thus, I can use a DGM to formulate an image reconstruction constraint between I_t and I_{t-1} using the estimated depth Z_t and camera pose $P_{t\to t-1}$ as introduced in the previous subsection. However, such a pairwise photometric consistency constraint is very noisy due to illumination variation, low texture, occlusion, etc. Recently, Iyer *et al.* (Iyer et al., 2018) proposed a composite transformation constraint for self-supervised visual odometry learning. By combining the pairwise image reconstruction constraint with the composite transformation constraint, I propose a multi-view image reprojection constraint that is robust to noise and provides strong self-supervision for my multi-view depth and visual odometry learning. As shown in Figure 5.3(c), the output depth maps and relative camera poses together with the input sequence are fed into a differentiable geometric module (DGM) that performs differentiable image warping of every previous view of the subsequence into the current view.

Denote the input image sequence (shown in Figure 5.3(a)) as $\{I_t \,|\, t = 0...N-1\}$, the estimated depth maps as $\{Z_t \,|\, t = 0...N-1\}$, and the camera poses as the transformation matrices from frame t to $t-1$: $\{P_{t\to t-1} \,|\, t = 0...N-1\}$. The multi-view reprojection loss is

$$L_{fw} = \sum_{t=0}^{N-1} \sum_{i=0}^{t-1} \sum_{\Omega} \lambda_t^i \omega_t^i \, |I_t - \hat{I}_t^i| \tag{5.1}$$

where $\hat{I}_t^i$ is the i^{th} view warped into t^{th} view, Ω is the image domain, ω_t^i is a binary mask indicating whether a pixel of I_t has a counterpart in I_i, and λ_t^i is a weighting term that decays exponentially based on $t-i$. Image pairs that are far away naturally suffer from larger reprojection error due to interpolation and moving foreground, so I use λ_t^i to reduce the effect of such artifacts. ω_t^i and $\hat{I}_t^i$ use the function ϕ:

$$\omega_t^i, \hat{I}_t^i, F_{t\to i} = \phi(I_i, Z_t, P_{t\to i}, K) \tag{5.2}$$

where $F_{t\to i}$ is a dense flow field for 2D pixels from view t to view i, which is used to compute flow consistency. K is the camera intrinsic matrix. The pose change from view t to i, $P_{t\to i}$ can be obtained by a composite transformation as

$$P_{t\to i} = P_{i+1\to i} \cdot \ldots \cdot P_{t-1\to t-2} \cdot P_{t\to t-1} \tag{5.3}$$

The function ϕ in Equation 5.2 warps image I_i into I_t using Z_t and $P_{t\to i}$. That function is a DGM (Zhou et al., 2017) that performs a series of differentiable 2D-to-3D, 3D-to-2D projections and bi-linear interpolation operations (Jaderberg et al., 2015).

In the same way, I reverse the input image sequence and perform another pass of depth $\{Z_t \,|\, t = N-1...0\}$ and camera pose $\{P_{t\to t+1} \,|\, t = N-1...0\}$ estimation, obtaining the backward multi-view reprojection loss L_{bw}. This multi-view reprojection loss can fully exploit the temporal information in my ConvLSTM units from multiple previous views by explicitly putting constraints between the current view and every previous view.

A trivial solution to Equation 5.1 is for ω_t^i to be all zeros. To prevent the network from converging to the trivial solution, I add a regularization loss L_{reg} to ω_t^i, which gives a constant penalty to locations where ω_t^i is zero.

$$L_{reg} = \sum_{\Omega} |\omega_t^i - 1| \tag{5.4}$$

Forward-backward Flow Consistency Loss

A forward-backward consistency check has become a popular strategy in many learning-based tasks because it provides self-supervision and regularization. It has been used in tasks such as optical flow(Hur and Roth, 2017), registration (Zhang, 2018), and depth estimation (Yin and Shi, 2018; Godard et al., 2017; Vijayanarasimhan et al., 2017). Similar to (Yin and Shi, 2018; Vijayanarasimhan et al., 2017) I use the dense flow field as a hybrid forward-backward consistency constraint for both the estimated depth and pose.

I first introduce a forward-backward consistency constraint on a single pair of frames and then generalize to a sequence. Let us denote a pair of consecutive frames as I_A and I_B, and their estimated depth maps and relative poses as Z_A, Z_B, $P_{A\to B}$, and $P_{B\to A}$. I can obtain a dense flow field $F_{A\to B}$ from frame I_A to I_B using Equation 5.2. Similarly, I can obtain $F_{B\to A}$ using Z_B, $P_{B\to A}$. Using $F_{B\to A}$ I can compute a pseudo-inverse flow $\hat{F}_{A\to B}$ ("pseudo-" due to occlusion and interpolation) as

$$\omega_A^B, \hat{F}_{A\to B}, F_{A\to B} = \phi(-F_{B\to A}, Z_A, P_{A\to B}, K) \tag{5.5}$$

This is similar to Equation 5.2 except that I am interpolating $F_{A \to B}$ from $-F_{B \to A}$ instead of I_t from I_i. Therefore, I can formulate the flow consistency loss as

$$L_{flowconsist} = \omega_A^B \cdot |F_{A \to B} - \hat{F}_{A \to B}| + \omega_B^A \cdot |F_{B \to A} - \hat{F}_{B \to A}| \tag{5.6}$$

This is performed for every consecutive pair of frames in the input sequence. Unlike the multi-view reprojection loss, which is computed for all pairs, I only compute flow-consistency on pairs of consecutive frames because as the magnitude of the flow increases, i.e., for frame pairs that are far apart, pseudo-inverses become inaccurate due to interpolation.

Smoothness Loss

Local smoothness is a common assumption for depth estimation. Following Zhan *et al.* (Zhan et al., 2018), I use an edge-aware smoothness constraint which is defined as

$$L_{smooth} = \sum_{t=0}^{N-1} \sum_{\Omega} |\nabla \xi_t| \cdot e^{-|\nabla I_t|} \tag{5.7}$$

where ξ_t is the inverse depth. $e^{-|\nabla I_t|}$ puts small weights on edges such that non-smoothness of depth map at edges will not be penalized.

Absolute depth loss

The combination of multi-view reprojection loss L_{fw}, L_{bw} defined in Equation 5.1, forward-backward flow-consistency loss $L_{flowconsist}$ defined in Equation 5.6, and smoothness loss L_{smooth} defined in Equation 5.7 can form an unsupervised training strategy for the network. This manner of training is suitable for cases where there is no groundtruth depth available, which is true for the majority of real-world scenarios. However, the network trained in this way only produces depth at a relative scale. So optionally, if there is groundtruth depth available, even sparsely, I can train a network to estimate depth at absolute scale by adding the absolute depth loss defined as

$$L_{depth} = \sum_{t=0}^{N-1} \sum_{\Omega} |\xi_t - \hat{\xi}_t| \tag{5.8}$$

In addition, I can replace the local smoothness loss in Equation 5.7 by the similarity of the gradient of depth relative to that of the groundtruth depth; this can be defined as

$$L_{smooth} = \sum_{t=0}^{N-1} \sum_{\Omega} |\nabla \xi_t - \nabla \hat{\xi}_t| \tag{5.9}$$

5.3.1 Training pipeline

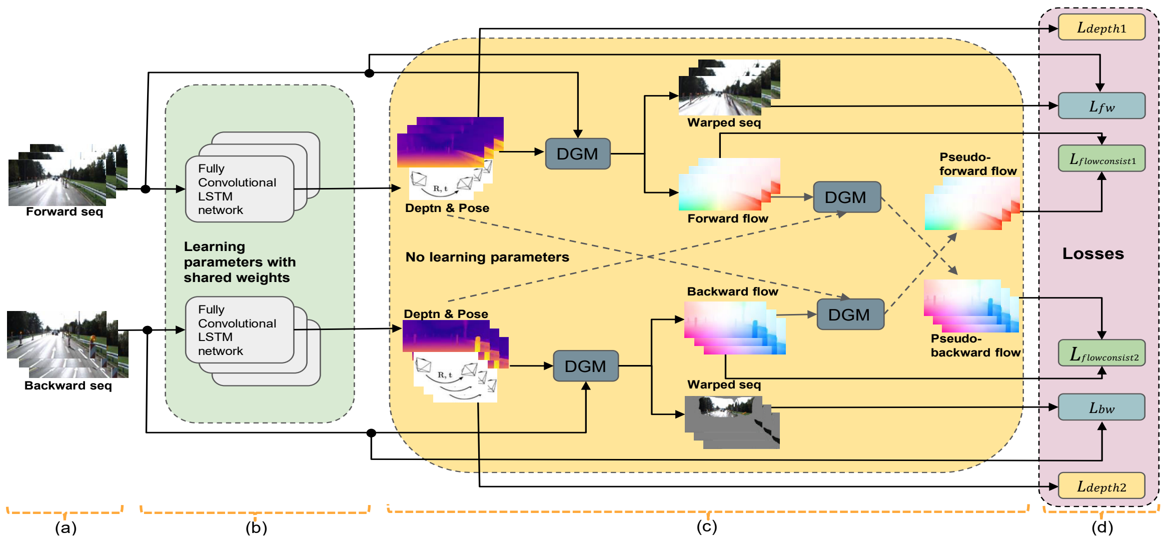


Figure 5.3: Training pipeline of the proposed RNN-based depth and visual odometry estimation network. During training, my framework takes forward and backward 10-frame subsequences as input and uses multi-view image reprojection, flow-consistency, and optionally groundtruth depth to train my depth and visual odometry networks. “DGM” indicates a differentiable geometric module.

The full training pipeline of my method is shown in Figure 5.3. Groups of N consecutive keyframes (I use $N = 10$ in all my experiments) are formed as input sequences S_{fw}. Because the image reprojection constraints are ambiguous for very small baselines, the keyframe selection is based on the motion between successive frames; I discard frames with baseline motion smaller than σ. The keyframes are grouped in a sliding window fashion such that more training data can be generated. Before passing the sequence to the network for training, I also reverse the sequence to create a backward sequence S_{bw}, which serves as a data augmentation. More importantly, it is used to enforce the forward-backward constraints.

The input sequence S_{fw} is generated offline during the data preparation stage while the backward sequence S_{bw} is generated online during the data preprocessing stage.

S_{fw} and S_{bw} are fed into the forward and backward networks with shared weights; each generates a sequence of depth maps and camera poses as shown in Figure 5.3. The estimated depth maps and camera poses are then utilized to generate dense flows to warp previous views to the current view through a differentiable geometric module (DGM) (Zhou et al., 2016; Yin and Shi, 2018). Furthermore, I utilize DGMs to generate the pseudo-inverse flows for both the forward and backward flows. By combining image warping loss, flow-consistency loss and optionally absolute depth loss, I form the full training pipeline for my proposed framework.

Once trained, the framework can run on arbitrary-length sequences without grouping frames into fixed-length subsequences. To bootstrap the depth and pose estimation, the hidden states for the ConvLSTM units are initialized at zero for the first frame. All following estimations will then depend on the hidden states from the previous time-step.

5.4 Results

In this section I show a series of experiments using the KITTI driving dataset (Geiger et al., 2013; Geiger et al., 2012) to evaluate the performance of my RNN-based depth and visual odometry estimation method. Results on colonoscopic videos are provided in the next chapter.

As mentioned in Section 5.1, my architecture can be trained in a supervised or unsupervised mode. Therefore, I evaluated both supervised and unsupervised versions of my framework. In the following experiments I named the supervised version as *ours-sup* and the unsupervised version as *ours-unsup*. I also performed detailed ablation studies to show the impact of the different constraints, architecture choices and estimations at different time-steps.

5.4.1 Implementation details

I set the weights for depth loss, smoothness loss, forward-backward consistency loss and mask regularization to 1.0, 1.0, 0.05 and 0.05, respectively. The weight for the image reprojection loss was taken to be $\frac{1}{2^{\delta}-1}$, where δ is the number of frame intervals between source and target frame. I used the Adam (Kingma and Ba, 2014) optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$. The initial learning rate was set as 0.0002. The training process is very time-consuming for my multi-view depth and odometry estimation network. One strategy I used to speed up the training process, without losing accuracy, was first to pretrain the network with the consecutive view reprojection loss for 20 epochs. Then I fine-tuned the network with the multi-view reprojection loss for another 10 epochs.

5.4.2 Training datasets

I used the KITTI driving dataset (Geiger et al., 2012) to evaluate my proposed framework. To perform a consistent comparison with existing methods, I used the division into training and evaluation cases according to the Eigen Split approach (Eigen and Fergus, 2015) to train and evaluate my depth estimation network. From the 33 training scenes, I generated 45200 10-frame sequences. The KITTI dataset uses stereo cameras that produce frames in pairs. My method does not utilize stereo information, so I used the stereo camera as two monocular cameras. I resized the images from 375×1242 to 128×416 for computational efficiency and to be comparable with existing methods. The image reprojection loss is driven by motion parallax, so I discarded all static frames with baseline motion less than $\sigma = 0.3$ meters during data preparation. 697 frames from the 28 test scenes were used for quantitative evaluation. For odometry evaluation I used the KITTI Odometry Split (Geiger et al., 2012), which contains 11 sequences with ground truth camera poses. I followed (Zhou et al., 2017; Zhan et al., 2018), which uses sequences 00-08 for training and 09-10 as evaluation.

