2012

3D Colon Segment and Endoscope Motion Reconstruction from Colonoscopy Video

DongHo Hong
Iowa State University

3D colon segment and endoscope motion reconstruction from colonoscopy video

by

DongHo Hong

A dissertation submitted to the graduate faculty
In partial fulfillment of the requirements for the degree of
DOCTOR OF PHILOSOPHY

Major: Computer Science

Iowa State University
Ames, Iowa
2012

TABLE OF CONTENTS

ACKNOWLEDGEMENT

I am heartily thankful to my research advisor Dr. Wallapak Tavanapong. This thesis would not have been possible without her excellent guidance, patience, and support.

I would like to thank Dr. Johnny S. Wong, my research co-advisor, for his great support. I would also like to thank Dr. Carl K. Chang, Dr. Les Miller and Dr. Yan-Bin Jia for valuable advices.

I would like to thank my parents in law for praying for my successful academic achievements all the time.

I would like to express my special thanks to my parents. My parents have been supporting me for 33 years so that I can keep studying without distraction in an affluent environment.

I would like to express my deepest gratitude to my wife TaeHee Kim. She was always there supporting me with infinite patience. I would also like to thank my son Jinu for praying for me and waiting for me to complete this work and come back as a good father.

This work was supported in part by the National Science Foundation STTR Grant No. IIP-0956847 and EndoMetric Corporation. Opinions and findings in this paper are those of the authors and do not represent those of the funding agencies.

This thesis is dedicated to my father DaeSoon Hong, my mather SunJo Rho, and my wife TaeHee Kim.

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

This dissertation addresses the problem of reconstruction of the 3D structure of colon segments and the 3D motion of an endoscope from colonoscopy images. Colonoscopy is currently the preferred screening modality for early colorectal cancer (CRC) detection, claiming about 50,000 annual deaths in the US. During colonoscopy, the endoscopist performs diagnosis and/or treatment based on images of the interior of the human colon taken by the camera at the tip of the endoscope.

The reconstruction of the virtual colon and the endoscope motion from colonoscopy images has the potential to (1) improve endoscopist's understanding of the colon structure of the patient and the endoscopist's navigation technique at the time of colonoscopy; (2) determine the estimated area and location of the colon mucosa uninspected by the endoscopist; domain experts estimated that as much as 95% of the colon mucosa should be inspected; and (3) bridge the gap between virtual colonoscopy and optical colonoscopy. Besides our previous work, we found no other techniques that address the same problem in the literature. We propose a set of new algorithms and evaluate them using images inside an Olympus synthetic colon as well as colonoscopy images from real colonoscopy procedures. Edges of colon fold contours are first detected and processed to generate the wire frame of the reconstructed virtual colon. We propose a colon fold contour estimation algorithm using a single colonoscopy image. We introduce *Depth-from-Intensity*---the depth and shape estimation of colon folds using brightness intensity of pixels. We introduce an algorithm to estimate the amount of protrusion and width of colon folds and interpolate the folds and the colon wall to connect the folds in 3D space. We present our new method for reconstructing the virtual colon from sequential colonoscopy images and deriving corresponding camera motions. We discuss two quality metrics derived using the information from the reconstruction: spiral score and percentage of areas uninspected. Spiral score is found strongly correlated with the ground truth visualization quality on 159 colonoscopy videos. The percentage of areas uninspected metric is of great interest to domain experts.

Broader impact: The proposed work has potential to improve the amount of the colon mucosa inspected in clinical practice; thereby, potentially increasing the protective effect of colonoscopy against CRC. The technology may change the way colonoscopy is performed in routine practice and may be used for training new endoscopists. Finally, the algorithm design will add to existing knowledge in endoscopy video processing and may be beneficial for other endoscopy procedures.

CHAPTER 1 INTRODUCTION

Colonoscopy is currently the preferred colorectal cancer screening modality among Fecal Occult Blood Test, flexible sigmoidoscopy, CT Colonography (CTC), and Double Contrast Barium Enema. Colorectal cancers are malignant tumors that develop in the colon and rectum. Colorectal cancer claims about 50,000 lives annually in the US [1]. Its annual death rate is second only to that of lung cancer. The survival rate is higher if the cancer is found and treated early before metastasis to lymph nodes or other organs occurs. Colonoscopy allows for inspection of the inner lining of the human colon and therapeutic operations such as polyp removal in a single procedure. Over 14 million colonoscopic procedures are performed annually in the US alone [2]. This number is increasing as the current Medicare guidelines suggest that each US citizen undergo colonoscopy at least once every 10 years starting at age 50.

The colon is 150 cm long nested inside the human abdomen. The colon consists of six major anatomical segments: rectum, sigmoid, descending colon, transverse colon, ascending colon, and cecum. Colonoscopy consists of two phases: an insertion phase and a withdrawal phase. During the insertion phase, a flexible endoscope (a flexible tube with a tiny video camera at the tip) is advanced under direct vision via the anus into the rectum and then gradually into the cecum or the terminal ileum (the last part of the small intestine). During the withdrawal phase, the endoscope is gradually withdrawn. The camera has a wide-angle lens and generates a video signal of the interior of the human colon, which is displayed on a monitor for real-time diagnosis by the endoscopist. The purpose of the insertion phase is to reach the cecum or the terminal ileum. Careful mucosa inspection and diagnostic or therapeutic interventions such as biopsy, polyp removal, etc., are performed during the withdrawal phase. Colonoscopy is a complex procedure to master. Currently, the endoscopist has no prior information about the structure of the colon to navigate through this long tubular organ with many turns. The endoscopist learns the structure of the colon during the procedure.

Colonoscopy has contributed to a marked decline in the number of colorectal cancer related deaths. However, recent data suggest a significant average miss rate of 4-12% for the detection of even large polyps and cancers [3-5]. A polyp miss rate as high as 80% was reported [6]. The miss rate may be related to poor colon preparation, the location of the lesion in the colon and/or the experience of the endoscopist, but there are no prospective studies to investigate these factors. To minimize the miss rate, consensus guidelines for quality inspection were proposed in 2006 by American Society of Gastrointestinal Endoscopy and American College of Gastroenterology [7].

Recent years have seen several advances in colonoscopy. For instance, a number of algorithms have been investigated for measuring objective quality of colonoscopy (i.e., how well the colon was inspected during the procedure) [8-11], polyp detection [12, 13], anatomical landmark detection [14], 3D reconstruction of a cylinder generalized colon from colonoscopy images [15], and 3D reconstruction of the colon surface for surgical planning [16, 17], or image-guided automated colonoscopy [18].

Unlike these techniques, we investigated a set of new algorithms combined together to reconstruct a 3D virtual colon segment from one or more neighboring colonoscopy images using standard endoscopes and without additional hardware or prior knowledge of the structure of the colon. The reconstruction of the virtual colon is very challenging, but has great potential for (1) understanding the colon structure of the patient; (2) understanding the endoscopist’s navigation technique at the time of colonoscopy; (3) determining the estimated location and the area of the colon mucosa inspected by the endoscopist; and (4) deriving objective metrics of quality of colonoscopy.

1.1. Contributions

1. We are the first to investigate the problem of 3D reconstruction of the colon structure from colonoscopy images without requiring prior knowledge of the coarse structure of the colon or additional hardware except the computer used for capturing and image analysis. Most of the prior attempts focus on reconstruction of the 3D colon surface from 2D endoscopy images [16, 17, 19] for detailed surgical planning or image-guided automated colonoscopy [18]. These techniques are not suitable for reconstruction of the colon structure. As the first investigator, we discovered a number of problems that have not been explored in the literature.

   We propose the first fully automated technique for creating a virtual colon from one or more 2D colonoscopy image. The technique consists of a number of new algorithms. (1) The pre-processing step selects an informative frame (frame that is in focus and contains sufficient number of edges), detects edges that are likely of colon folds, and estimates the center of the innermost colon fold. It then determines whether the frame is a good candidate for reconstruction or not. For instance, frames with a close-up inspection of the colon wall without the distant lumen are not suitable for the reconstruction. In order to create a 3D virtual colon segment, we require closed colon fold contours to form a wire frame of the reconstructed virtual colon. (2) Automatic fold contour estimation obtains closed fold contours. We introduce two algorithms, one of which uses only a single colonoscopy image and the other using sequential images. Either of these techniques can be applied. (3) Fold shape and depth estimation estimates the fold shape and the distance from the camera (depth) for each fold. We investigated a new algorithm, *Depth-from-Intensity* (*DfI*). *DfI* first estimates the depth using the pixel brightness intensity of qualified fold pixels and then obtains the shape. *DfI* also estimates the colon fold width and height (the amount of protrusion) seen at these pixels to give a more realistic colon structure. Brightness intensity calibration is also performed before the calibrated intensity is used to calculate the depth of the pixel as the light source of the endoscope does not distribute light evenly depending on the location of the pixels. A reverse projection of the edge pixels for the frame reverses project the pixels into a local 3D coordinate system. (4) For reconstruction of the colon structure from multiple sequential frames, we extend an existing 3D model registration to transform the local model reconstructed from the current frame to the global model built from all the previous frames. Camera motions are obtained as a by-product. This method can recover a fold partly seen in one frame (e.g., behind another colon fold because the colon bends at that fold or the fold is behind the camera), but is clearly seen in a subsequent frame. (5) Rendering of the virtual colon is done when required by the endoscopist.