5.4.3 Depth estimation

To evaluate the depth estimation component of my multi-view depth and odometry network, I compare to the state-of-the-art CNN-based depth estimation methods. My network takes advantage of previous images and depths through recurrent units and thus achieves best performance when running on a continuous video sequence. However, it would be unfair to compare to single-view methods when my method uses multiple views. On the other hand, if I also used only a single view for my method, I would fail to reveal the full capacity of my framework. Therefore, in order to present a more comprehensive depth evaluation, I report both my depth estimation results with and without previous views' assistance. *Ours-sup (single-view)* is the single view (or first view) depth estimation result of my framework, which also shows the bootstrapping performance of my approach. *Ours-sup (multi-view)* is the tenth view depth estimation result from my network. I evaluate the performance of depth prediction based on the following metrics: mean absolute relative error (Abs Rel), root mean squared error (RMSE) , root mean squared log error (RMSE (log)) and the accuracy under threshold

$(\delta_i < 1.25^i, i = 1, 2, 3)$.

Methods	Dataset	Supervised		Error metric				Accuracy metric		
		depth	pose	RMSE	RMSE log	Abs Rel	Sq Rel	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Zhou *et al.* (Zhou et al., 2017)	CS+K			6.709	0.270	0.183	1.595	0.734	0.902	0.959
Liu *et al.* (Liu et al., 2015)	K	✓		6.523	0.275	0.202	1.614	0.678	0.895	0.965
Eigen *et al.* (Eigen and Fergus, 2015)	K	✓		6.307	0.282	0.203	1.548	0.702	0.890	0.958
Yin *et al.*(Yin and Shi, 2018)	K			5.857	0.233	0.155	1.296	0.806	0.931	0.931
Zhan *et al.* (Zhan et al., 2018)	K		✓	5.585	0.229	0.135	1.132	0.820	0.933	0.971
Zou *et al.* (Zou et al., 2018)	K			5.507	0.223	0.150	1.124	0.793	0.933	0.973
Godard *et al.* (Godard et al., 2017)	CS+K		✓	5.311	0.219	0.124	1.076	0.847	0.942	0.973
Atapour *et al.* (Atapour-Abarghouei and Breckon, 2018)	K+S*	✓		4.726	0.194	0.110	0.929	0.923	0.967	0.984
Kuznietsov *et al.* (Kuznietsov et al., 2017)	K	✓	✓	4.621	0.189	0.113	0.741	0.875	0.964	0.988
Yang *et al.* (Yang et al., 2018a)	K	✓		4.442	0.187	0.097	0.734	0.888	0.958	0.980
Fu *et al.* (ResNet) (Fu et al., 2018b)	K	✓		2.727	<u>0</u>	**0.072**	0.307	<u>0.932</u>	<u>0.984</u>	0.994
Ours-unsup (multi-view)	K			2.320	0.153	0.112	0.418	0.882	0.974	0.992
Ours-sup (single-view)	K	✓		<u>1.949</u>	0.127	0.088	<u>0.245</u>	0.915	<u>0.984</u>	<u>0.996</u>
Ours-sup (multi-view)	K	✓		**1.698**	**0.110**	<u>0.077</u>	**0.205**	**0.941**	**0.990**	**0.998**

Table 5.3: Quantitative comparison of my network with other state-of-the-art CNN-based methods on the KITTI (Geiger et al., 2012) dataset using the division into training and evaluation cases according to Eigen Split (Eigen and Fergus, 2015). *Ours sup (single-view)* is the evaluation of single-view depth-estimation results. *Ours sup (mult-view)* is the evaluation of results generated with the assistance of nine previous views. Even though my method is not restricted to a fixed number of frames per sequence during prediction or evaluation, I still use 10-frame sequences here for consistency with the training. I discuss continuous estimation results in Section 5.4.5, studying ablation. The bold numbers are results that rank first and the underlined results those that rank second. All results are capped at 80m depth.

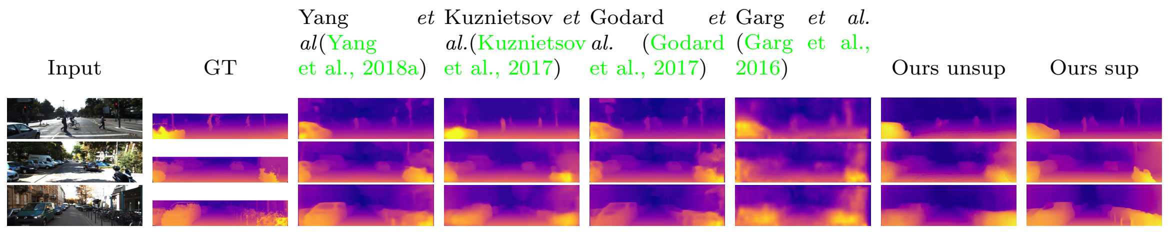

Input
GT
Yang et al(Yang et al., 2018a)
Kuznietsov et al.(Kuznietsov et al., 2017)
Godard et al. (Godard et al., 2017)
Garg et al. (Garg et al., 2016)
Ours unsup
Ours sup

Figure 5.4: Visual comparison of depth maps between the state-of-the-art methods for three randomly selected images. For visualization the groundtruth depth is interpolated. My method captures more details in thin structures, such as the motorcycle and columns in the lower right corner of figure rows 2 and 3.

As shown in Table 5.3, *ours-sup (multi-view)* performs significantly better than all of the other supervised (Liu et al., 2015; Eigen and Fergus, 2015; Atapour-Abarghouei and Breckon, 2018; Yang et al., 2018a; Fu et al., 2018b; Kuznietsov et al., 2017) and unsupervised (Zhou et al., 2017; Yin and Shi, 2018; Zou et al., 2018; Zhan et al., 2018; Godard et al., 2017) methods. The unsupervised version of my network outperforms the state-of-the-art unsupervised methods as well as several supervised methods. Both the supervised and unsupervised version of my network outperform the respective state-of-the-art by a large margin. Figure 5.4 shows a visual comparison of my method with other methods.

My method consistently captures more detailed structures, e.g., the motorcycle and columns in the lower right corner of the figures in rows 2 and 3.

5.4.4 Pose estimation

I used the KITTI Odometry Split to evaluate my visual odometry network. For pose estimation I directly ran my method through the whole sequence instead of dividing into 10-frame subsequences. I compared to the state-of-the-art learning-based visual odometry methods (Zhan et al., 2018; Zhou et al., 2017; Yin and Shi, 2018) as well as a popular monocular SLAM method: ORB-SLAM (Mur-Artal et al., 2015). I used the KITTI Odometry evaluation criterion (Geiger et al., 2012), which computes the average translation and rotation errors over subsequences of lengths 100m, 200m, ... , 800m.

Prior to the evaluation I needed to align the trajectories of both the monocular ORB-SLAM and the unsupervised learning-based visual odometry methods with groundtruth because they suffer from scale ambiguity. This alignment was done using evo[1]. The supervised version of my method (absolute depth supervision) and the stereo supervised method (Zhan et al., 2018) are able to estimate camera translations at absolute scale, so there was no alignment processing for these two methods.

Methods	Seq 09		Seq 10	
	t_{err}(%)	r_{err}(deg/m)	t_{err}(%)	r_{err}(deg/m)
ORB-SLAM (Mur-Artal et al., 2015)	15.30	0.003	3.68	0.005
GeoNet (Yin and Shi, 2018)	43.76	0.160	35.60	0.138
Zhou *et al.* (Zhou et al., 2017)	17.84	0.068	37.91	0.178
Zhan *et al.* (Zhan et al., 2018)	11.92	0.036	12.62	0.034
DeepVO *et al.* (Wang et al., 2017b)	-	-	8.11	0.088
Our unsupervised	9.88	0.034	12.24	0.052
Our supervised	9.30	0.035	7.21	0.039

Table 5.4: Quantitative comparison of visual odometry results on the KITTI Odometry dataset. t_{err} is the percentage of average translational error and r_{err} is the average degree per meter rotational error.

Table 5.4 shows the quantitative comparison results based on the KITTI Visual Odometry criterion. Figure 5.5 shows a visual comparison of the full trajectories for all the methods. Including my method, all the full trajectories of the learning-based

[1]github.com/MichaelGrupp/evo

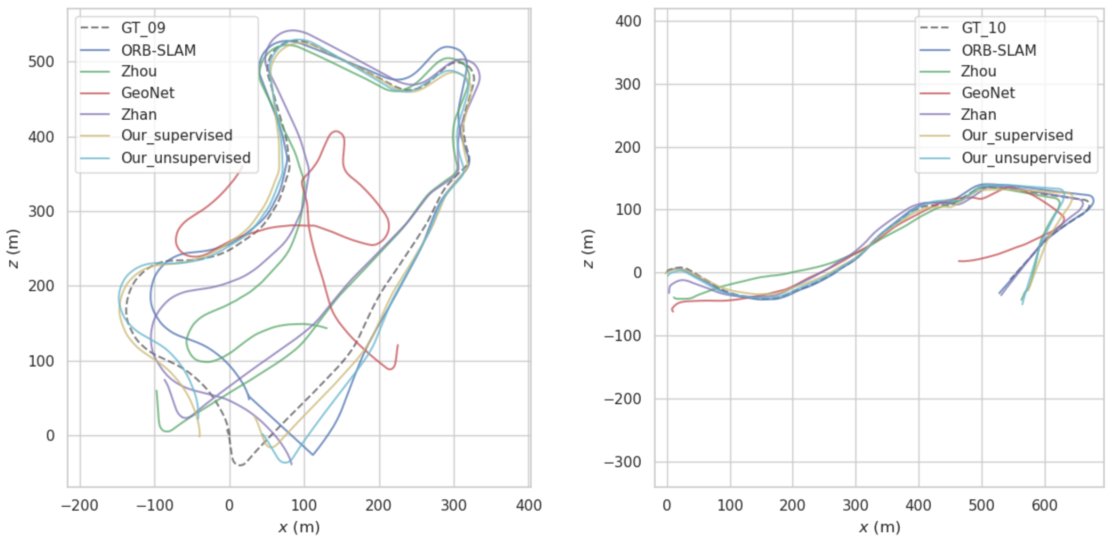


Figure 5.5: Visual comparison of full trajectories on Seq 09 (left) and 10 (right). My predictions are closest to groundtruth (GT_09 and GT_10).

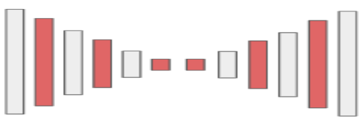
visual odometry methods were produced by integrating frame-to-frame relative camera poses over the whole sequence without any drift correction.

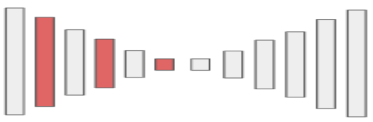
The methods (Zhou et al., 2017; Yin and Shi, 2018) take a small subsequence (5 frames) as input and estimate relative poses between frames within the subsequence. There is no temporal correlation between different subsequences, so the scales are different between those subsequences. However, my method can perform continuous camera pose estimation within a whole video sequence for arbitrary length. The temporal information is transmitted through recurrent units for arbitrary length and thus maintains a consistent scale within each full sequence.

5.4.5 Ablation study

In this section I investigate the important components in the proposed depth and visual odometry estimation network, namely placements of the recurrent units, multi-view reprojection and forward-backward consistency constraints.

Placements of recurrent units. Convolutional LSTM units are essential components for my framework to leverage temporal information in depth and visual odometry estimation. Thus, I performed a series of experiments to demonstrate the influence of these recurrent units as well as the choice for the placements of recurrent units in the network architecture. I tested three different architecture choices, which are shown


(a) Full LSTM


(b) Encoder LSTM

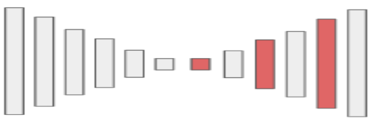

(c) Decoder LSTM

Figure 5.6: Three different architectures depend on the placements of recurrent units. (a) There is a convolutional LSTM placed after every convolution or deconvolution layer. (b) Convolutional LSTMs are placed only in the encoder. (c) Convolutional LSTMs are placed only in the decoder.

in Figure 5.6. The first one is interleaving LSTM units across the whole network (full LSTM). The second one is interleaving LSTM units across the encoder (encoder LSTM). The third one is interleaving LSTM units across the decoder (decoder LSTM). Table 5.5 shows the quantitative comparison results. It can be seen that the encoder LSTM performs significantly better than the full LSTM and the decoder LSTM. Therefore, I chose the encoder LSTM as my depth estimation network architecture.

Method	RMSE	RMSE log	Abs Rel	Sq Rel
full LSTM	1.764	0.112	0.079	0.214
decoder LSTM	1.808	0.117	0.082	0.226
encoder LSTM	**1.698**	**0.110**	**0.077**	**0.205**

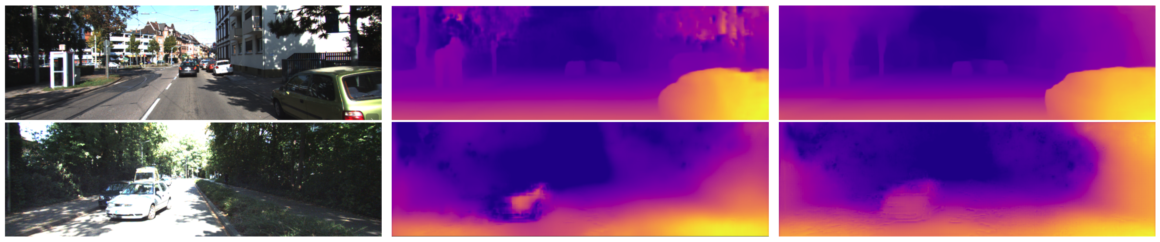
Table 5.5: Ablation study on network architectures. The evaluation data and protocol are the same as table 5.3.

Multi-view reprojection and forward-backward consistency constraints. To investigate the performance gain from the multi-view reprojection and forward-backward consistency constraints, I conducted another group of experiments. Table 5.6 shows the quantitative evaluation results. I compared among three methods: with only the consecutive image reprojection constraint (*Ours-d*), with the consecutive image reprojection constraint and the forward-backward consistency constraint (*Ours-dc*), and with the multi-view reprojection constraint and the forward-backward consistency constraint (*Ours-mc*).

As shown by the results of the last two rows in Table 5.6, the multi-view reprojection loss is more important in the unsupervised training. Figure 5.7 shows a qualitative comparison between networks trained using consecutive image reprojection loss and those trained using multi-view reprojection loss. It can be seen that multi-view reprojection loss provides better supervision to areas that lack groundtruth depth.

Method	RMSE	RMSE log	Abs Rel	Sq Rel
Ours-d	1.785	0.116	0.081	0.214
Ours-dc	1.759	0.113	0.079	0.215
Ours-mc	1.698	0.110	0.077	0.205
Ours-dc unsup	2.689	0.184	0.138	0.474
Ours-mc unsup	2.361	0.157	0.112	0.416

Table 5.6: Ablation study on multi-view reprojection and forward-backward flow consistency constraints. *d* stands for consecutive image reprojection. *m* stands for multi-view image reprojection. *c* stands for forward-backward flow consistency constraint. The first three rows are comparison among forms of supervised training, and the last two rows are between forms of unsupervised training.



(a) Input (b) Consecutive reproj. (c) Muti-view reproj.

Figure 5.7: On two input frames, visual examples between networks trained using consecutive image reprojection loss and those using multi-view reprojection loss. Results in the first row were produced using *ours-sup*, and results in the second row were produced using *ours-unsup*.

Estimation with different temporal-window sizes. Table 5.7 shows a comparison among depth estimations with different temporal-window sizes, i.e., the number of frames forming the temporal summary.

Window size	RMSE	RMSE log	Abs Rel	Sq Rel
1	1.949	0.127	0.088	0.245
3	1.707	0.110	0.077	0.206
5	1.699	0.110	0.077	0.205
10	1.698	0.110	0.077	0.205
20	1.711	0.117	0.077	0.208
Whole seq.	1.748	0.119	0.079	0.214

Table 5.7: Depth estimations with different time-window sizes.

Here I use the Eigen Split 697 testing frames for these sliding-window-based evaluations. In addition, I also ran through each whole testing sequence and again performed evaluation on those 697 testing frames. The result demonstrates that 1) the performance of the depth estimation increases with the number of depth estimations performed before the current estimation; 2) the performance of the depth estimation does not increase after 10 frames; 3) even though my network is trained on 10-frame based subsequences, it can succeed on arbitrary length sequences.

5.5 Conclusion

In this chapter I presented an RNN-based, multi-view method for depth and camera pose estimation (RNN-DP) from outdoor monocular video sequences. I demonstrated that my method can be trained either supervised or unsupervised and that both produce superior results compared to the state-of-the-art in learning-based depth and visual odometry estimation methods. My novel network architecture and the novel multi-vew reprojection and forward-backward consistency constraints let the system effectively utilize the temporal information from previous frames for current frame depth and camera pose estimation. In addition, I have shown that my method can run on arbitrary length video sequences while producing temporally coherent results.

At this point in my discussion, questions remain as to 3D reconstruction from colonoscopic video sequences. The lack of groundtruth depth and the severe specularity and occlusion problems in colonoscopic videos make it impossible to directly train the RNN-DP. I will introduce how these problems are resolved and how a complete 3D colon surface can be created in the next chapter.

Chapter 6

Real-time 3D Reconstruction from Colonoscopic Videos

Colonoscopy is the most commonly used technique for lesion (typically polyps) screening inside the large intestine (colon), despite the existence of other techniques like colonography (virtual colonoscopy). It examines human colons by directly viewing video frames produced by a camera installed at the tip of an endoscope. If a polyp is detected, it can be immediately excised or biopsied by the tube's built-in tools. The Polyp Detection Rate (PDR) is an important criterion for evaluating the quality of a colonoscopy. One important reason for missing polyps is that they have not been inside the field of view of any video frame, at least not with adequate quality, that is, that the colonic surface was not fully surveyed. The accuracy (PDR) of a colonoscopy is thus highly affected by the percentage of coverage. There are at least three reasons for missing regions: 1) lack of orientations of the camera to the full circumference of the colon; 2) occlusion by the colon structure itself, especially, by the narrow rings in colon called haustral ridges; 3) poor ability of a human to notice the missing regions from the first-person perspective, especially when the endoscopist is focusing on finding polyps. One such example is in Fig. 6.1. Hong *et al.* (Hong et al., 2007) evaluated the missing region of a procedure by virtual colonoscopy. 23% of the surface was missing in a simulation.

Our solution is real-time dense 3D reconstruction of colon chunks with display of the missing regions as blank or highlighted areas in the reconstruction. I accomplish this by a novel deep-learning-driven dense SLAM (simultaneous localization and mapping) system that can produce a camera trajectory and a dense reconstructed surface for colon chunks (small lengths of colon, typically 8 to 20cm in length). In this chapter, I

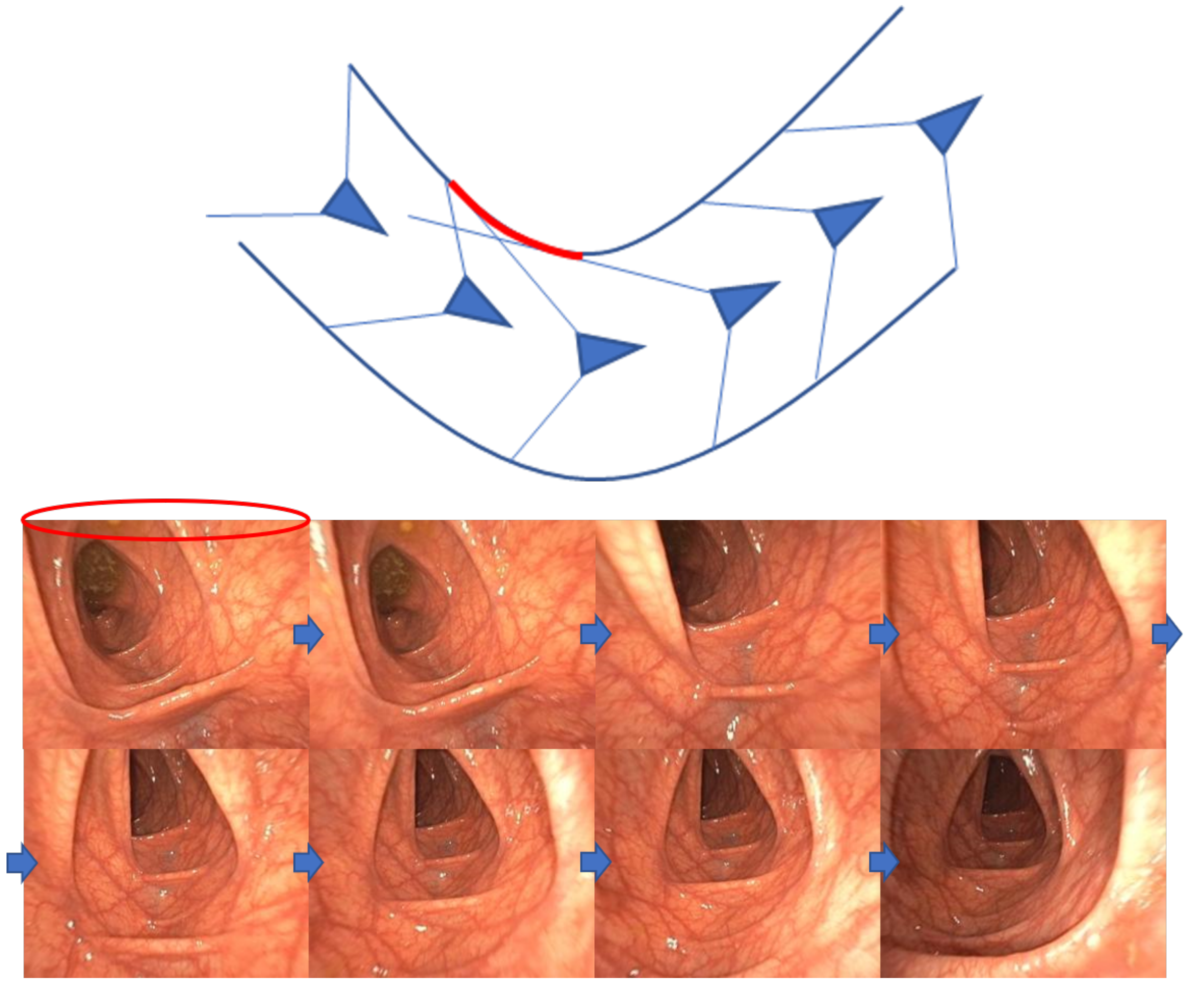


Figure 6.1: One example of a missing region in colonoscopy. The camera is quickly moving through a high-curvature region (flexure). The above image is an illustration of the camera path, their orientations and their field of view. The red region is not inside the field of view of any of the camera poses. The respective region is marked by a red circle in the bottom snapshots.

will introduce the full pipeline of our RNN-SLAM system but with more focus on the deep learning part of the system. First, I introduce a CNN-based informative frame selector that automatically selects frames that are clear enough for depth estimation. Then I introduce an RNN-based method for depth and visual odometry estimation from colonoscopic video.

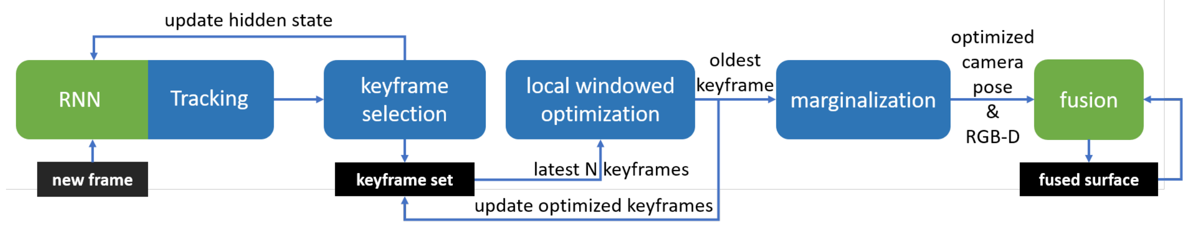


Figure 6.2: Full RNNSLAM pipeline.

6.1 Full pipeline

The full pipeline of our RNNSLAM is described in Fig. 6.2. This pipeline is adapted from DSO (Engel et al., 2018b). The original DSO includes the tracking, keyframe selection, local windowed optimization and marginalization modules. We generate a unified model by adding an RNN to the tracking module and adding a fusion module to the end. In particular, we propose an strategy to interactively combine RNN with the DSO framework. At a high level, the functions of these modules are

1. CNN-based informative frame selection.
2. RNN: predict depth and pose for each informative input frame.
3. Tracking: based on RNN prediction, refine the camera pose based on intensity.
4. Keyframe selection: make decisions to establish new keyframes and to update RNN's hidden states.
5. Local windowed optimization, i.e., bundle adjustment: jointly optimize some sparse depth values in each keyframe and their camera poses.
6. Marginalization: marginalize and output the oldest keyframe's camera pose.
7. Fusion: use the RGB values, the RNN-predicted depth map and optimized camera poses to fuse into a global, textured mesh.

Steps 4-7 have been mainly developed by my colleague, Ruibin Ma. The full details can be found in (Ma et al., 2019). I will introduce the training of CNN-based informative frame selection and the RNN-based depth and pose estimation in detail in the following sections.

6.2 CNN-based informative frame selection

Endoscopic videos contain a large fraction of non-informative frames. Non-informative frames includes tissue surface being obscured by fecal matter, motion blur, the camera being too close to the tissue surface, water flushing (in colonoscopic video), etc. Explicitly extracting features and training classifiers to identify these various kinds of non-informative frames is very difficult. A deep learning method, on the other hand, can directly learn from raw images to distinguish informative frames from non-informative frames without the need of manually crafted features; thus, it is very suitable for this task.

Distinguishing informative frames from non-informative frames is a binary classification problem if provided labels. We have adopted the VGG16 (Simonyan and Zisserman, 2014) network architecture; other network architectures such as googlenet (Szegedy et al., 2015) or resnet (He et al., 2016) could certainly be used as well. The input to the network is a single RGB frame and the output is its probability of being an informative-frame.