3. We implemented our technique, designed our experiments using an Olympus synthetic colon model, designed our performance metrics, and evaluated our techniques. The evaluation turns out to be most difficult since there is no ground truth for quantitative measurements. There are no endoscopy devices with global positioning capability to provide camera locations or orientations during colonoscopy. The colon is not rigid. It also moves due to breathing, heart condition, etc. We are not able to obtain very accurate ground truth of the colon structure during colonoscopy to account for these issues despite our several attempts. A virtual colon from a prior CT Colonography (CTC) cannot be used to obtain ground truth since the virtual colon from CTC presents the colon structure when patients are in supine or prone positions, not sideways as in colonoscopy. Another potential way to obtain very accurate ground truth is to have the endoscopist perform colonoscopy on a patient while an MRI scan is on-going;

however, such attempts have never been tried. Several precautions need to be considered before conducting this type of test on patients. Subjective evaluation may be the only option to go about for future clinical trials.

For this study, we used an Olympus synthetic colon model and measured fold distances and fold circumferences as ground truth. The measurements were taken by our domain expert. We conducted two sets of experiments. The first set is for evaluation of our reconstruction result from a single colonoscopy image. The average error per fold of the fold depths and the fold circumferences compared to the ground truth are 4.1 and 12.1 mm, respectively. The average depth error of 4.1 mm is small, less than 1/5 of the distance the endoscope camera used in this study can see (20-150 mm). The average error for estimating the circumference of around 12 mm is less than 1/10 of the circumference of folds in the colon model (140-200 mm). The second set of experiments is for evaluation of our solution using multiple colonoscopy images. We evaluated both the reconstructed endoscope motions and the reconstructed colon structures. We do not have the ground truth for the amount of the endoscope motion, but we can observe the direction of the endoscope motion from watching the video. Our method gives correct camera rotation directions for 88% of the cases, but only gives 73% correct camera translation directions of the cases used in our study. The method finds more colon folds than the technique using a single colonoscopy image as anticipated, but does not reduce the error rate for the colon structure reconstruction compared to the technique using a single colonoscopy image. We suggest further improvements.

4. We explored whether the computed results can be used to derive good quality metrics for objectively measuring quality of the colon mucosa inspection during colonoscopy. We investigated two metrics. One quality metric is called *spiral score*, a coarse measurement of

the number of times the camera sees all four sides of the colon wall. It was previously shown [20] that the spiral score metric is strongly correlated with the ground truth determined by the domain expert. However, the previous method is too slow to provide feedback to the endoscopist during the procedure. We show that we can derive this metric as feedback. Our study on 159 colonoscopy videos found that the metric has a strong correlation with the quality of the colon mucosa inspection ground truth. We implemented the software and installed it at two educational rooms in Mayo Clinic, Rochester, MN. The data is currently being analysed by a separate team of researchers for whether or not the feedback improves quality of colonoscopy performed by the fellows using those two rooms. Another quality metric is the percentage of the colon mucosa area unseen and map of these unseen areas. Our domain experts see great potential values for this metric. Using the reconstructed colon structure and reconstructed camera motions, we can derive the aforementioned measurements. We developed prototype software to illustrate the idea.

1.2. Organization

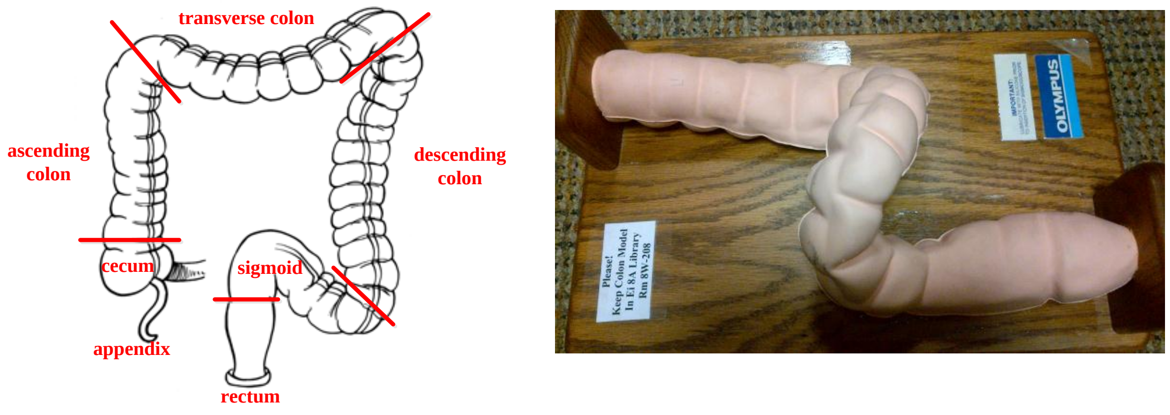
In Chapter 2, we provide background on the human colon structure and colonoscopy as well as important characteristics of endoscope camera lens (fish-eye lens) and its projection model, which causes distortion of endoscopy images. Also, we describe related work on existing 3D model reconstruction methods and 3D colon surface reconstruction using those methods. In Chapter 3, we present the pre-processing step necessary for selecting an image suitable for reconstruction. In Chapter 4, we present the colon fold contour estimation algorithm using a single colonoscopy image along with experimental results demonstrating the effectiveness of the technique. In Chapter 5, we present the shape and depth estimation of colon folds using *Depth-from-Intensity* (*DfI*) as well as supporting performance evaluation result. In Chapter 6, we discuss our solution for constructing the virtual colon using multiple images. In Chapter 7, we discuss two quality measurements derived from the reconstruction result and evaluate the effectiveness of these techniques on existing colonoscopy videos. We provide conclusions and description of our future work in Chapter 8.

CHAPTER 2 BACKGROUND AND RELATED WORK

There is no other existing work on 3D reconstruction of the structure of the colon from 2D colonoscopy video. Existing works focus on 3D reconstruction of the colon surface. To understand the complexity of our problem, we first provide background on the colon anatomy and colonoscopy. Next, we discuss camera calibration and 3D shape reconstruction techniques.

2.1. The human colon and colonoscopy

The colon is a non-rigid tube about 150 cm long connecting the small intestine to the anus. The colon consists of six major anatomical segments: rectum, sigmoid, descending colon, transverse colon, ascending colon, and cecum. See Figure 2.1. The colon has exterior layer and interior layers of muscles. The exterior muscle (tenae coli) is a muscle separated into 3 smooth flat bands that run almost the entire length of the colon. The interior muscles wrap around the colon. Haustra are small pouches caused by sacculation since the tenae coli is shorter than the colon. Haustral folds lie in between adjacent haustra. Haustral folds are colon folds we refer to in this dissertation. Colon mobility (the contraction of the muscles) moves the contents inside the colon slowly toward the rectum. The colon mucosa is the first layer of the inner lining of the colon that we can see. Some parts of a haustral fold may protrude, but the other part may be completely flat as illustrated in Figure 2.1(b). Haustral folds are not of the same size. We do not find any reports on the distances among haustral folds, the average number of folds per colon, or the average circumference of haustral folds in different sections of the colon. All these issues add additional complexity in reconstructing the colon structure.



(a) (b)

Figure 2.1 (a) Simplified diagram showing the structure of the colon; (b) A fold with only half of it protruding and a fold with all sides protruding can be observed in this Olympus synthetic colon model used for training endoscopists and used in our study. Images are courtesy of Piet C. de Groen.

As mentioned in the introduction, colonoscopy consists of two phases: an insertion phase and a withdrawal phase. During the insertion phase, a flexible endoscope (a flexible tube with a tiny video camera at the tip) is advanced under direct vision via the anus into the rectum and then gradually into the cecum or the terminal ileum (part of the small intestine). The objective during the insertion phase is to reach the cecum quickly without causing perforation or pain to the patient. During the withdrawal phase, the endoscope is gradually withdrawn and careful colon examination is expected. In the US, endoscopists are recommended to remove all detected polyps, although not all of them are malignant. The same practice may not be applied in other countries.

The tiny video camera at the tip of the endoscope has a wide-angle lens (fish-eye lens) to allow a wider view volume for physicians to examine the inside of an organ without much tilting and panning. The tip of the endoscope can be flipped upside down or left and right. Fish-eye lenses cause distortion in the projected 2D images seen by the endoscopst during colonoscopy. The lens cannot be auto focused. The endoscopist needs to place the camera at a proper distance from the colon wall in order to see the colon wall clearly. To inspect behind the haustra folds for hidden polyps, the endoscopist may move the camera back and forth, angle the camera, or inject additional air/CO_2 to deflate the fold. Some advanced video processors, such as Olympus endoscopes, adjust illumination automatically. Fujinon endoscopes have two light sources, but Olympus endoscopes have three light sources. These issues add to the complexity of the problem.

Colonoscopy is complex to master. Experienced endoscopists state that as much as possible of the colon mucosa (all sides or 360 degrees inspection) of the colon wall should be examined throughout the withdrawal phase of colonoscopy. Figure. 2.2(A-B) shows a desirable withdrawal inspection pattern in which the lumen is seen in the camera field of view (*called lumen view*) or lumen image. The colon center axis is at the center of the lumen (green area). Figure 2.2(C-D) shows a different inspection pattern where one side of the colon wall is seen each time without the lumen (*called wall view*) or wall image. Depending on the configuration of the anatomy and the location of the camera in different anatomic sections in the colon, a combination of wall view or lumen view may be required to see all mucosa.