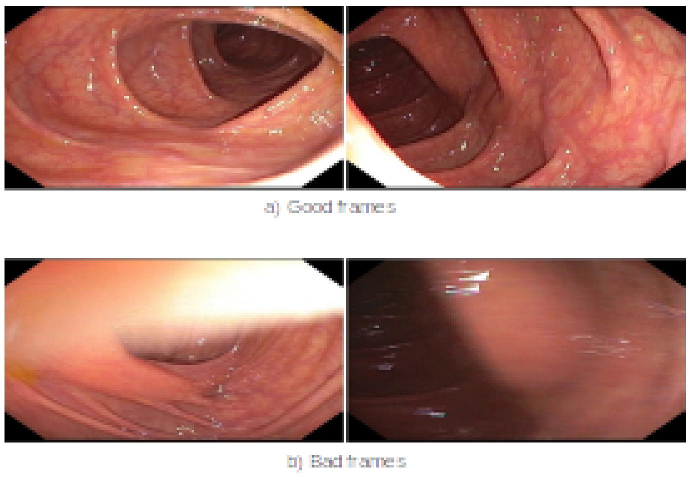


Figure 6.3: Example of informative and non-informative frames in a colonoscopic video.

Figure 6.3 shows an example of informative and non-informative frames in a colonoscopic video. We manually divided 50,000 images from five patients into the two classes as our training data. We tested the performance of the trained model on two other patients; it achieves 98.6 % accuracy.

6.3 RNN-based depth and visual odometry estimation from colonoscopic video

In Chapter 5, I introduced a Recurrent Neural Network architecture for Depth and Pose estimation (RNN-DP). RNN-DP is composed of a depth estimation network and a camera pose estimation network. However, it cannot be directly trained on colonoscopy videos because there is no groundtruth depth available. In addition, the pose estimation network in RNN-DP is trained based on image reprojection error, which is severely affected by the specular points and occlusions in colonoscopy videos. Therefore, in this section I present several new strategies that allow RNN-DP to be successfully trained on colonoscopy videos.

Preparation of training data To solve the problem of the lack of groundtruth depth, I used Structure from Motion (SfM) (Schönberger and Frahm, 2016) to produce a sparse depth map for each individual colonoscopy video frame. These sparse depth maps are then used as groundtruth for RNN-DP training. The sparse depth map for each frame is composed of a subset of points that are visible to the frame. I collected 60 colonoscopy videos, each containing about 20K frames. Then we grouped every 200 consecutive frames into a subsequence with an overlap of 100 frames with the previous subsequence. Thereby I generated about 12K subsequences from 60 colonoscopy videos. Then I ran SfM on all the subsequences to generate sparse depth maps for each frame. The sparse depth map generated by SfM is scale-ambiguous. Therefore, I further normalized the depth map within each subsequence by dividing by the median depth value of each subsequence. Finally, I broke the colonoscopic video frames and the corresponding sparse depth maps into 10-frame subsequences. These subsequences were then used to train RNN-DP.

Details of training The full training pipeline of our RNN-DP is shown in Figure 6.4. The colonoscopic training part corresponds to our new strategies. To avoid the error from specularity (saturation), I computed a specularity mask M^t_{spec} for each frame based on an intensity threshold. Image reprojection error at saturated regions were explicitly masked out by M^t_{spec} during training. Figure 6.5 shows an example of a specularity mask. Colonoscopy images also contain severe occlusions by haustral ridges, so a point in one image may not have any matching point in other images. The original RNN-DP did not handle occlusion explicitly. In order to properly train it on colonoscopy videos,

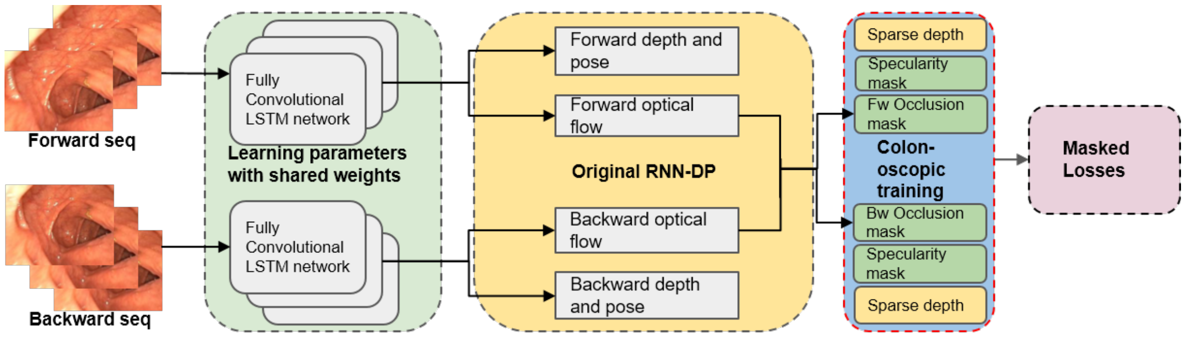
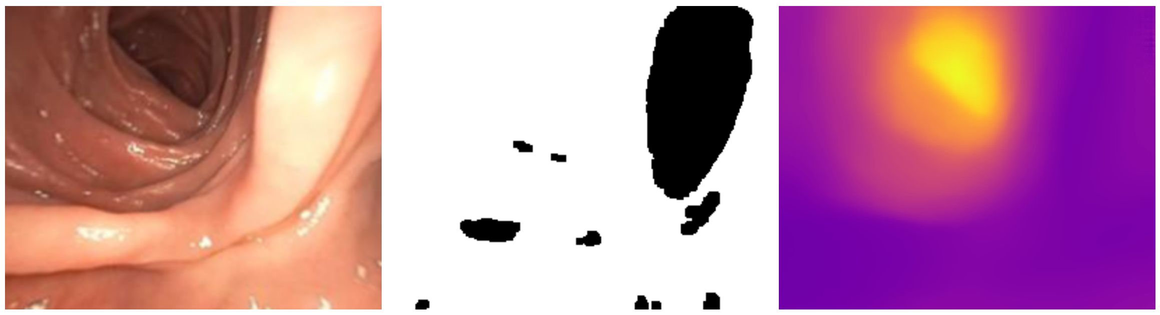


Figure 6.4: The full training pipeline of our RNN-DP for colonoscopic videos.



(a) Input image (b) Specularity mask (c) Predicted depth

Figure 6.5: Example of specularity mask and depth prediction from RNN-DP.

I compute an occlusion mask M^t_{occ} to explicitly mask out image reprojection errors at occluded regions. Following Yin's(Yin and Shi, 2018) work, the occlusion mask is determined by a forward-backward geometric consistency check. Formally it is defined as

$$M^t_{OCC} = \begin{cases} 1 & \text{if } \Delta f_{t\to s} < max(\alpha, \beta ||f_{t\to s}||_2) \\ 0 & \text{otherwise} \end{cases} \tag{6.1}$$

where $f_{t\to s}$ is the optical flow from current frame t to a neighboring frame s; $\Delta f_{t\to s}$ is a flow difference computed by a forward-backward consistency check; α and β were set to be (3.0, 0.05). Figure 6.6 shows an example of the occlusion mask M^t_{OCC}.

Our improved RNN-DP outputs frame-wise depth maps and tentative camera poses (relative to the previous keyframe). They are used to initialize the photoconsistency-based tracking (Engel et al., 2018b) that refines the camera pose.

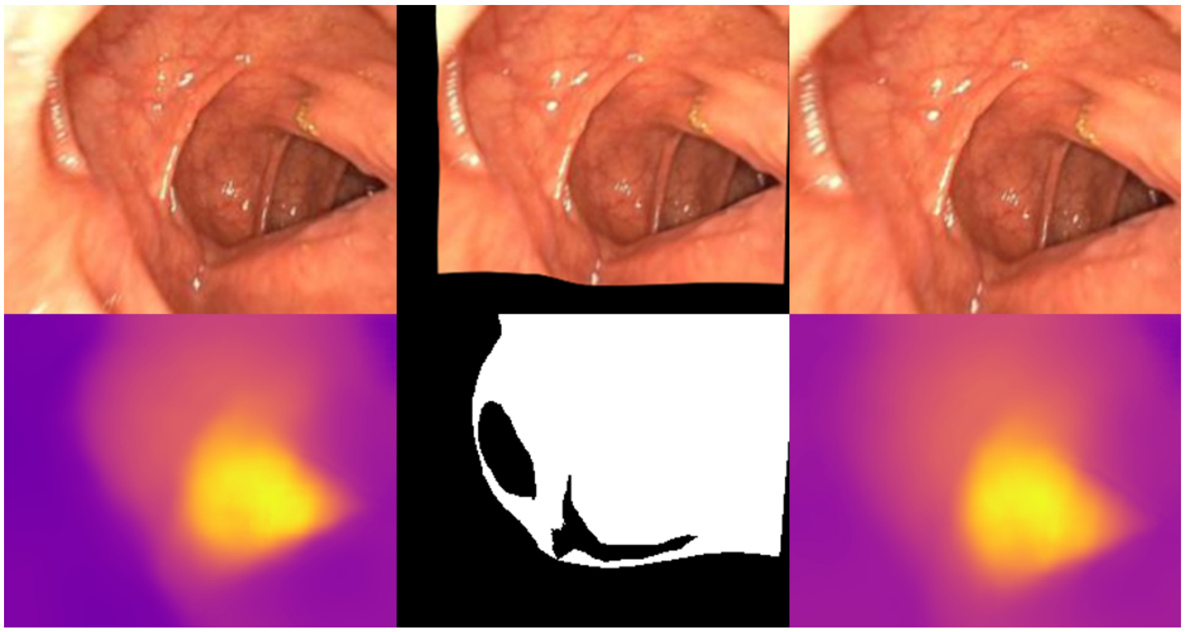


Figure 6.6: Example of occlusion mask. In the top row from left to right are the current input image, the warped image from the previous view to the current view, and the previous image. In the bottom row from left to right are the predicted depth map of the current image, the occlusion mask for the projection and the predicted depth for the previous view.

6.4 RNN-driven SLAM system

The SLAM system is improved using the RNN-DP network introduced above. In the keyframe selection module, when a new keyframe is established, the original DSO-SLAM used the dilated projections of existing active points to set the depth map for this keyframe, which is used in the new tracking tasks. The resulting depth map is sparse, noisy and subject to scale drift. In our method we set the depth map for this keyframe using the depth prediction from the network. Our depth maps are dense, more accurate and scale-consistent. As a result, using the RNN-DP-produced depth maps make the SLAM system easier to bootstrap, which is known to be a common problem for SLAM. In addition, the SLAM system improves the result of raw RNN-DP predictions by optimization; this improvement is important to eliminate accumulated camera drift of RNN-DP. In summary, this is a win-win strategy.

6.5 Results

6.5.1 Qualitative results

Depth Map Prediction Fig. 6.7 shows depth maps estimated by RNN-DP. Given a colonoscopic video sequence, the RNN-DP produces high quality depth maps in real time.

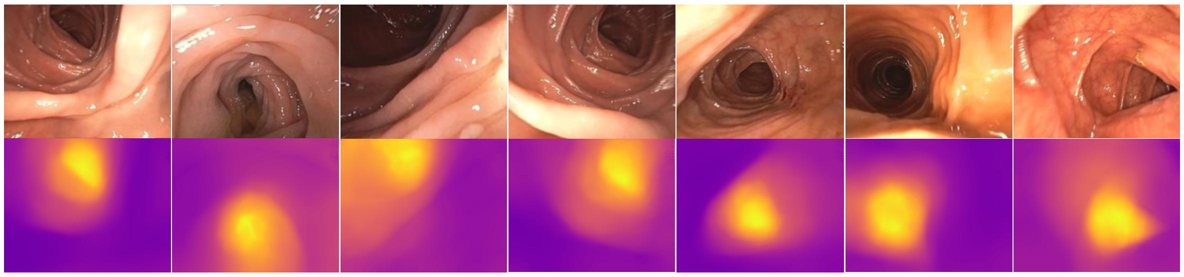


Figure 6.7: Examples of our RNN-DP estimated depth maps.

Fusion Process Using the process briefly detailed in Section 6.1, the estimated depth maps are incrementally fused into a colon chunk. Fig. 6.8 shows the incremental fusion process of a chunk of colon. The camera is moving along with the latest keyframe. The snapshots were captured in real time.

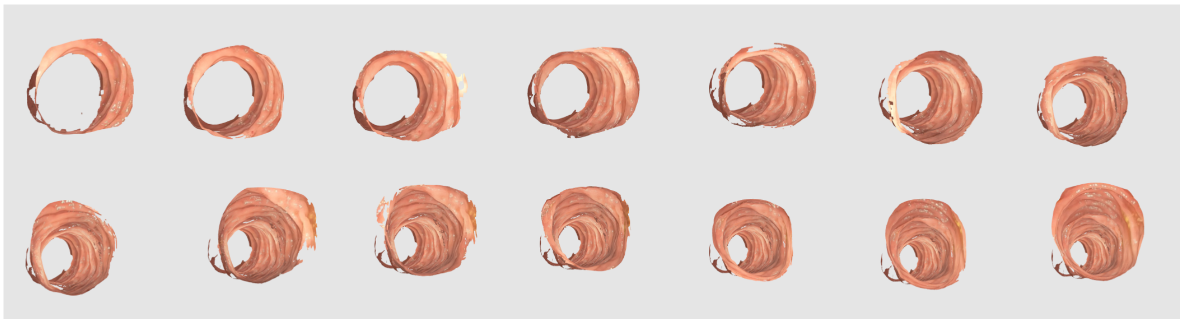


Figure 6.8: The fusion process for a chunk of a colon (from left to right, then top to bottom).

Reconstructions Fig. 6.9 shows 12 reconstruction examples. To date, our method is the only one to generate high quality meshes of real colonoscopic videos. The reconstructions capture the curvature of the colonic surface such as the haustral ridges. They can be further used for missing region analysis.

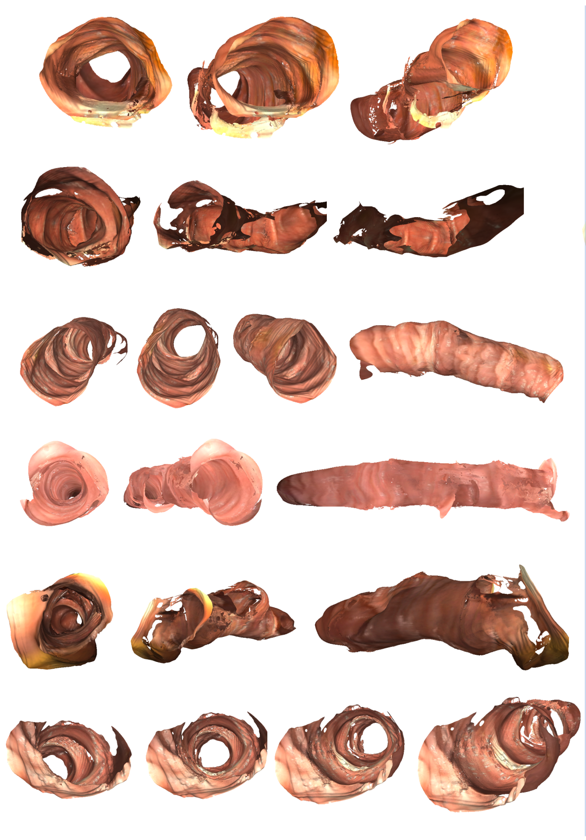
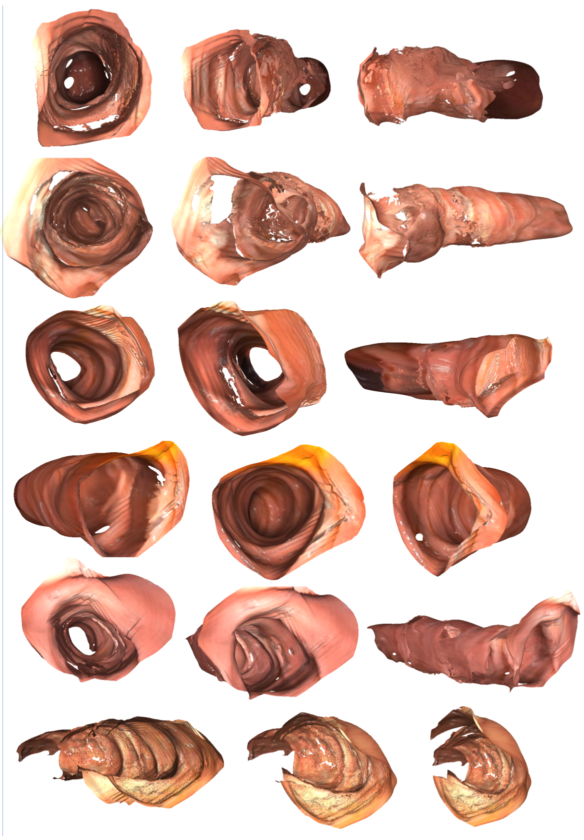


Figure 6.9: 12 reconstructed colon chunks from three or four points of view each.

Missing Region Visualization The primary objective of our method is to visualize regions missed in a colonoscopy. Using our method, the missing regions are explicitly shown as the blank regions in the colon surface (wherever the cylinder is not complete). In Figs. 6.10, 6.11 and 6.12 we show three examples of missing regions visualized by our system from clinical colonoscopic videos. In Fig. 6.10 the top part (red circle) of the colonic surface is not surveyed by the camera. Consequently, the respective part of the 3D model is missing. In Fig. 6.11 the region behind a haustral ridge was blocked by it. The missing regions are highlighted in the reconstruction (by black color). In Fig. 6.12 there is a larger portion of the surface missing because the camera never turned toward the left side of the images. Half of the colonic surface is missing. The missing regions in these examples have been verified by our colonoscopist coauthor, Dr. McGill.

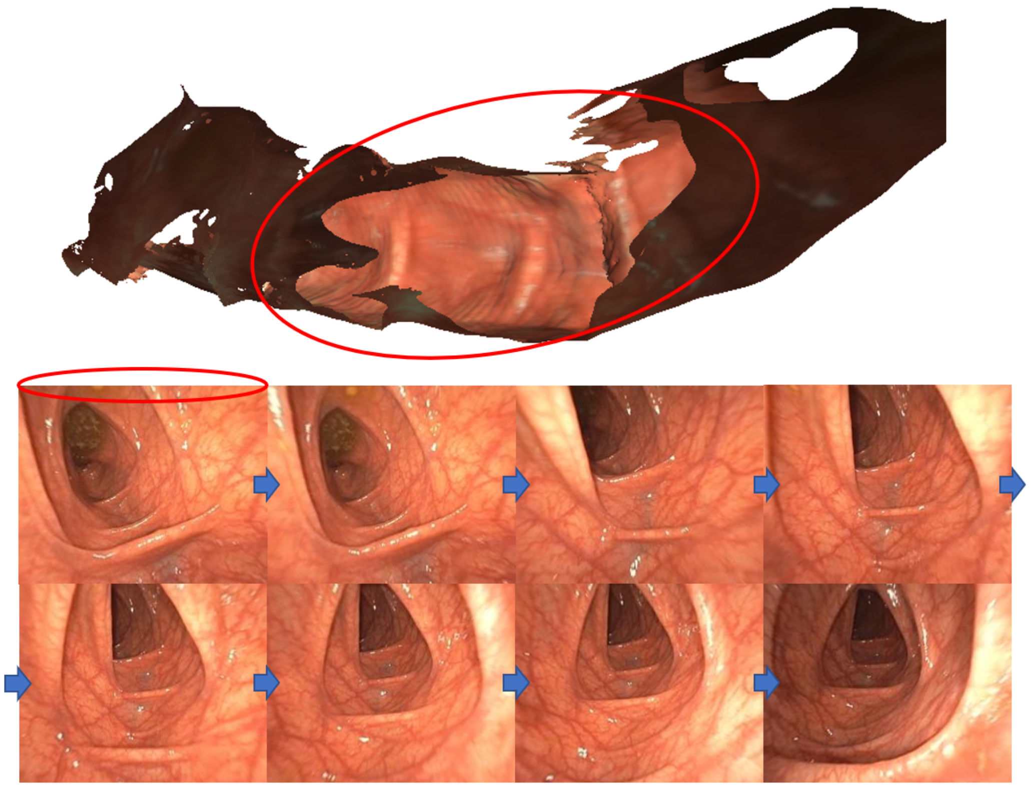


Figure 6.10: Missing region caused by lack of camera orientations.

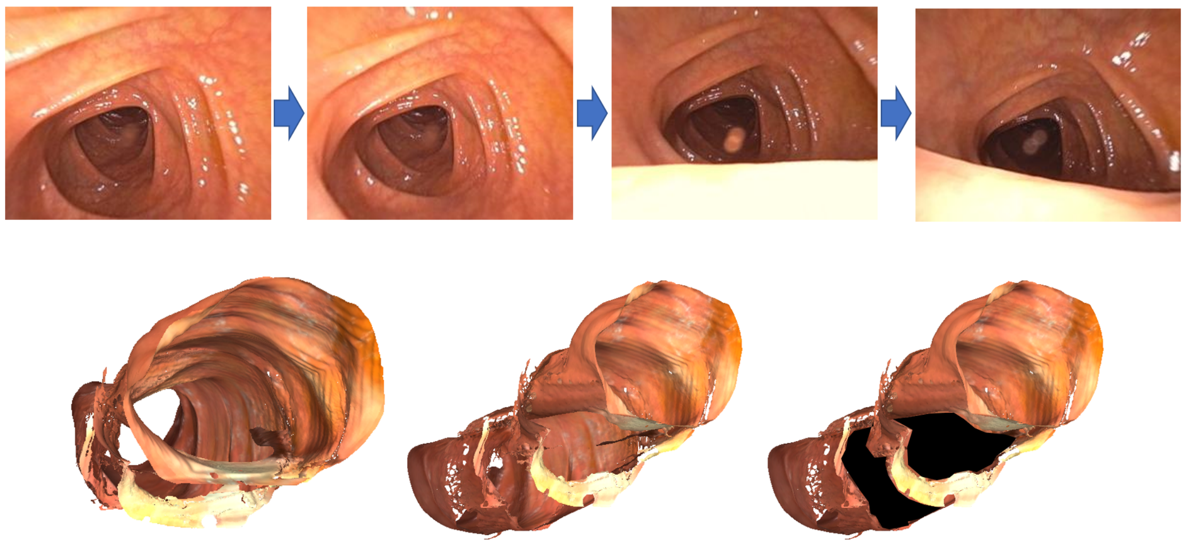


Figure 6.11: Missing region caused by a haustral ridge occlusion.

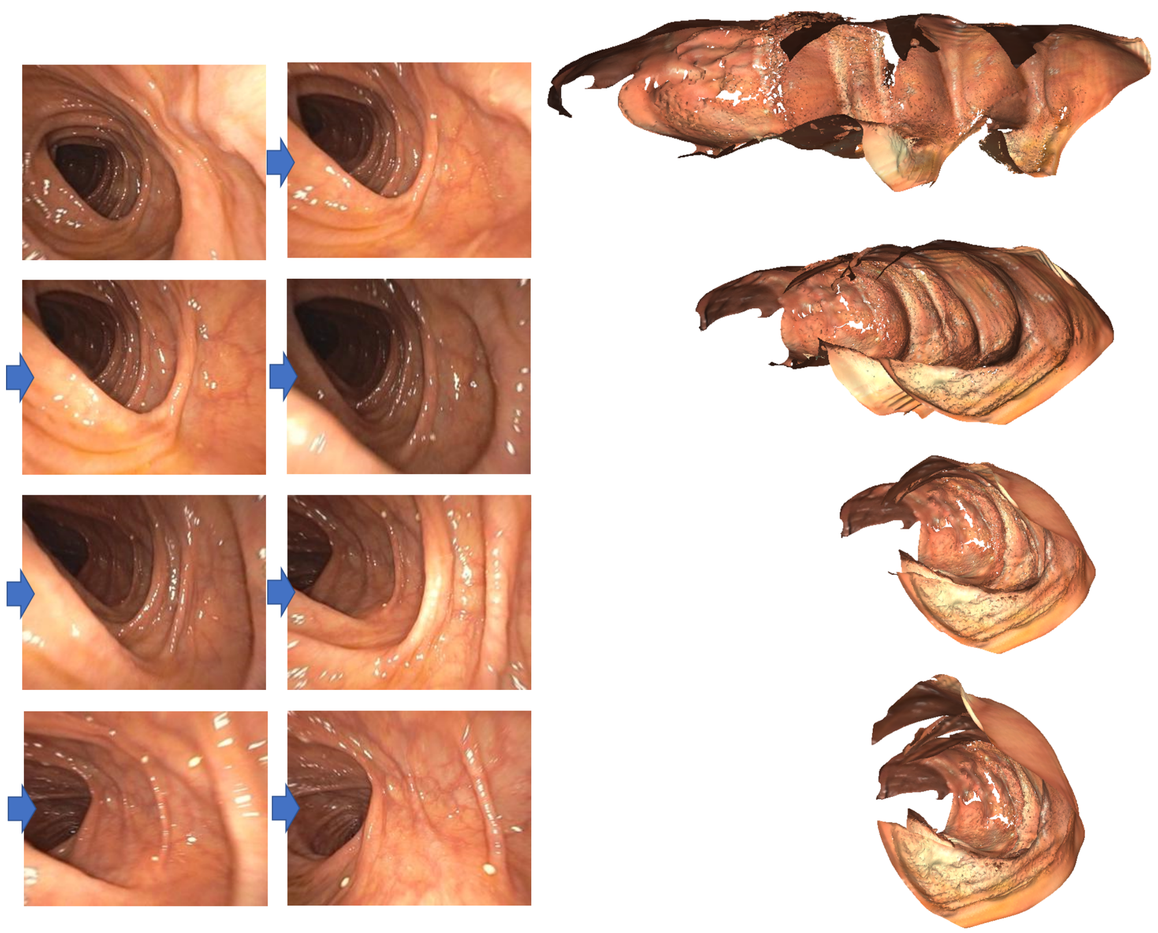


Figure 6.12: Missing region caused by lack of camera orientations. Half of the colon wall is missing because it was not surveyed by the camera (bottom right side of the snapshots).

6.5.2 Quantitative results

In this section, I provide some quantitative evaluation results of our RNNSLAM. Since there is no accurate dense depth map available as groundtruth, I only provide quantitative results for the trajectory accuracy.