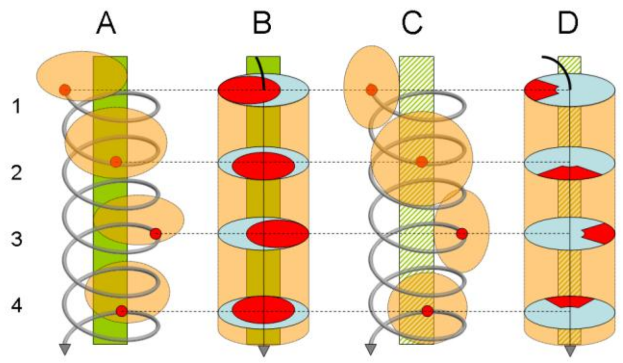


Figure 2.2 Diagram showing withdrawal inspection pattern: A and C show the spiral shape of the movement of the tip of the endoscope. B and D show the visual field in red. Green bar represents the colon lumen; A and B reflect views in which the distant proximal colon is visible (or lumen view). C and D reflect views in which the distant proximal colon is absent and only colon wall is seen (or wall view). The diagram is courtesy of our medical advisor, Piet C. de Groen.

2.2. Camera calibration

A camera calibration technique calculates intrinsic parameters of a camera and corrects a distorted image by re-projecting the object according to the perspective projection model. In this section, we discuss properties of wide-angle lenses and common projection models.

The pinhole camera model is not suitable for modeling fish-eye lens since it follows the perspective projection model to project objects in 3D space onto a 2D plane. In the perspective projection model, the wider field of view (FOV), the more distortion on a projected image. If the field of view (FOV) is 180˚, a finite sized canvas cannot contain the projected image. Figure 2.3(a) shows that 180˚ FOV ($\theta=90^\circ$) needs an infinite sized canvas ($r = f \tan 90^\circ$) where FOV is 2θ; the canvas size is $2r$ and f is a focal length. Any of the four projection models, stereographic projection, equidistance projection, equisolid angle projection, or orthogonal projection can be used for fish-eye lenses [21, 22].

$$r = 2f\tan\frac{\theta}{2} \qquad \text{(stereographic projection)}$$

$$r = f\theta \qquad \text{(equidistance projection)}$$

$$r = 2f\sin\frac{\theta}{2} \qquad \text{(equisolid angle projection)}$$

$$r = f\sin\theta \qquad \text{(orthogonal projection)}$$

The fish-eye lens projects front hemispherical surroundings on the hemispherical surface. The above projection models represent how to transform a pixel on a hemispherical surface to a flat surface. Figure 2.3(b) illustrates transformation of u to u' by the equidistance projection model where u is a pixel on the hemisphere and u' is the corresponding pixel on the flat surface. This transformation causes distortion of images. Every frame of a colonoscopy video is distorted like this. Therefore, the geometric measurement directly on the colonoscopy image might yield incorrect result. For this reason, camera calibration or image correction is essential as a preprocessing step to most of the 3D reconstruction techniques [16, 17, 19]. Kannala et al. [22] provide a generic calibration method for various fish-eye projection models and Stehle et al. [23] use this generic method for 3D reconstruction of the colon surface. They need to calibrate the camera with several images of a planar calibration object. In this study, we do not calibrate the camera, but assume that we know the projection model for the camera lens. That is, we let the user choose the desired camera projection model among the above fish-eye lens models.

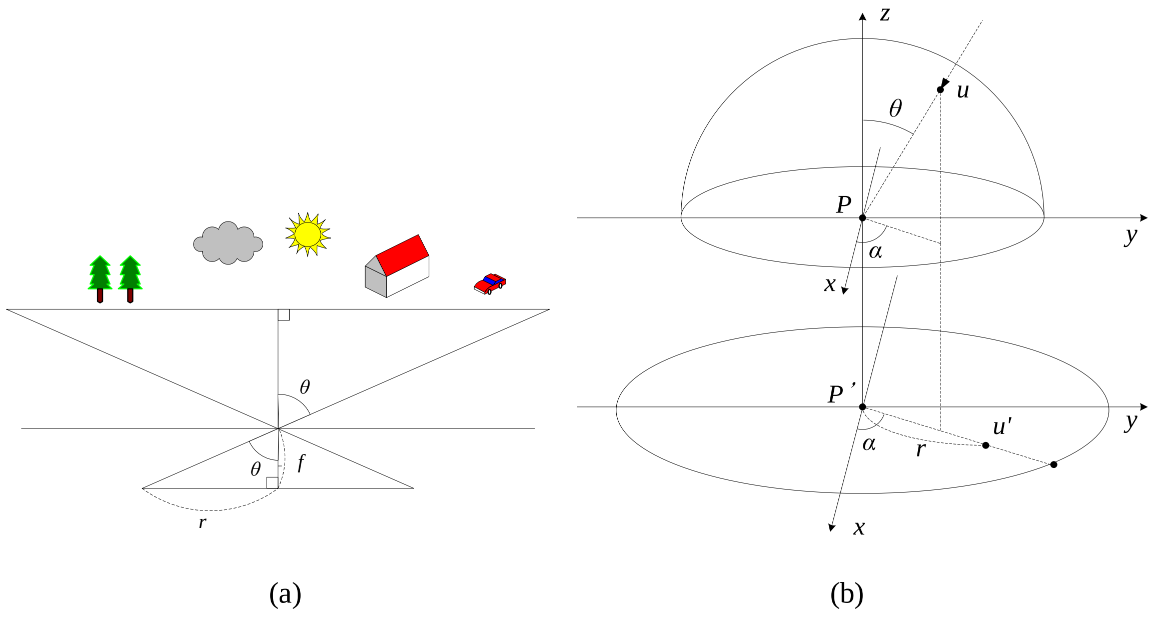


Figure 2.3(a) The pinhole model (perspective projection model) requires an infinite sized canvas ($r = f\ tan90°$) when FOV is 180° ($\theta=90°$). (b) Fish-eye projection model; the bottom is the distorted image and the top is the hemispherical environment in front of a camera with a fish-eye lens.

2.3. 3D shape reconstruction techniques

3D model reconstruction from 2D images is a challenging topic in computer vision and computer graphics communities. There have been many different approaches such as shape from shading [24-32], stereo matching [33-38], shape from texture [39, 40], shape from motion [41-45], etc.

The shape-from-shading approach computes the 3D shape of a surface from the brightness of a single grayscale image of that surface. The basic idea is that the gradient and the distance of an object surface can be calculated by the brightness of that surface area. The conventional shape-from-shading algorithms assume that the light source is infinitely far from an object so that the ray of light comes parallel everywhere on the object surface. Another common assumption of shape-from-shading algorithms is that the object surface is Lambertian. The Lambertian surface [46] diffuses incoming light such that the amount of the diffused light (photon) is proportional to the cosine of the angle between the surface normal and the direction that the light is reflected. Although the amount of photon is decreased proportionally to the cosine of the emission angle, the observed area also increases, resulting in the same increase in the amount of the photon. Consequently, the luminance of the surface is same regardless of the direction from which it is observed. Figure 2.4 shows a diagram explaining the property of Lambertian surface.

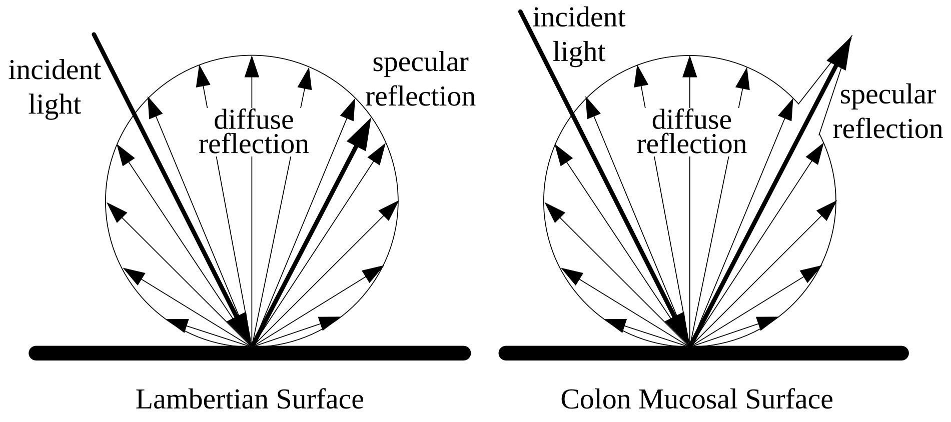


Figure 2.4 Lambertian surface is shown on the left of the diagram; the colon surface reflects much more light to the direction of specular reflection than the Lambertian surface does.

The shape-from-motion approach reconstructs a 3D shape of an object from multiple sequential images. Usually, this approach can be divided into several sub-processes. At first, prominent feature points on an object are selected. Next, the correspondence of these feature points across the image sequence is established and the relative motion between the object and the camera is estimated. Finally, the 3D shape of the object is computed. A critical limitation of the shape-from-motion approach is that a target object must be rigid. In other words, the shape of an object must not change between sequential images.