Trajectory accuracy

I used an open-source library named evo (Grupp, 2017) for quantitative camera trajectory evaluation. To demonstrate the superiority of our RNNSLAM, I compare it to both the-state-of-the-art geometric reasoning based method (DSO (Engel et al., 2018b)) and deep learning based method (RNN-DP (Wang et al., 2019b)). In order to conduct the quantitative comparison, a groundtruth trajectory is needed. To generate high quality camera trajectories in an offline manner, I used colmap (Schönberger and Frahm, 2016), a state-of-the-art SfM software that incorporates pairwise exhausted matching and global bundle adjustment. These trajectories were then used as "groundtruth" for our evaluation

Evaluation metrics I use the absolute pose error (APE) to evaluate global consistency between the real-time system estimated and the colmap-generated "groundtruth" trajectory. I define the relative pose error E_i between two poses $P_{gt,i}, P_{est,i} \in \mathrm{SE}(3)$ at timestamp i as

$$E_i = (P_{gt,i})^{-1} P_{est,i} \in \mathrm{SE}(3) \tag{6.2}$$

The APE is defined as

$$APE_i = ||trans(E_i)||_2 \tag{6.3}$$

where $trans(E_i)$ refers to the translational components of the relative pose error. Then different statistics can be calculated on the APEs of all timestamps, e.g., the RMSE:

$$\mathrm{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} APE_i^2} \tag{6.4}$$

The estimated trajectories from RNNSLAM, DSO and RNN-DP are at their own scale and sampling rates. Therefore, before computing the APE, an additional scale alignment and data association step was applied.

The quantitative evaluation results are shown in Fig. 6.13 and Table 6.1. For all the results, the lower, the better since I am measuring the error. Fig. 6.13 shows evaluation results on one colonoscopic sequence. Fig. 6.13.a shows the absolute pose error (APE) of the three approaches across different time steps in the colonoscopic sequence; it can be seen that our result (red) has the lowest APE at most times. Fig. 6.13.b shows corresponding statistics computed using the APE across the whole sequence; it is clear that our result is significantly better than the other two approaches. Fig. 6.13.c shows a top-down view of the trajectories of the three approaches together with the groundtruth (colmap); and our result (red) aligns more closely to the groundtruth. I repeated the evaluation in Fig. 6.13 for 12 testing colonoscopic sequences, and in table 6.1 I show the statistics of Fig. 6.13.b but averaged across 12 colonoscopic sequences: I achieved the best result on all the metrics.

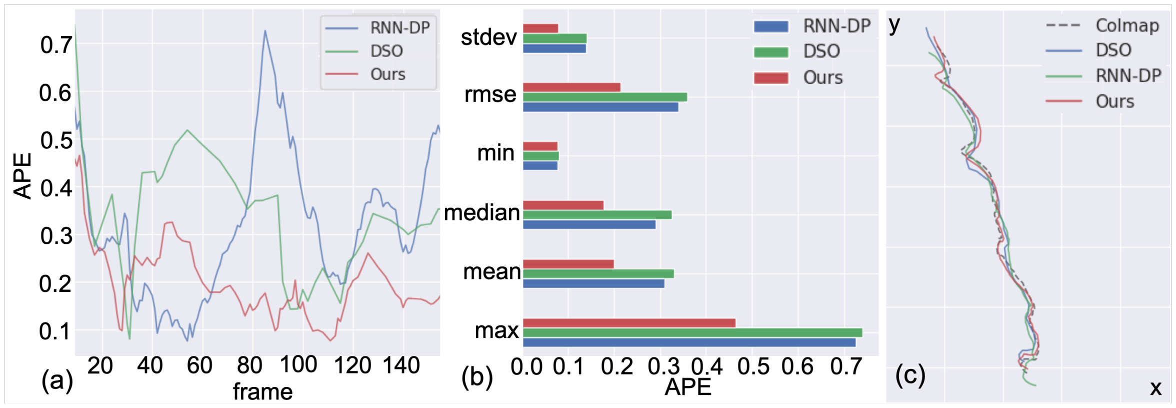


Figure 6.13: Evaluation result on one colonscopy sequence. (a) APE of the three approaches across the whole sequence. (b) Statistics based on APE. (c) A bird's-eye view of the full trajectories.

Method	rmse	std	min	median	mean	max
RNN-DP	0.617	0.253	0.197	0.518	0.560	1.229
DSO	0.544	0.278	0.096	0.413	0.465	1.413
Ours	**0.335**	**0.157**	**0.074**	**0.272**	**0.294**	**0.724**

Table 6.1: Average statistics based on the APE across 12 colonoscopic sequences

Chapter 7

Conclusion and Discussion

In this chapter I review the contributions of this dissertation and discuss related issues and future directions. In Section 7.1 I summarize the support for the claims and thesis specified in Chapter 1. In Section 7.2 I discuss the general issues related to this dissertation, and in Section 7.3 I discuss the future research directions and other possible applications that our methods can be extended to.

7.1 Summary of contributions

The claims for contributions and the thesis stated in Chapter 1 were supported in Chapter 3-6, as follows:

1. *A novel recurrent neural network that can take advantage of temporal information for supervised or unsupervised learning of monocular video visual odometry and depth.*

   The fusion-guided SfMS method described in Chapter 3 can produce a textured 3D surface, called an endoscopogram, in an offline manner. However, for some endoscopic procedures such as colonoscopy, real-time is a very high priority. Therefore, a deep-learning-based real-time depth and odometry estimation method (RNN-DP) was introduced in Chapter 5. RNN-DP utilizes recurrent units, called convolutional LSTMs, in convolutional neural networks that enable a multi-view depth and pose estimation scheme.

2. *An innovative combination of depth and pose estimation network that allows the RNN to be trained through two novel loss functions.*

The RNN-DP also innovatively combines depth and pose estimations to 1) allow the pose net to benefit from having depth as extra input and 2) allow the depths to benefit from multi-view reprojection constraints. Two novel losses, a forward-backward flow consistency loss and a multi-view image reprojection loss, were also introduced, allowing the RNN-DP to be trained with very few or even no groundtruth depth maps. This is important for cases such as endoscopic videos where there no groundtruth depth maps are available. I demonstrated the training of RNN-DP in both supervised and unsupervised fashions on the KITTI driving dataset; the results beat the state-of-the-art.

3. *A novel approach that interactively combines RNN with visual SLAM so as to achieve real-time surface reconstruction from colonoscopic video. The prior knowledge learned by RNN provides a good initialization for the SLAM. The SLAM, on the other hand, performs optimization based regularization to the estimated depth and pose that resolves the drifting problem.*

   The RNN-DP mentioned above can be trained in an unsupervised manner using multi-view image reprojection loss and forward-backward flow consistency loss in outdoor scenes. However, the drastic lighting condition changes, severe occlusions, and vast number of specular points in colonoscopic videos prevent the RNN-DP from being trained directly in an unsupervised manner. In order to overcome these problems, I introduced several new training strategies for the RNN-DP in Chapter 6, Section 6.3. Specularity and occlusion masks were introduced to exclude those regions from computing the image reprojection loss. Sparse depth maps were also generated using SfM for for every small chunk where the colon can be treated as not deforming. These sparse depth maps were then used as groundtruth that provide additional guidance in training the RNN-DP.

   Once trained, the RNN-DP can generate depth maps and camera poses through a forward prediction. However, since there is no post processing, the depth maps cannot be directly fused due to accumulated error in predicted camera poses. Therefore, as described in Chapter 6 we proposed an RNN-SLAM framework wherein the SLAM is initialized using the predicted depth maps and camera poses from RNN-DP. Such initialization resolves the boostrapping and scale drift problem of an ordinary SLAM system. At the same time, the depth maps and camera poses are also optimized by the SLAM system through local bundle adjustment, which allows the depth maps to be fused through a surfel meshing method.

4. *An approach that integrates fusion into the iterative algorithmic frame-by-frame 3D reconstruction so as to produce more temporally consistent results.*

   Reconstruction of deforming surfaces imaged in endoscopic videos appears to require both a frame-based reconstruction method and a surface-to-surface registration method. However, the frame-based reconstruction technique SfMS is an iterative algorithm that lacks temporal constraints. In addition, the surface-to-surface registration method TSD is an iterative algorithm that requires good initialization. As described in Chapter 3, combining the two into a fusion-guided SfMS method incorporates temporal constraints into the SfMS reconstruction and eliminates the need of manual selection of good initial frames for the TSD registration. This is achieved by using TSD to produce a fused reference surface that is shared by all frame-by-frame reconstructions to estimate their reflectance models and guide the SfS reconstructions. The fusion-guided SfMS achieved fully automatic reconstruction from endoscopic video to a high quality endoscopogram without manual intervention.

5. *An optimization-based multi-view texture fusion algorithm that minimize within-patch intensity gradient magnitude differences and inter-patch-boundary color differences.*

   The light source is attached to the tip of an endoscope, so the lighting conditions drastically change across different frames in an endoscopic video. To produce an endoscopogram with seamless texture, I introduced a novel multi-surface texture fusion method in Chapter 3, Section 3.5. The method is composed of an MRF-based texture initialization algorithm and an optimization-based texture fusion algorithm. The method tries to eliminate differences across texture seam boundaries while preserving the gradient level details. As a result, the texture fusion method eliminates the seam boundaries across different frames while preserving finer details such as blood vessels. This method can also be adopted to other applications, e.g., 3D reconstructed faces from internet photos, that often have illumination inconsistencies of the texture on the fused surfaces.

6. *A novel deep learning-based informative frame selection method that can automatically select frames that are suitable for 3D reconstruction.*

There are a lot of garbage frames in endoscopic videos due to motion blur and the camera being too close to the tissue surface, and in colonoscopy, water flushing, fecal matter, etc. These frames cannot be used for 3D reconstruction and will make the 3D reconstruction algorithm fail if not being correctly removed. Furthermore, the 3D reconstruction from colonoscopic video requires to be real time, so the garbage frame removal method needs to be real time as well. Chapter 6, Section 6.2 introduced a deep-learning-based informative frame selection method that can achieve 98.6% accuracy and runs in real time.

Technical contribution claims are summarized as follows:

1. *A simulator for non-rigid 3D reconstruction evaluation.*

   In order to quantitatively evaluate the performance of our method of 3D reconstruction from endoscopy, a simulator and a simulation-based evaluation method were introduced in Chapter 3, Section 3.6. First a textured CT surface that preserves the realism of both texture and geometry is generated. Then realistic deformation, lighting and even the camera path are created using a 3D graphics software called Blender. Finally, a synthetic endoscopic video together with groundtruth depth maps are generated through rendering. A series of experiments were performed using the synthetic endoscopic videos to quantitatively evaluate the performance of our method comparing to a state-of-the-art 3D reconstruction method.

2. *A full pipeline that integrates reconstruction, geometry fusion and texture fusion into an automatic process.*

   As mentioned above I introduced the fusion-guided SfMS, TSD and texture fusion methods in Chapter 3. Software that combines all three methods is also produced and documented at (Wang, 2020). The software allows the generation of an endoscopogram from an endoscopic video to be fully automatic.

3. *Clinical evaluation software for tumor drawing on endoscopic video or endoscopogram and transfer to the CT space.*

   The purpose of generating an endoscopogram from an endoscopic video is to aid the radiation treatment planning. In order to do so, the tumor identified in an endoscopic video or an endoscopogram needs to be transferred and visualized in the CT space. The clinical evaluation software, called Endo2CT, introduced in

Chapter 3, Section 3.7 serves this purpose. A contour can be drawn on either the endoscopogram or the selected endoscopic frames to circle a tumor. The drawn region is then dilated and transferred into the CT space for visualization. Furthermore, the drawn region can be saved for later usage or compared to the GTV (Gross Tumor Volume) generated by a CT-based tumor localization method. A preliminary clinical study has also been carried out using the Endo2CT software. We analyzed 12 patient cases for whom endoscopic videos and planning CT scans were both available. The results suggest notable clinical benefits of fusing endoscopic video with CT using our proposed fusion-guided SfMS and TSD methods for radiation treatment planning.

On the basis of the above contributions and their successful use in 3D reconstruction from endoscopic videos, I have established the following thesis:

Thesis: *Endoscopography reconstructs a full 3D textured surface from an endoscopic video. We call this textured surface an endoscopogram. This opens the door for novel 3D visualizations of patient anatomy derived solely from endoscopic data and their combination with other sources of anatomical information.*

The code for RNN-DP is available at `https://github.com/wrlife/RNN_depth_pose`. The code for fusion-guided SfMS is available at `https://bitbucket.org/unc_endoscopogram/fusion-guided_sfms/src/master/`. The code for clinical evaluation tool is available at `https://bitbucket.org/unc_endoscopogram/endo2ct/src/master/`.

7.2 Aspects and generality of the contributions

1. *Deformable surface reconstruction*

   This dissertation generally dealt with the deformable surface reconstruction problem. The common methods are usually restricted to the use of an RGB-D camera that with an additional depth channel, objects with specific shape or the use of a template. I treated the problem as a single-view 3D reconstruction and a deformable registration problem. This treatment bypasses the needs of dense matching and triangulation from the video of a deformable object. But it also introduces a new challenge, the ambiguities in single-view 3D reconstruction. The fusion-guided SfMS method introduced in this dissertation deals with the ambiguity by using the multi-view reconstructed sparse points as a guidance. The

RNN-DP method deals with the ambiguity by learning robust prior knowledge from a large amount of data and using novel regularization losses. Our methods do not require any prior knowledge about the scene or object and thus can be adopted to any deformable surface reconstruction problem.

2. *Reflectance model estimation*

   In SfMS the surface reflectance model is estimated separately for each individual frame. Since the albedo or the texture of the surface is assumed to be constant in our SfS model, separate reflectance models for each frame account for the texture changes and complex tissue properties as well as the inter-reflection of lights in the human anatomy environments. At the same time, to prevent the inconsistency in the estimated geometries caused by the separate reflectance model estimations, I used a fused surface that shared by all reflenctance model estimations. Such separate reflectance model estimation with shared reference geometry helps resolve the problems in SfMS reconstruction and the TSD registration to some extent. From the above, we can also see that in order to further improve the SfMS method, more sophisticated modeling of the reflectance model and temporal consistency between adjacent frames should be considered.