The stereo-matching approach calculates the depth of an object using disparity in appearance and location of features of a pair of images of the same object. Ideally, the images should be taken by two different static cameras located on a same horizontal line at a fixed distance from each other. To simulate two static cameras using a single moving camera, the camera motion must be predictable to maintain a geometrical relationship such as position and direction of the camera each time an image is taken. Because the movement of the camera is not predictable, stereo-matching is not suitable for reconstruction from endoscopy images.

The shape-from-texture algorithm computes the shape of a surface from shapes of individual patterns on the surface with relatively same patterns. Particularly, disparity refers to the difference in appearance and locations of features.

Stereo-matching techniques are not suitable for endoscopy images because the movement of the camera is not predictable. The shape-from-texture techniques are not suitable for endoscopy images due to inconstant patterns on the colon wall. Shape-from-shading and shape-from-motion techniques are widely used for reconstructing the colon surface from endoscopy images.

Deguchi [19] proposed a 3D colon surface reconstruction technique with the shape-from-shading (SfS) approach. The 3D colon reconstruction using a SfS algorithm has three main challenges which are opposed to the assumption of the conventional SfS algorithms mentioned above. First, an endoscope uses localized and pointed light source instead of distant and parallel light source. Second, the colon surface in the regions with water drop, which reflects more of the incoming light to the specular reflection direction than the other directions, is not Lambertian. Third, the endoscope uses fish-eye lens that does not follow the perspective projection model. The author just assumed that the colon surface is Lambertian to simplify the problem. Then, they developed their own procedure to correct distortion caused by fish-eye lens to follow perspective projection and extended the conventional SfS algorithm to cover the case of near point light source and perspective projection. The author used the fact that the light source is located almost at the same place with the camera lens, which means that the gray level of a certain pixel depends on the distance from the view point as well as the gradient of the object surface. The author described the image irradiance equation as

$$E = \sigma \frac{F(\cos\theta)}{r^2}, \tag{2.1}$$

where E is image brightness and σ is a constant coefficient of imaging system parameters together with the intensity of the light. F is a bi-directional reflectance distribution function. The angle between the surface normal and optical axis, θ, and the distance between the light source (same with the camera) and the observed surface, r, are variables. Since there are two variables in the equation we can solve it only when one of the two variables is given. To solve this problem, the author proposed an algorithm to reconstruct the object shape by repeatedly computing equal-distance contours from a given initial contour at the distance of r from the camera. Then, we can solve Equation (2.1) to get the gradient (θ) for each point on the contour. As a result, we can derive a normal vector of each surface patch for each contour point, which means that we can obtain the shape of the object along the contour. Next, the subsequent equal-distance contour from the camera is computed as follows. Based on the gradient of each surface patch, we find points at a certain distance away from the center of each surface patch such that these points are at equal distance from the camera. Connection of those calculated points is the evolved equal-distance contour for the next iteration. This step is repeated until a number of iterations are reached.

Koppel et al. [17] adopted the shape-from-motion (SfM) technique to reconstruct a 3D colon surface and mapped texture from the input video on the reconstructed model. As mentioned, the SfM algorithms use more than one image so it is critical that the object is rigid or not changing its shape among these images. However, the authors did not address this non-rigid property of the colon. First, as a preprocessing step, they do camera calibration and correct image distortion caused by endoscope's fish-eye lens. Second, corresponding points between images are found using one of the two following schemes. The first scheme is suitable for videos with a high frame rate since it continuously tracks feature points between images. The second scheme employs the Scale-Invariant Feature Transform (SIFT) algorithm to find interesting points and the features of these points [47]. Third, the Random Sample Consensus (RANSAC) [48] algorithm is used to select 5 or 8 best features according to whether the 5-point polynomial algorithm (5P) [49] or 8-point linear algorithm (8L) [50] is used to compute the camera motion parameters (e.g., camera position and angle). The 5P or 8L algorithm calculates the parameters again using the remaining features for validation. After repeated hypothesizing and validation, the best hypothesis is retained, which is the resulting camera motion parameters. Fourth, the colon surface shape can be constructed using a stereo matching technique given the camera motion parameters. Finally, the surface shapes from different sets of images are registered to each other to create a more complete colon surface.

Kaufman et al. [16] combined the SfS algorithm and the SfM algorithm. They reconstruct a partial colon surface using the SfS algorithm proposed by Prados and Faugeras [24] which handles moving local light and light attenuation. Then, with SfM algorithm, the authors estimate the camera motion parameters using features on the partial 3D surfaces created by the SfS algorithm. Using these calculated camera parameters, the authors register these partial 3D surfaces to create a more complete colon surface.

In Zhou et al. [15], an *optical-flow-based* method performs optical flow analysis to align neighboring 3D circles (not necessarily corresponding to colon folds) and determine the distance among them. The technique can generate a small 3D colon segment, but has the following limitations. First, some colon segments (e.g., transverse colon) may not be circular as assumed. Second, the technique assumes that the neighboring circles are not occluded; however, partial occlusion is typical in practice. Third, it does not model colon fold thickness. Therefore, the mucosa behind thick folds may not be simulated.

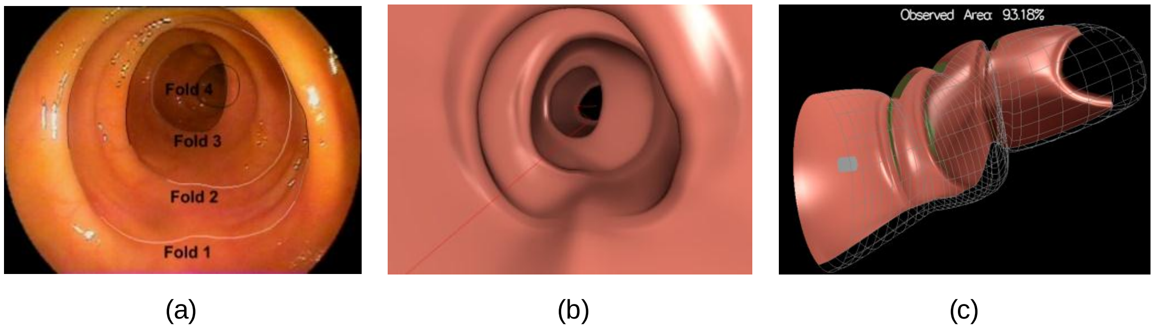


Figure 2.5 (a) Input image with four colon fold contours manually labeled; brightness of each contour pixel represents the intensity change along the norm direction of the pixel. (b) Front view of the produced virtual colon; (c) Same virtual colon with one side of the colon mucosa rendered and unseen areas in green along with the percentage of colon mucosa seen after a simple flying through the virtual colon from the first fold to the last fold without lateral tip deflection.

In our previous work, we proposed an algorithm to reconstruct a 3D virtual colon segment from a single colonoscopic image given manually drawn contours of colon folds [51]. Figure 2.5(b-c) shows the front view and side view of the virtual colon reconstructed from Figure 2.5(a). For each fold, we (i) reverse project the location of each pixel on the manually drawn contour of the fold to a unit vector in 3D space while correcting distortion due to the fish-eye lens used in endoscope cameras and (ii) estimate the depth---distance of the fold from the camera, the fold diameter, the orientation of the plane the fold is laid on, and the fold thickness. Finally, the reconstructed folds in 3D space are connected and the colon wall is generated. Assume that folds in a small tubular colon segment are bounded by the same maximum diameter. Thus, a small fold seen in the image is at a farther distance from the camera than a larger fold. Fold thickness estimation is important as thick folds can block the view of small lesions behind them. More inspection efforts are needed for these folds. The drawbacks of this algorithm are as follows. 1. Manual labeling of colon folds is infeasible in practice. 2. The depth estimation and the calculation of the slant of a fold and the plane on which the fold is laid require the assumption that a fold shape is a perfect circle, which is not the case in practice. 3. The fold surface creation has errors due to the use of the fixed fold width. We addressed these drawbacks in the proposed work.

2.4. 3D model registration

3D model registration techniques find correspondence between two given 3D models and give transformation parameters to register together. We register 3D colon segments reconstructed from each colonoscopy video frame. Many existing 3D model registration techniques are available in the literature [52-55]. Iterative Closest Point (ICP) algorithm [56] based techniques need initial guess and iteration to find the best match. Instead, we discover the correspondence of 3D points of models from 2D images by tracking points. Arun et al. [57] and Horn et al. [58] presented a closed-form solution to calculate the transformation parameters that gives the least mean squared error between the two point sets after the transformation when the correspondence of points between two sets are given. However, the solution sometimes gives a reflection instead of a correct rotation matrix when the two point sets have severely different patterns. Umeyama [59] suggested a solution that always produces the best transformation parameters even for those severely corrupted data. We used Umeyama’s solution to find transformation parameters which rotates, scales, and rotates a local colon model to register into the global colon model.

The brief review of Umeyama's work is as follows.

- Problem definition: Given $M_x = \{p_1, p_2, \ldots, p_n\}$ and $M_y = \{q_1, q_2, \ldots, q_n\}$ where M_x and M_y are corresponding point patterns in 3D space, minimize $\frac{1}{n}\left\|M_y - (SRM_x + T)\right\|^2$ where T is a 3x1 translation matrix, R is a 3x3 rotation matrix, and S is a scalar scaling factor.