3. *Training a depth estimation network*

   Under perspective projection the depth is scale-ambiguous: all depth values along a viewing ray are valid. Therefore, in single-view depth estimation local smoothness and gradient level similarity are more important than the absolute scale of the depth. When training a depth estimation network, there are usually multiple losses; two of the most commonly used are absolute depth error and gradient level depth error. We can assign a lot larger weight to the gradient level depth error, e.g., 100 times larger weights than the absolute depth error, because the gradient level depth error is not affected by the scale ambiguity in the depth.

   Another important aspect in training a depth estimation network is incorporating photometric and geometric constraints. Such constraints are the foundation for traditional geometric reasoning-based multi-view 3D reconstruction methods. Joint estimation of surface normal, semantic segmentation and optical flow has also shown to be successful in training a depth estimation network.

4. *Combination of CNNs with geometric reasoning based methods*

   The combination of SLAM and CNN that I introduced in Chapter 6 demonstrated that the traditional computer vision methods and CNN-based methods are not contradictory but instead complementary. The CNNs learns strong priors of a problem given large amount of training data. However, the trained models in many cases are still non-deterministic and thus in some cases lead to unpredictable results. Traditional methods, on the other hand are mostly deterministic but require strong prior knowledge or assumptions. The trained CNN models can perfectly serve this purpose.

   Traditional methods can also be integrated with CNNs in many forms. For example, CNNs can take the results of traditional methods as input, CNNs can use traditional methods in their loss computations, and the outputs of CNNs and of traditional methods can be combined through post-processing.

7.3 Future work

In this section, I discuss some remaining theoretical and technical issues as well as some future research directions.

1. *Fusion-guided SfMS can only fuse up to twenty frames*

   One limitation of fusion-guided SfMS is that it can only fuse up to twenty frames. This is because with the increasing number of frames the computational complexity of the groupwise TSD registration grows exponentially. There are several potential solutions to this problem: 1) only compute the attraction forces within a small number of neighboring frames in temporal order and compute the attraction forces in a sliding window fashion; 2) perform incremental fusion instead of groupwise fusion, wherein the fused surface can be used to guide the following reconstructions and evolve over time; 3) perform hierarchical fusion by generating multiple endoscopograms, where each endoscopogram corresponds to the reconstruction of a short sequence or the fusion of multiple endoscopograms.

2. *Using the patient CT extracted surface to guide the reconstruction*

   As mentioned above, template-based 3D reconstruction is commonly used for the deformable surface 3D reconstruction. In our endoscopic case, for each patient we have the endoscopic video and the corresponding patient CT. A 3D surface can be extracted from the CT. This surface can potentially be used as a template to guide the 3D reconstruction from the endoscopic video. However, the large deformation between CT and endoscopy and the low resolution of the CT extracted surface need to be taken into consideration. One possible solution is to use the deformation between the CT extracted surface to the endoscopogram as the initial deformation between the CT space and the endoscopy. Then the deformed CT surface can be refined using the shading information in each individual image to obtain more detailed geometry.

3. *The quality of the depth maps and camera poses estimated by RNN-DP are not good enough for direct fusion*

   One of the bottlenecks in our current RNN-SLAM system is the windowed average. The windowed average was developed to solve the inconsistency in the estimated dense depth maps by RNN-DP. If the RNN-DP estimated depth maps had good enough quality, the windowed average would not be necessary. Several possible solutions to such problem are the following: 1) Incorporate a geometric consistency loss in training the RNN-DP. Current RNN-DPs are based on photometric consistency and forward-backward flow consistency losses. Further imposing geometric consistency in the training of RNN-DP is a straightforward solution. 2) Integrate feature-metric bundle adjustment into the training of the RNN-DP. The feature-metric bundle adjustment proposed by Tang *et al.* can potentially be integrated into the RNN-DP to enforce multi-view geometry constraints in the form of feature-metric error. The multi-veiw photometric consistency loss being used in RNN-DP performs poorly for illumination changes, moving objects and textureless regions. Using CNN-learned features instead of the raw pixels for the bundle adjustment can potentially improve the performance of RNN-DP.

4. *The performance of RNN-DP is bounded by the accuracy of the estimated depth maps from SfM*

   In order to train the RNN-DP on the colonoscopic data, I proposed using SfM-produced depth maps as additional supervision. However, this also cause the performance of RNN-DP to be bounded by the accuracy of SfM's estimation. Currently, there are groups using synthetic data to train depth estimation networks for colonoscopic videos and showed promising results. Synthetic datasets have the advantage of unlimited sizes, easy acquisition and accurate groundtruth. But the adaptation of a synthetic-data-trained network to a real dataset is not trivial. Therefore, the combination of SfM-produced depth maps and synthetic colonoscopic dataset can potentially take advantage of both methods and improve the performance of RNN-DP on colonoscopic data. Creating a more realistic colonoscopic video simulator or a better transfer learning technique are also possible future directions.

5. *Extension of our methods to other applications*

   The RNN-DP is invented for continuous depth and camera pose estimation from monocular videos. But its architecture, which interleaves recurrent units with convolutional units, can be adopted to many other applications that needs temporal information for prediction. One such example can be optical flow estimation from video sequences. Both the network architecture and the forward-backward flow consistency loss can be useful for optical flow estimation. Another similar application is image registration. The forward-backward flow consistency loss can be imposed on the estimated deformation fields between images.

   The fusion-guided SfMS and the RNN-SLAM methods introduced in this dissertation were developed upon pharyngoscopic and colonoscopic videos. However, they are not restricted to these two type of endoscopies. Other endoscopies such as esophagogastroduodenoscopy, enteroscopy, otoscopy, and bronchoscopy can also be target applications of our systems.

BIBLIOGRAPHY

Ahmed, A. H. and Farag, A. A. (2006). A new formulation for shape from shading for non-lambertian surfaces. In *Computer Vision and Pattern Recognition (CVPR)*.

Anjyo, K. (1997). Tour into the picture. In *ACM SIGGRAPH 97 Visual Proceedings: The art and interdisciplinary programs of SIGGRAPH'97*, page 301.

Atapour-Abarghouei, A. and Breckon, T. P. (2018). Real-time monocular depth estimation using synthetic data with domain adaptation via image style transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 18, page 1.

Bartoli, A., Gérard, Y., Chadebecq, F., Collins, T., and Pizarro, D. (2015). Shape-from-template. *Pattern Analysis and Machine Intelligence (PAMI)*, 37(10):2099–2118.

Bauer, M. and Bruveris, M. (2011). A new riemannian setting for surface registration. *arXiv preprint arXiv:1106.0620*.

Beardsley, P., Torr, P., and Zisserman, A. (1996). 3d model acquisition from extended image sequences. In *European conference on computer vision*, pages 683–695. Springer.

Bronstein, A. M., Bronstein, M. M., and Kimmel, R. (2006). Generalized multidimensional scaling: a framework for isometry-invariant partial surface matching. *Proceedings of the National Academy of Sciences*, 103(5):1168–1172.

Casser, V., Pirk, S., Mahjourian, R., and Angelova, A. (2019). Unsupervised monocular depth and ego-motion learning with structure and semantics. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0.

Choy, C. B., Xu, D., Gwak, J., Chen, K., and Savarese, S. (2016). 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *European conference on computer vision*, pages 628–644. Springer.

Community, B. O. (2018). *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam.

Davison, A. J., Reid, I. D., Molton, N. D., and Stasse, O. (2007). Monoslam: Real-time single camera slam. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (6):1052–1067.

Dellaert, F., Seitz, S. M., Thorpe, C. E., and Thrun, S. (2000). Structure from motion without correspondence. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No. PR00662)*, volume 2, pages 557–564. IEEE.

Dosovitskiy, A., Fischer, P., Ilg, E., Hausser, P., Hazirbas, C., Golkov, V., Van Der Smagt, P., Cremers, D., and Brox, T. (2015). Flownet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2758–2766.

Durou, J.-D., Falcone, M., and Sagona, M. (2008). Numerical methods for shape-from-shading: A new survey with benchmarks. *Computer Vision and Image Understanding*, 109(1):22–43.

Eigen, D. and Fergus, R. (2015). Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2650–2658.

Eigen, D., Puhrsch, C., and Fergus, R. (2014). Depth map prediction from a single image using a multi-scale deep network. In *Advances in neural information processing systems*, pages 2366–2374.

Elad, A. and Kimmel, R. (2003). On bending invariant signatures for surfaces. *IEEE Transactions on pattern analysis and machine intelligence*, 25(10):1285–1295.

Engel, J., Koltun, V., and Cremers, D. (2018a). Direct sparse odometry. *IEEE transactions on pattern analysis and machine intelligence*, 40(3):611–625.

Engel, J., Koltun, V., and Cremers, D. (2018b). Direct sparse odometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence.*

Engel, J., Schöps, T., and Cremers, D. (2014). Lsd-slam: Large-scale direct monocular slam. In Fleet, D., Pajdla, T., Schiele, B., and Tuytelaars, T., editors, *Computer Vision – ECCV 2014*, pages 834–849, Cham. Springer International Publishing.

Freedman, D., Blau, Y., Katzir, L., Aides, A., Shimshoni, I., Veikherman, D., Golany, T., Gordon, A., Corrado, G., Matias, Y., et al. (2020). Detecting deficient coverage in colonoscopies. *arXiv preprint arXiv:2001.08589.*

Fu, H., Gong, M., Wang, C., Batmanghelich, K., and Tao, D. (2018a). Deep ordinal regression network for monocular depth estimation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

Fu, H., Gong, M., Wang, C., Batmanghelich, K., and Tao, D. (2018b). Deep ordinal regression network for monocular depth estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2002–2011.

Gaidon, A., Wang, Q., Cabon, Y., and Vig, E. (2016). Virtual worlds as proxy for multi-object tracking analysis. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4340–4349.

Garg, R., BG, V. K., Carneiro, G., and Reid, I. (2016). Unsupervised cnn for single view depth estimation: Geometry to the rescue. In *European Conference on Computer Vision*, pages 740–756. Springer.

Gatzke, T., Grimm, C., Garland, M., and Zelinka, S. (2005). Curvature maps for local shape comparison. In *Shape Modeling and Applications, 2005 International Conference*, pages 244–253. IEEE.

Geiger, A., Lenz, P., Stiller, C., and Urtasun, R. (2013). Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*.

Geiger, A., Lenz, P., and Urtasun, R. (2012). Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Godard, C., Mac Aodha, O., and Brostow, G. J. (2017). Unsupervised monocular depth estimation with left-right consistency. In *CVPR*, volume 2, page 7.

Gotardo, P. F. and Martinez, A. M. (2011). Non-rigid structure from motion with complementary rank-3 spaces. In *CVPR 2011*, pages 3065–3072. IEEE.

Grupp, M. (2017). evo: Python package for the evaluation of odometry and slam. https://github.com/MichaelGrupp/evo.

Gu, X., Wang, Y., Chan, T. F., Thompson, P. M., and Yau, S.-T. (2004). Genus zero surface conformal mapping and its application to brain surface mapping. *IEEE Transactions on Medical Imaging*, 23(8):949–958.

Han, Y., Lee, J.-Y., and Kweon, I. S. (2013). High quality shape from a single rgb-d image under uncalibrated natural illumination. In *International Conference on Computer Vision (ICCV)*.

Hartley, R. and Zisserman, A. (2003). *Multiple view geometry in computer vision*. Cambridge university press.

Hartley, R. I. (1997). In defense of the eight-point algorithm. *IEEE Transactions on pattern analysis and machine intelligence*, 19(6):580–593.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Hoiem, D., Efros, A. A., and Hebert, M. (2005). Automatic photo pop-up. In *ACM SIGGRAPH 2005 Papers*, pages 577–584.

Hong, D., Tavanapong, W., Wong, J., Oh, J., and De Groen, P. C. (2014). 3d reconstruction of virtual colon structures from colonoscopy images. *Computerized Medical Imaging and Graphics*, 38(1):22–33.

Hong, W., Wang, J., Qiu, F., Kaufman, A., and Anderson, J. (2007). Colonoscopy simulation. In *Proc.SPIE.*

Horn, B. K. (1970). Shape from shading: A method for obtaining the shape of a smooth opaque object from one view.

Hur, J. and Roth, S. (2017). Mirrorflow: Exploiting symmetries in joint optical flow and occlusion estimation. In *Proc. of Int'l Conf. on Computer Vision (ICCV).*

Iyer, G., Murthy, J. K., Gunshi Gupta, K., and Paull, L. (2018). Geometric consistency for self-supervised end-to-end visual odometry. *arXiv preprint arXiv:1804.03789.*

Jaderberg, M., Simonyan, K., Zisserman, A., et al. (2015). Spatial transformer networks. In *Advances in neural information processing systems*, pages 2017–2025.

Kao, C. Y., Osher, S., and Qian, J. (2004). Lax–friedrichs sweeping scheme for static hamilton–jacobi equations. *Journal of Computational Physics*, 196(1):367–391.

Kaufman, A. and Wang, J. (2008). 3d surface reconstruction from endoscopic videos. In *Visualization in Medicine and Life Sciences*, pages 61–74. Springer.