- Solution:

$$R = USV^T$$

$$S = 1/\sigma_p^{2}\, tr(DS)$$

$$T = \mu_q - SR\mu_p$$

where

$$\mu_p = \frac{1}{n}\sum_{k=1}^{n} p_k$$

$$\mu_q = \frac{1}{n}\sum_{k=1}^{n} q_k$$

$$\sigma_p^2 = \frac{1}{n}\sum_{k=1}^{n}\left\|p_k - \mu_p\right\|^2$$

$$\sigma_q^2 = \frac{1}{n}\sum_{k=1}^{n}\left\|q_k - \mu_q\right\|^2$$

$$\Sigma_{pq} = \frac{1}{n}\sum_{k=1}^{n}(q_k - \mu_q)(p_k - \mu_p)^T$$

UDV^T is a singular value decomposition of Σ_{pq}

$$S = \begin{cases} I & \text{if } det(\Sigma_{pq}) \geq 0 \\ \text{diag}(1, 1, \ldots, 1, -1) & \text{if } det(\Sigma_{pq}) < 0\text{'} \end{cases}$$

where $tr()$ gives the trace of the input matrix and $det()$ gives the determinant of the input matrix.

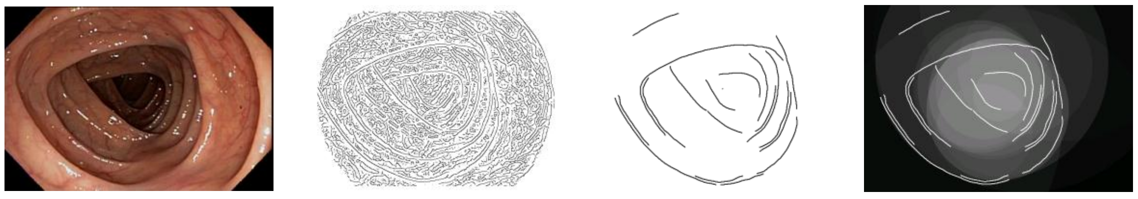
2.5. Conclusion

Most existing works focus on colon surface reconstruction using the SfS or the SfM approach. These existing 3D model reconstruction algorithms are not suitable for tubular colon structure reconstruction. The SfS would wrongly express lumen as relatively far surface not a hole. The colon folds and the hidden area behind folds cannot be expressed. The SfM technique requires that the object be rigid, which is not the case for the colon. The work by Zhou et al. [15] has several limitations as aforementioned.

CHAPTER 3 PREPROCESSING

We propose a new solution for reconstructing the colon structure from one or more sequential colonoscopy images. The challenges are many. The colon and colonoscopy are complex. The colon has many turns and only some colon folds are seen in one image during colonoscopy. Colon folds may be occluded in one frame, but are clearly seen in another frame. There are many types of noise that could be mistaken as colon folds such as blood vessels, stool, polyps, biopsy or therapeutic instruments in a frame. Fish-eye camera lenses cause image distortion. There are varying camera motions such as push/pull and torque motions as well as camera tip movements. Furthermore, there is also automatic adjustment of light distribution by the endoscope video processor, etc. We address several of these challenges.

In this chapter, we present our pre-processing step that detects edges that are likely of colon folds and estimates the center of the innermost colon fold seen in the image. Next, it discards the frame that is not a good candidate for reconstruction, for instance, the frame showing a close-up colon wall. The edges and the center of the innermost colon fold are used in subsequent steps.



(a) (b) (c) (d)

Figure 3.1 (a) Original image; (b) After Canny edge detection of (a); (c) After removal of meandering and short edges of (b); (d) Overlapping of circumcircles of edges in (c)

3.1. Determining candidate fold edges

We first smooth the grey scale image of an input image using a Gaussian kernel and apply Canny edge detection with threshold parameter values that give many continuous edges. Figure 3.1(b) shows an example result. Second, we cut an edge segment with sharp turns into smaller segments and remove the short ones. This process also separates branch edges from the main edges. The pseudo code of the algorithm is shown in Figure 3.2. A result after pre-processing is shown in Figure 3.1(c).

```
pre-processing(C){
  Input: C: Color image; W: Window size in number of pixels
  Output: O: Set of remaining edges
  E = { };
  I = rgb2gray(C);                    // convert an RGB color input image to a grayscale image
  I = smooth(I, $\sigma_{smooth}$);          // smooth image I using a Gaussian kernel
  E = canny(I, cannyLow, cannyHigh); // E is a set of edges as a result of Canny edge detection
  /*  $p_i$: an edge pixel of edge e; $p_i$ and $p_{i+1}$ are neighboring pixels; |e|: the number of pixels of edge e; $d_i$: a
  Freeman chain code [0,…,7] [60] of the direction from $p_i$ to $p_{i+1}$ */
  while (E is not empty) {
    e← removefrom(E); // remove an edge from E; the order of removal is not important
    isSharpTurn← false;   // assume that the edge e has no sharp turns
    for ( i ← 1; $i \leq |e|$ && isSharpTurn ==false; i++) {
      for ( k ← i; k < i + W && isSharpTurn ==false; k++) {
        if ( $d_k$== ($d_i$ + 3)%8 || $d_k$ == ($d_i$ + 4)%8 || $d_k$ == ($d_i$ + 5)%8 ) {
            // Pixel k is in an opposite direction of pixel i. Cut this edge at $p_k$
             e* ←{ $p_j \mid i \leq j \leq k$}; // e* is assigned all edge pixels from $p_j$ to $p_k$ of this edge
             e** ← { $p_j \mid k < j \leq |e|$};      // e** is assigned all edge pixels of this edge after pixel $p_k$
             O ←O ⊔ {e*};                // add e* in the output edge set O
             E ← E ⊔ {e**};              // add the remaining edge into E
             isSharpTurn ← true;
          } // end if
      }   // end for k
      if (isSharpTurn ==false) O←O ⊔ {e};
    } // end for i
  }// end while
  O←removeShortEdges(O, lenThres); // remove an edge in O with fewer pixels than lenThres
}
```

Figure 3.2 Pseudo code of the algorithm to select candidate fold edges

3.2. Estimation of the center of the innermost fold (CIF)

For each edge segment, we draw a circumcircle of a triangle drawn with two tips at the two ends of the edge segment and the last tip at the midpoint of the edge segment. The estimated CIF should be inside the most overlapped regions by these circumcircles because the innermost fold is nested inside the outer folds. However, we may have noise edges that are not part of a fold contour such as blood vessels, stools or operational instruments. We use the following algorithm to find candidate CIF and remove CIF that is likely incorrect due to noise.

Given $I = \{p \mid p$ is a pixel of an input image$\}$ and $E = \{e \mid e$ is a remaining edge after pre-processing$\}$,

1. Find the darkest area D_I in I using the intensity histogram of I, or B_I,

   $B_I = \{b_i \mid b_i$ is the number of pixels of I whose intensity value is i, where $0 \leq i \leq 255\}$.

   $D_I = \{p \mid p$ is a pixel in I assigned to any bin between b_0 and b_k, where k is the smallest value that satisfies $\sum_{i=0}^{k} b_i \geq A \cdot W \cdot H\}$.

   W and H are the width and height of the input image, respectively and A is a constant between 0 and 1 exclusive.

2. Calculate the degree of overlap $v(p)$ for each pixel p using Equation (3.1). The rationale is that we want the value of $v(p)$ to be high when p is within the most overlapped regions created from fold edges.

   $$v(p) = \sum_i l(e_i), e_i \in E_p, \tag{3.1}$$

   where $l(e)$ is the length (number of edge pixels) of an edge e and $E_p = \{e | e$ is an edge whose circumcircle encloses the pixel $p\}$. Instead of counting the number of overlapping circumcircles that contain this pixel p, we sum the length of the edges that form these circumcircles. Fold edges are generally longer than most noise edges. The sum is better than a simple count as it gives a higher weight to long edges.

3. Calculate CIF denoted as p^* by first determining whether an image belongs to Case 1 or Case 2. Case 1 is the case when the average intensity of the darkest area D_I is less than an intensity threshold *IT*; p^* is the center of the darkest area in this case. Otherwise, the image belongs to Case 2 which mostly represents non-dark lumen images or wall images. The precise calculation of CIF is as follows.

$$p^* = \begin{cases} \left(\frac{\sum_i x(p_i)}{|D_I|}, \frac{\sum_i y(p_i)}{|D_I|}\right), p_i \in D_I & \mathit{Case}\ 1, \\ \left(\frac{\sum_i x(p_i)}{|D_{I'}|}, \frac{\sum_i y(p_i)}{|D_{I'}|}\right), p_i \in D_{I'} & \mathit{Case}\ 2, \end{cases}$$

where $|D_I|$ is the number of pixels in the darkest region D_I; $x(p_i)$ and $y(p_i)$ denote the x-coordinate and the y-coordinate of a pixel p_i, respectively. For Case 2, $I' = \{p \mid p$ is a pixel in I with the largest $v(p)\}$.

4. We discard the CIF if the condition below is satisfied.

$$v(p^*) < lenRatio * \sum_i l(e_i) \text{ or } \sum_i l(e_i) < len \text{ where } e_i \in E.$$

We chose smaller *lenRatio* and *len* threshold values for images belonging to Case 1 than those of Case 2. For Case 1, we aim to exclude the darkest area caused by stools, dark blood vessels or shadow of folds from being mis-considered as the lumen surrounded by the innermost fold. For Case 2, we aim to exclude wall images with short edges forming some insignificant overlapping circumcircles.

5. We observe that true CIFs detected from sequential frames are typically near each other except when there is a large camera motion. Based on the Euclidean distance of CIFs in sequential frames, we discard likely false CIFs if the following condition is met.

$$\Delta(p^*_i, p^*_{i-1}) > dist \text{ or } \Delta(p^*_{i-1}, p^*_{i-2}) > dist \text{ or } \Delta(p^*_{i-2}, p^*_{i-3}) > dist,$$

where p^*_i indicates CIF of frame i and $\Delta(p^*_i, p^*_j)$ represents the Euclidean distance between CIFs found in frame *i* and in frame *j*. The values of constants *A*, *IT*, *lenRatio*, *len*, and *dist* were determined experimentally.

3.3. Handling a thick fold

Some folds are thick, resulting in two parallel edges of the same fold. Only the inner edge of the thick fold is desired. We observe that the brightness intensity along the edge-cross-section toward CIF increases (dark to bright) after crossing the edge. To detect an outer edge segment of a fold, we first compute the norm vector for each edge pixel of an edge segment. The direction of the norm vector is 90° of that of the tangential vector of this pixel toward CIF. The vector length depends on the size of the image. Next, we compute the difference as the sum of the intensity values on the norm vector before crossing the edge pixel less the sum of the intensity values on the norm vector after crossing the edge pixel. A negative difference (dark to light) indicates that the edge segment is likely the outer edge of the colon fold, which we discard.

3.4. Conclusion

In this chapter, we present our pre-processing step in detail. The parameter values for the edge detection, *W*, and *lenThres* used in the pseudo code in Figure 3.2 and the vector length used for removing outer edge segments of thick edges were determined experimentally. These values and effectiveness of the pre-processing are reported together with our study in Chapter 7 on evaluation of quality measurements derived from our 3D reconstruction results.

CHAPTER 4 COLON FOLD CONTOUR ESTIMATION USING A SINGLE COLONOSCOPY IMAGE

In this chapter, we describe a new method to detect closed colon fold contours from a single image. We have edge segments and the center of the innermost fold from the pre-processing step. We need to assign edge segments to colon folds, which is very challenging. Given an input image in Figure 4.1(a), consider mapping complex edge segments in Figure 4.1(b) to fold contours in Figure 4.1(c). The solution is rather subjective since there is no precise ground truth. For instance, edge segment e_6 in Figure 4.1 (b) has three candidate pairs in clockwise direction: e_2, e_7, and e_8, any of which is acceptable. Among these possible choices, a good solution should yield a virtual colon segment closest to the real colon segment. Fourth, the colon shape is not always circular. For instance, the transverse colon typically has triangular fold shapes. The colon is not rigid. Hence, the shape can easily be deformed depending on the amount of air/CO_2 used to inflate the colon. Finally, once edge segments are classified for each fold contour, they need to be connected with smooth curves with the least deviation from the original contour shape. This should be achieved even for occluded or flat sections of fold contours.

Many existing image segmentation techniques are available in the literature [61-63]. For instance, level sets [64] and graph cuts [65, 66] methods may be modified for colon fold contour estimation to overcome the aforementioned challenges of noise, occlusion, and deformation of the colon. However, the modification is non-trivial, especially to handle occlusion. These techniques generally take significant time to generate region contours due to examination of pixel properties. These reasons motivated us to explore an alternative.

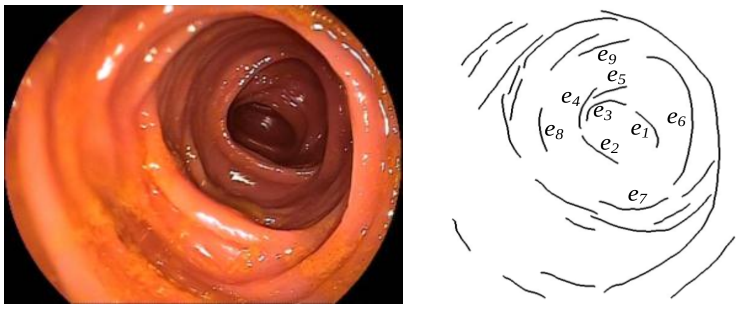
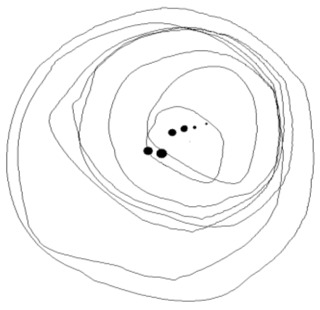


Figure 4.1 The process of fold contour completion. (a) Complex colonoscopic image. (b) Remaining edges after noise filtering. (c) Colon fold contours completed using our proposed technique; e_1 and e_3 are connected and form the innermost fold contour. Edges e_2, e_4 and e_6 are connected and form the second innermost fold contour. Edge e_5 is effectively removed. The small dots are estimated centers of colon folds forming the colon center axis.



4.1. Mapping Edge Segments to Colon Folds

Our problem statement is as follows. Given a set E of edge segments and the seed point s_0, classify the edge segments into subsets, each corresponds to a colon fold. Given E={e_1, e_2, e_3, e_4, e_5} and s_0 which is the CIF location, we group the edge segments in E into subsets by performing region expansion iteratively to estimate each fold region, starting from the innermost fold to the outerrmost fold. Each subset has a 1-1 mapping with a colon fold. Each fold region is bounded by its assigned edge segments and/or association/restriction lines created to represent the missing real edge segments. The already assigned edge segments are then removed from the subsequent consideration. The process is repeated until E becomes empty. Figure 4.2(a-e) illustrates the first iteration to find the region estimating the innermost fold. The region is conceptually expanding from the seed point s_0 in all directions (Figure 4.2(a)). During the expansion, *a tip event* occurs when the region reaches a tip of an edge segment for the first time. Figure 4.2(b) shows a tip event at p_1. At each tip event, we find the closest eligible tip to connect and check whether the two tips can be connected smoothly. Eligible tips must be visible (not blocked by any edge segments or restriction lines) from s_0 and in an opposite

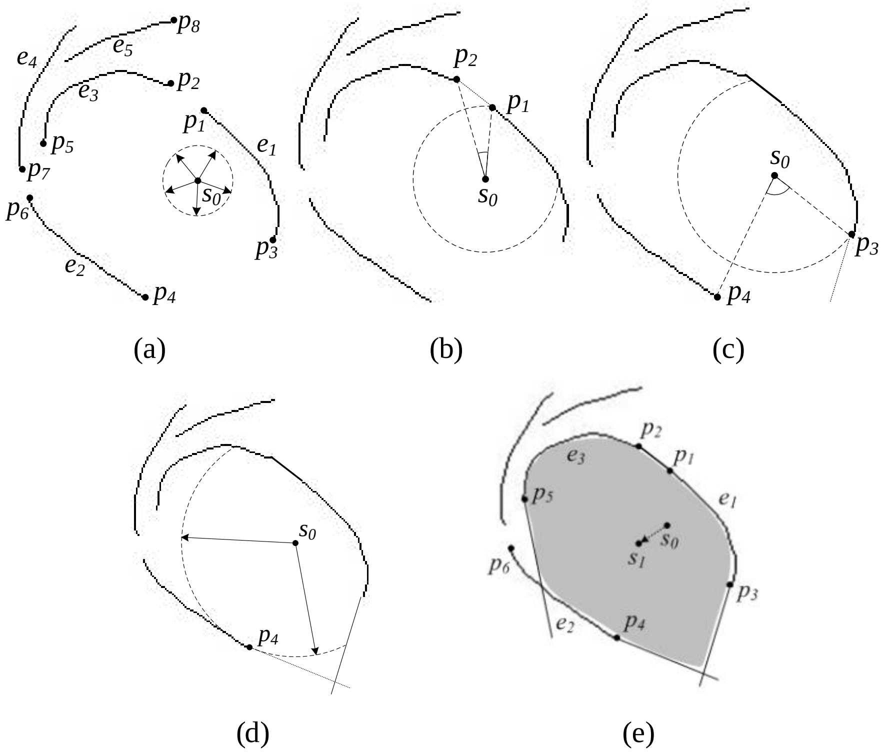


Figure 4.2 (a-e) Illustration of how to find edge segments of the innermost contour using the seed point s_0, which results in e_1 and e_3 assigned to the innermost contour and removed from consideration in the next iteration; (f) s_1 is the new estimated colon center of the innermost fold. See text for explanation. Region expansion of the 2nd innermost fold starts with s_1 as the seed.

tip direction of the tip creating the tip event. In Figure 4.2(b), p_2 is the closest eligible tip of p_1. We connect the two tips if a smooth connection between them is possible by checking whether the angle $\angle p_1s_0p_2$, the angle between p_1p_2 and the tangential line of p_1, the angle between p_1p_2 and the tangential line of p_2, each is less than a small threshold. If so, the connection between them is possible by *association*. In Figure 4.2(b), the association is possible. We use a line to represent the association between the two tips and treat edge e_1, the association line, and edge e_3 as one long edge segment. In Figure 4.2(c), the tip event occurs at p_3; however, its closest eligible tip p_4 cannot be associated with p_3. In this case, we draw a tangential half-line from p_3 as shown in Figure 4.2(c). We call this line the *restriction line*. The association or restriction line is created to represent a missing edge segment. Similarly, restriction lines from p_4 and p_5 are drawn as shown in Figure 4(d and e). At the end, the region stops expanding because it is bounded by edge segments and/or association, restriction lines, or image boundary (Figure 4.2(e)). The edge segment of which both tips is on the region boundary (i.e., the associated e_1 and e_3) is assigned to the current colon fold and the new center point (s_1 in Fig. 4(e)) of the bounded region is computed. Notice that edge e_2 is not assigned to the innermost fold because one of its tips, p_6, is not on the region boundary. The assigned edge segments are removed from E and the next iteration of region expansion starts again with the modified E and the new seed point s_1 to detect the next fold contour.