Kim, S. M., McCulloch, T. M., and Rim, K. (2000). Pharyngeal pressure analysis by the finite element method during liquid bolus swallow. *Annals of Otology, Rhinology & Laryngology*, 109(6):585–589.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980.*

Koenderink, J. J., Van Doorn, A. J., and Stavridi, M. (1996). Bidirectional reflection distribution function expressed in terms of surface scattering modes. In *European Conference on Computer Vision (ECCV).*

Kong, C. and Lucey, S. (2019). Deep non-rigid structure from motion. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1558–1567.

Kumar, A. C., Bhandarkar, S. M., and Mukta, P. (2018). Depthnet: A recurrent neural network architecture for monocular depth prediction. In *1st International Workshop on Deep Learning for Visual SLAM,(CVPR)*, volume 2.

Kuznietsov, Y., Stückler, J., and Leibe, B. (2017). Semi-supervised deep learning for monocular depth map prediction. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6647–6655.

Laina, I., Rupprecht, C., Belagiannis, V., Tombari, F., and Navab, N. (2016). Deeper depth prediction with fully convolutional residual networks. In *3D Vision (3DV), 2016 Fourth International Conference on*, pages 239–248. IEEE.

Lee, J., Park, S. W., Kim, Y. S., Lee, K. J., Sung, H., Song, P. H., Yoon, W. J., and Moon, J. S. (2017). Risk factors of missed colorectal lesions after colonoscopy. *Medicine*, 96(27).

Li, Z. and Snavely, N. (2018). Megadepth: Learning single-view depth prediction from internet photos. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Liu, F., Shen, C., and Lin, G. (2015). Deep convolutional neural fields for depth estimation from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5162–5170.

Long, J., Shelhamer, E., and Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440.

Luo, Y., Ren, J., Lin, M., Pang, J., Sun, W., Li, H., and Lin, L. (2018). Single view stereo matching. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

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

Mahjourian, R., Wicke, M., and Angelova, A. (2018). Unsupervised learning of depth and ego-motion from monocular video using 3d geometric constraints. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Mahmood, F. and Durr, N. J. (2018). Deep learning and conditional random fields-based depth estimation and topographical reconstruction from conventional endoscopy. *Medical image analysis*, 48:230–243.

Malti, A. and Bartoli, A. (2014). Combining conformal deformation and cook–torrance shading for 3-d reconstruction in laparoscopy. *Biomedical Engineering, IEEE Transactions on*, 61(6):1684–1692.

Malti, A., Bartoli, A., and Collins, T. (2011). Template-based conformal shape-from-motion from registered laparoscopic images. In *Conference on Medical Image Understanding and Analysis (MIUA)*.

Malti, A., Bartoli, A., and Collins, T. (2012). Template-based conformal shape-from-motion-and-shading for laparoscopy. In *Information Processing in Computer-Assisted Interventions (IPCAI)*.

Mayer, N., Ilg, E., Hausser, P., Fischer, P., Cremers, D., Dosovitskiy, A., and Brox, T. (2016). A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4040–4048.

Mohr, R., Quan, L., and Veillon, F. (1995). Relative 3d reconstruction using multiple uncalibrated images. *The International Journal of Robotics Research*, 14(6):619–632.

Mur-Artal, R., Montiel, J. M. M., and Tardos, J. D. (2015). Orb-slam: a versatile and accurate monocular slam system. *IEEE Transactions on Robotics*, 31(5):1147–1163.

Mur-Artal, R., Montiel, J. M. M., and Tardós, J. D. (2015). ORB-SLAM: A versatile and accurate monocular SLAM system. *IEEE Trans. Robotics*, 31(5):1147–1163.

Mur-Artal, R. and Tardós, J. D. (2017). ORB-SLAM2: an open-source SLAM system for monocular, stereo, and RGB-D cameras. *IEEE Trans. Robotics*, 33(5):1255–1262.

Newcombe, R. A., Lovegrove, S. J., and Davison, A. J. (2011). Dtam: Dense tracking and mapping in real-time. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2320–2327. IEEE.

Pfaehler, E., De Jong, J. R., Dierckx, R. A., van Velden, F. H., and Boellaard, R. (2018). Smart (simulation and reconstruction) pet: an efficient pet simulation-reconstruction tool. *EJNMMI physics*, 5(1):1–18.

Pieper, S., Halle, M., and Kikinis, R. (2004). 3d slicer. In *2004 2nd IEEE international symposium on biomedical imaging: nano to macro (IEEE Cat No. 04EX821)*, pages 632–635. IEEE.

Pollefeys, M., Van Gool, L., Vergauwen, M., Verbiest, F., Cornelis, K., Tops, J., and Koch, R. (2004). Visual modeling with a hand-held camera. *International Journal of Computer Vision*, 59(3):207–232.

Prados, E. and Faugeras, O. (2005). Shape from shading: a well-posed problem? In *Computer Vision and Pattern Recognition (CVPR)*.

Prados, E. and Faugeras, O. (2006). Shape from shading. In Paragios, N., Chen, Y., and Faugeras, O. D., editors, *Handbook of mathematical models in computer vision*, pages 375–388. Springer.

Qi, X., Liao, R., Liu, Z., Urtasun, R., and Jia, J. (2018). Geonet: Geometric neural network for joint depth and surface normal estimation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Rogers, B. and Graham, M. (1979). Motion parallax as an independent cue for depth perception. *Perception*, 8(2):125–134.

Ronneberger, O., Fischer, P., and Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer.

Ros, G., Sellart, L., Materzynska, J., Vazquez, D., and Lopez, A. M. (2016). The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3234–3243.

Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386.

Salzmann, M. and Fua, P. (2010). Deformable surface 3d reconstruction from monocular images. *Synthesis Lectures on Computer Vision*, 2(1):1–113.

Saxena, A., Sun, M., and Ng, A. Y. (2008). Make3d: Learning 3d scene structure from a single still image. *IEEE transactions on pattern analysis and machine intelligence*, 31(5):824–840.

Scaramuzza, D., Martinelli, A., and Siegwart, R. (2006). A flexible technique for accurate omnidirectional camera calibration and structure from motion. In *Fourth IEEE International Conference on Computer Vision Systems (ICVS'06)*, pages 45–45. IEEE.

Scharstein, D. and Szeliski, R. (2002). A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International journal of computer vision*, 47(1-3):7–42.

Schonberger, J. L. and Frahm, J.-M. (2016). Structure-from-motion revisited. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4104–4113.

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

Schwab, R. J., Gefter, W. B., Pack, A. I., and Hoffman, E. A. (1993). Dynamic imaging of the upper airway during respiration in normal subjects. *Journal of Applied Physiology*, 74(4):1504–1514.

Shin, Y., Qadir, H. A., Aabakken, L., Bergsland, J., and Balasingham, I. (2018). Automatic colon polyp detection using region based deep cnn and post learning approaches. *IEEE Access*, 6:40950–40962.

Siegel, R. L., Miller, K. D., and Jemal, A. (2019). Cancer statistics, 2019. *CA: a cancer journal for clinicians*, 69(1):7–34.

Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Sun, J., Ovsjanikov, M., and Guibas, L. (2009). A concise and provably informative multi-scale signature based on heat diffusion. In *Computer graphics forum*, volume 28, pages 1383–1392. Wiley Online Library.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9.

Tang, C. and Tan, P. (2018). Ba-net: Dense bundle adjustment network. *arXiv preprint arXiv:1806.04807*.

Tateno, K., Tombari, F., Laina, I., and Navab, N. (2017). Cnn-slam: Real-time dense monocular slam with learned depth prediction. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 00, pages 6565–6574.

Thirion, J.-P. (1998). Image matching as a diffusion process: an analogy with maxwell's demons. *Medical image analysis*, 2(3):243–260.

Tokgozoglu, H. N., Meisner, E. M., Kazhdan, M., and Hager, G. D. (2012). Color-based hybrid reconstruction for endoscopy. In *Computer Vision and Pattern Recognition Workshops (CVPRW)*.

Triggs, B., McLauchlan, P. F., Hartley, R. I., and Fitzgibbon, A. W. (1999). Bundle adjustment—a modern synthesis. In *International workshop on vision algorithms*, pages 298–372. Springer.

Ummenhofer, B., Zhou, H., Uhrig, J., Mayer, N., Ilg, E., Dosovitskiy, A., and Brox, T. (2017). Demon: Depth and motion network for learning monocular stereo. In *IEEE Conference on computer vision and pattern recognition (CVPR)*, volume 5, page 6.

Van Rijn, J. C., Reitsma, J. B., Stoker, J., Bossuyt, P. M., Van Deventer, S. J., and Dekker, E. (2006). Polyp miss rate determined by tandem colonoscopy: a systematic review. *American Journal of Gastroenterology*, 101(2):343–350.

Vijayanarasimhan, S., Ricco, S., Schmid, C., Sukthankar, R., and Fragkiadaki, K. (2017). Sfm-net: Learning of structure and motion from video. *arXiv preprint arXiv:1704.07804*.

Wang, P., Berzin, T. M., Brown, J. R. G., Bharadwaj, S., Becq, A., Xiao, X., Liu, P., Li, L., Song, Y., Zhang, D., et al. (2019a). Real-time automatic detection system increases colonoscopic polyp and adenoma detection rates: a prospective randomised controlled study. *Gut*, 68(10):1813–1819.

Wang, R. (2020). Fusion-guided sfms.

Wang, R., Pizer, S. M., and Frahm, J.-M. (2019b). 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 (CVPR).*

Wang, R., Schwörer, M., and Cremers, D. (2017a). Stereo dso: Large-scale direct sparse visual odometry with stereo cameras. In *International Conference on Computer Vision (ICCV), Venice, Italy.*

Wang, S., Clark, R., Wen, H., and Trigoni, N. (2017b). Deepvo: Towards end-to-end visual odometry with deep recurrent convolutional neural networks. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2043–2050. IEEE.

Wu, C., Varanasi, K., Liu, Y., Seidel, H.-P., and Theobalt, C. (2011). Shading-based dynamic shape refinement from multi-view video under general illumination. In *International Conference on Computer Vision (ICCV).*

Xingjian, S., Chen, Z., Wang, H., Yeung, D.-Y., Wong, W.-K., and Woo, W.-c. (2015). Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *Advances in neural information processing systems*, pages 802–810.

Yang, N., Wang, R., Stückler, J., and Cremers, D. (2018a). Deep virtual stereo odometry: Leveraging deep depth prediction for monocular direct sparse odometry. In *European Conference on Computer Vision*, pages 835–852. Springer.

Yang, N., Wang, R., Stueckler, J., and Cremers, D. (2018b). Deep virtual stereo odometry: Leveraging deep depth prediction for monocular direct sparse odometry. In *ECCV.*

Yao, Y., Luo, Z., Li, S., Fang, T., and Quan, L. (2018). Mvsnet: Depth inference for unstructured multi-view stereo. In *The European Conference on Computer Vision (ECCV).*

Yin, Z. and Shi, J. (2018). Geonet: Unsupervised learning of dense depth, optical flow and camera pose. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

Zaharescu, A., Boyer, E., Varanasi, K., and Horaud, R. (2009). Surface feature detection and description with applications to mesh matching. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 373–380. IEEE.

Zeng, Y., Wang, C., Gu, X., Samaras, D., and Paragios, N. (2013). A generic deformation model for dense non-rigid surface registration: A higher-order mrf-based approach. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 3360–3367. IEEE.

Zhan, H., Garg, R., Saroj Weerasekera, C., Li, K., Agarwal, H., and Reid, I. (2018). Unsupervised learning of monocular depth estimation and visual odometry with deep feature reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 340–349.

Zhang, J. (2018). Inverse-consistent deep networks for unsupervised deformable image registration. *arXiv preprint arXiv:1809.03443*.

Zhang, R., Tsai, P.-S., Cryer, J. E., and Shah, M. (1999). Shape-from-shading: a survey. *Pattern Analysis and Machine Intelligence*, 21(8):690–706.

Zhao, H., Li, H., Maurer-Stroh, S., and Cheng, L. (2018). Synthesizing retinal and neuronal images with generative adversarial nets. *Medical image analysis*, 49:14–26.

Zhao, Q., Price, J., Pizer, S. M., Niethammer, M., Alterovitz, R., and Rosenman, J. G. (2015). Surface registration in the presence of missing patches and topology change. In *MIUA*, pages 8–13.

Zhao, Q., Price, T., Pizer, S., Niethammer, M., Alterovitz, R., and Rosenman, J. (2016). The endoscopogram: A 3d model reconstructed from endoscopic video frames. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 439–447. Springer.

Zhou, T., Brown, M., Snavely, N., and Lowe, D. G. (2017). Unsupervised learning of depth and ego-motion from video. In *CVPR*, volume 2, page 7.

Zhou, T., Tulsiani, S., Sun, W., Malik, J., and Efros, A. A. (2016). View synthesis by appearance flow. In *European conference on computer vision*, pages 286–301. Springer.

Zollhöfer, M., Dai, A., Innmann, M., Wu, C., Stamminger, M., Theobalt, C., and Nießner, M. (2015). Shading-based refinement on volumetric signed distance functions. *ACM Transactions on Graphics (TOG)*, 34(4).

Zou, Y., Luo, Z., and Huang, J.-B. (2018). Df-net: Unsupervised joint learning of depth and flow using cross-task consistency. In *The European Conference on Computer Vision (ECCV)*.