The pseudocode of the technique is shown in *classifyEdgeSegments(E,s)* where E is the set of n edge segments and a seed point s which are the result of the pre-processing step. The output is the set G which is a set of sets of edge segments for each contour i and C is the set of center points of all detected fold contours. The technique calls *RegionExpansion(E, s, c, G, C*) in line 2 with c initialized to one where c is an iteration index. Each edge has two end points and we assign different labels (*clockwise* or *counterclockwise*) to these end points according to the direction with respect to s as a center. Thus, we represent edge $e_i = p_{ic}p_{i\hat{c}}$ where c stands for clockwise and $\hat{c}$ stands for counterclockwise. Classifying end points into two groups by its direction is important since we only want to connect two end points from different directional groups.

RegionExpansion is a recursive function that does region expansion one fold at a time from the innermost fold ($c=1$) to the outermost fold ($c=n$). Each time it assigns edge segments to a specific fold and adds them into the output set for the fold. We use four temporary sets: Q, Q', T, and F_c. The event queue Q keeps visible end points of given edges in increasing order of Euclidean distance from the center of the fold under consideration.

```
classifyEdgeSegments($E$, $s$){
Input:    $E=\{e_1, e_2,\ldots,e_n\}$ is a set of $n$ edge segments and $e_i=p_{ic}p_{i\hat{c}}$.
              $s(s_x, s_y)$: Seed point at the coordinates $s_x$, $s_y$
Output: $G=\{F_1, F_2,\ldots, F_n\}$ where $F_i=\{e_{i1}, e_{i2},\ldots,e_{im}\}$ (representing the $i^{th}$ fold)
              and $e_{ij}=p_{ijc}p_{ij\hat{c}}$ (representing the edges assigned to the $i^{th}$ fold)
              $C=\{s_1, s_2,\ldots, s_n\}$ where $s_i(s_{ix}, s_{iy})$ is the center point of the $i^{th}$ fold.
1.   $G \leftarrow \{\ \}$, $C \leftarrow \{\ \}$ // {} represents an empty set
2.   $(G, C) \leftarrow$ RegionExpansion($E$, $s$, 1, $G$, $C$)
3.   return ($G$, $C$)
}
```

Figure 4.3 Pseudo code for assigning edge segments to colon folds

```
regionExpansion($E$, $s$, $c$, $G$, $C$){
Input:    $E=\{e_1, e_2,\ldots,e_n\}$ is a set of $n$ edge segments and $e_i=p_{ic}p_{i\hat{c}}$.
              $s(s_x, s_y)$: Seed point // the estimated center of the current fold
              $c$: Iteration index
              $G=\{F_i | F_i=\{e_{i1}, e_{i2},\ldots,e_{im}\}$ and $1\leq i\leq c-1\}$   // $G=\{\ \}$ when $c=1$
              $C=\{s_i | s_i$ is the center of the $i^{th}$ fold, $1\leq i\leq c-1\}$   // $C=\{\ \}$ when $c=1$
Output: $G = G \cup \{F_c\}$ where $F_c$ is a set of edge segments assigned to the $c^{th}$ innermost fold.
                $C = C \cup \{s_c\}$ where $s_c$ is the estimated center point of the $c^{th}$ innermost fold.
1.  if $E$ is empty then return ($G$, $C$)
2.  $Q \leftarrow$ {end points of edges visible from $s$ in increasing order of Euclidean distance from $s$}
3.  $Q' \leftarrow \{\ \}$, $T \leftarrow \{\ \}$, $F_c \leftarrow \{\ \}$.
4.  while $Q$ is not empty do{
5.          Fetch the first element $p_{ij}$ from $Q$ . // $p_{ij}$ is the nearest end point of any edge $i$ from $s$
            // $c$ denotes the clockwise end point and $\hat{c}$ denotes the counter clockwise end point
6.          Draw a restriction line of $p_{ij}$. ($j=c$ or $\hat{c}$).
7.          Remove points blocked by the restriction line from $Q$ and $Q'$.
8.          Remove lines containing those points from $T$.
9.          $Q' \leftarrow Q' \cup \{p_{ij}\}$
10.         if there exists $p_{i^*\hat{j}}$ in $Q'$ where  $\angle p_{ij}\, s p_{i^*\hat{j}} < \alpha$ then { // $\alpha$ is a constant; association occurs
11.             Remove $p_{ij}$ and $p_{i^*\hat{j}}$ from $Q'$.
12.             Insert line $p_{ij}p_{i^*\hat{j}}$ into $T$. //connect the two points with a line
13.         }
14. }
            // A this point, a region is closed by real edges and restriction lines or image boundary.
15.   $s_c$ ← Center of the closed region
16.   while $Q'$ is not empty do{
17.       Fetch a point $p_{kj}$ from $Q'$.
18.       $F$ ←{ }          // $F$ is a temporary storage.
19.       $p_{k^*j}$ ←$p_{kj}$
20.     while there exists a line in $T$ that has $p_{k^*j}$ as an end point do{
21.             Let $p_{k^*j}p_{k^{**}j}$ be the line.
22.             $F$← $F$ ∪{$e_{k^*}$} // $e_{k^*} = p_{k^*j}p_{k^*j}$
23.             $p_{k^*j}$ ←$p_{k^{**}j}$    // *Search the chain of association*
24.       }
25.       if there exists $p_{k^*j}$ in $Q'$ then{
26.             $F_c$← $F_c$ ∪ $F$
27.             $F_c$← $F_c$ ∪{$e_{k^*}$} // $e_{k^*} = p_{k^*j}p_{k^*j}$
28.             $E$ ← E - $F_c$
29.       }
30.   }
31.   $G$ ← $G$ ∪ {$F_c$}
32.   $C$ ← $C$ ∪ {$s_c$}
33.   $c$ ← $c$+1
34.   ($G$, $C$) ← RegionExpansion($E$, $s_c$, $c$, $G$, $C$)
35.   return ($G$, $C$)
}
```

Figure 4.4 Pseudo code of the region expansion used in our method for assigning edge segments to colon folds

After initialization, line 5 gets the first event point p_{ij} from Q which is closest to the seed point and draw a line from p_{ij} which is tangential to e_i on p_{ij}. We call this tangential line a restriction line. In Line 7, we update Q and Q' by removing points blocked by the restriction line. This is to prevent other outer edge segments from being considered. In Line 8, we delete associations containing those blocked points from T. Next, we insert p_{ij} into Q' and check if there is a point associable with p_{ij}. If so, we insert the association into *T* and remove the two points from *Q'*. *Q'* is used to store points p_{ij} detected by region expansion and is updated in line 9 after each event point is processed where $1 \leq i \leq n$ and $j = c$ or $\hat{c}$. If there exist p_{ij} and $p_{i*\hat{j}}$ in *Q'* where the angle $\angle p_{ij} s p_{i*\hat{j}}$ is less than α (set to 25° in our implementation), we connect p_{ij} and $p_{i*\hat{j}}$ with a line and consider that e_i and e_{i*} are conceptually merged into one edge segment. See Figure 4.5. In other words, there is an association between these two edges. When an association occurs, the two associated poins are removed from *Q'* and stored in *T*. F_c keeps edge segments that are classified as belonging to the c^{th} innermost fold and is the output of the c^{th} iteration.

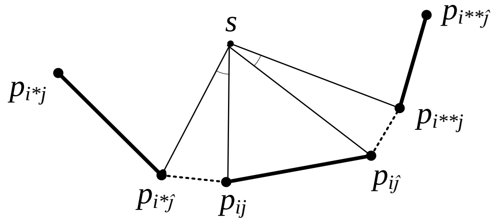


Figure 4.5 Diagram showing how an association is formed among edge segments

We repeat line 4-12 until *Q* is empty. In line 15, we calculate the center of the region defined by edge segments, restriction lines and image boundary and update *s* with the newly calculated center. Now, we start to get the first element of *Q'* and iterate lines 16-19 until *Q'* is empty. We also need to check *T* since *T* contains points that got removed from *Q'* through the association process. When *Q'* is empy, output F_c and s_c and check if *E* is empty. If it is, the procedure returns; otherwise, *c* gets incremented by 1, which means region expansion starts again for the next contour.

4.2. Closing Contours and Assigning Contour Pixel Properties

Let $G = \{F_1, F_2, \ldots, F_n\}$ where F_i is a set of edge segments of fold contour i from the inntermost fold contour ($i=1$) to the outermost contour ($i=n$). $C=\{s_1, s_2, \ldots, s_n\}$ where s_i is the center point of i^{th} fold contour. The problem statement is to estimate closed contours for all real colon folds given sets F and C. We want to complete as many contours as we can out of n contours; however, it is difficult to estimate a real contour shape if the total angle covered by edge segments of the contour is too small. This typically occurs when a fold is flat or a large portion of a fold is hidden behind the other fold. To solve this problem, we assume that the shapes of folds in an image are similar. Under this assumption we use the shape of the nearest completed contour (prefer the outer one than tge inner one if the condition is same) to complete a contour which cannot be closed by itself. Therefore, the algorithm has two rounds: one is to check for the closest outer contour and the other is to check for the closest inner contour. To indicate whether a contour is closed or not, we use $B = \{b_1, \cdots, b_n\}$ where b_i is *true* when contour i is closed; otherwise b_i is *false*. The first round starts estimation from the outermost fold contour F_n (i.e., the last contour found in *classifiedEdgeSegment*) to F_1. In this round, if F_i could not be closed by itself, then use the nearest outer completed contour $i+k$ ($k \geq 1$). The second round starts from the innermost contour 1 to the outermost contour n, using the shape of the nearest inner contour i-k ($k \geq 1$) when we cannot complete contour i using F_i by itself. For drawing the estimated curve to connect given edge segments, we use Bezier curve. The algorithm works as follow.

For each F_i, order its elements e_{ij} by its polar angle and connect $p_{ij\hat{c}}$ and p_{ij+1c} with Bezier curve if $\angle\ p_{ij\hat{c}}\, s p_{ij+1c} < ß$ (In our implementation $ß = 120°$) where j is the index of ordered edge segments. After that we update F_i with the result of connections and if F_i has one closed contour (an edge segment e_i of which $p_{ic} = p_{i\hat{c}}$) insert F_i into G' which is the set of completed contours. If more than one edge segments remains disconnected in F_i we discard this set. Otherwise we keep trying to connect two end points $p_{1\hat{c}}$ and p_{1c} of an edge e_1. The pseudo code is shown below.

```
completeContours(G, C){
Input.    $G=\{F_1, F_2,..., F_n\}$ where $F_i=\{e_{i1}, e_{i2},...,e_{im}\}$ and $e_{ij}=p_{ijc}p_{ij\hat{c}}$.
                $C=\{s_1, s_2,..., s_n\}$ where $s_i(s_{ix}, s_{iy})$ is the center point of $i^{th}$ fold.
Output. $G'=\{F'_1, F'_2,..., F'_n\}$ which is the set of completed folds.
1.    $B = \{b_i=false$ for all $i$ from $1=n\}$
2.    $G' \leftarrow \{\}$
3.    for ($idx \leftarrow n$; $idx \geq 1$; $idx \leftarrow idx -1$) do{ // first round
              // $F^*$ can be empty if the below $k$ is not found
4.          $F^* \leftarrow F_k$ where $b_k = true$ and $k$ is the smallest value larger than $idx$
5.          $(b_{idx}, F'_{idx}) \leftarrow$ adoptShape($F_{idx}, s_{idx}, F^*, s_k$)
6.          $G' \leftarrow G' \cup \{F'_{idx}\}$
7.    }
9.
10.   for ($idx \leftarrow 1$; $idx \leq n$; $idx \leftarrow idx +1$) do{ // second round
11.         $F^* \leftarrow F_k$ where $b_k = true$ and $k$ is the largest value smaller than $idx$
12.         $(b_{idx}, F'_{idx}) \leftarrow$ adoptShape ($F_{idx}, s_{idx}, F^*, s_k$)
13.         $G' \leftarrow G' \cup \{F'_{idx}\}$
14.   }
15.   return ($G'$)
}
```

Figure 4.6 Pseudo code for generating closed fold contours after edge segments for each fold is determined

We explain the functionality of *adoptShape(F, s, F^*,s^*)* through an example. Figure 4.7 shows e^* is the outer completed contour and e is a portion of the inner contour that we have detected in the previous step. We estimate e' (dash-line portion) from e^* by shrinking the completed contour to match the incomplete contour. The algorithm first translate s^* to s. Therefore, the two contours have the same center s in Figure 4.7. Next, we calculate two ratios: r_1 is the ratio of the length of $p_c\, s$ to the length of p'_c s and r_2 is the ratio of the length of $p_{\hat{c}}$ s to $p'_{\hat{c}}$ s. We interpolate the values of ratios between r_1 and r_2 for each sampling point on e^* and multiply each of the ratios to the length from s to the corresponding sampling point to get the length from s to the corresponding point on e'. The pseudo code of *adoptShape* follows.

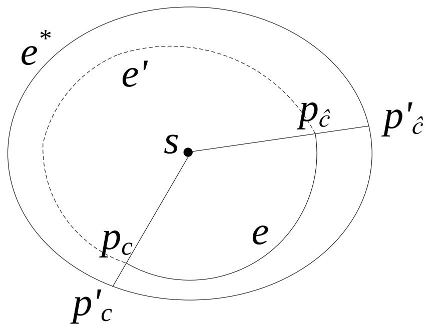


Figure 4.7 Diagram showing the use of the outer contour to estimate the missing portion e of the inner contour.

```
adoptShape(F, s, F*,s*){
Input. $F=\{e_1, e_2,...,e_m\}$ and $e_i=p_{ic}p_{i\hat{c}}$. // F is a set of edges assigned to this fold
            $s(s_x, s_y)$: Center point of the fold $F$.
            $F^*$: Completed fold of which shape is adopted to complete $F$.
            $s^*(s^*_x, s^*_y)$: Center point of fold $F^*$.
Output. $b$: true or false
        $F'$: Completed contour of $F$ if exists otherwise {}
1.  Order $F$ with $e_i$ in increasing order of polar angles
2.  $m \leftarrow |F|$
3.  if $m > 1$ then {                         // try to connect nearby edge segments
4.      for ($i \leftarrow 0$; $i \leq m$; $i \leftarrow i+1$) do{
5.          if $\angle p_{i\hat{c}}\, s\, p_{i+1c} < ß$ then{      // $ß = 120°$ in our implementation
6.              $F \leftarrow F - \{e_i, e_{i+1}\}$
7.              $e_i \leftarrow$ drawBezierCurve($p_{i\hat{c}}, p_{i+1c}, e_i, e_{i+1}$)
8.              $F \leftarrow F \cup \{e_i\}$
9.          }
10.         }
11.         if $F$ is closed then return ($b \leftarrow$ true, $F' \leftarrow F$ )
             // close the remaining open contour
12.  }
13.  $m \leftarrow |F|$ // $\gamma$ and $ß$ are constants
14.  if $m > 1$ then return ($b \leftarrow$ false, $F' \leftarrow$ {})
15.  if $m = 1$ then { // when only a single edge segment $e_1$ exists
                //    Large estimation error, do not close the contour
16.      if $\angle p_{1\hat{c}}\, s\, p_{1c} > \gamma$   then   return ($b \leftarrow$ false, $F' \leftarrow$ {})
              // Medium estimation error, close contour using another contour
17.      if  $ß < \angle p_{1\hat{c}}\, s\, p_{1c} < \gamma$ then {
18.           if $F^* \neq$ {} then return ($b \leftarrow$ {} the$F' \leftarrow$ adoptShape($F, s, F^*, s^*$)
19.           else return ($b \leftarrow$ false, $F' \leftarrow$ {})
20.      }
              // Small estimation error, close contour using local information
21.      if $\angle p_{1\hat{c}}\, s\, p_{1c} < ß$ then{
22.           $F \leftarrow F - \{e_1\}$
23.           $e_i \leftarrow$ drawBezierCurve($p_{1\hat{c}}, p_{1c}, e_1, e_1$)
24.           return ($b \leftarrow$ true, $F' \leftarrow F \cup \{e_1\}$)
25.      }
26. } // end of case for $m$=1
}
```

Figure 4.8 Pseudo code for adopting shape of another complete fold contour to fill the missing portion of a fold

drawBezierCurve($p_{ij}, p_{i^*j}, e_i, e_{i^*}$){

Input: p_{ij}: Start point of Bezier curve to draw.

p_{i^*j}: End point of Bezier curve to draw.

e_i : Edge segment on which p_{ij} is.

e_{i^*}: Edge segment on which p_{i^*j} is.

Output: e': Edge segment connecting e_i and e_{i^*} with Bezier curve.

e' is a closed contour if $e_i = e_{i^*}$.

1. Draw a tangential line l_i of e_i on p_{ij}.

```
2.  Draw a tangential line $l_{i*}$ of $e_{i*}$ on $p_{i*j}$.
3.  $c \leftarrow$ Cross point of $l_i$ and $l_{i*}$.
4.  $a \leftarrow$ angle between $l_i$ and $l_{i*}$.
5.  $r \leftarrow 0.9 \times (a / 180°) + 0.1$; the value of $r$ is obtained based on experiments
6.  $c_i \leftarrow p_{ij} + (c - p_{ij}) \times r$
7.  $c_{i*} \leftarrow p_{i*j} + (c - p_{i*j}) \times r$
8.  $e' \leftarrow$ BezierCurve($p_{ij}$, $c_i$, $c_{i*}$, $p_{i*j}$)
9.  return ($e'$)
}
```

Figure 4.9 Pseudo code for drawing Bezier curve to create a smooth curve between neighboring edge segments

Given closed contours, we now assign a value to each pixel on each contour to represent the fold thickness at that point on the contour. Recall that the contour consists of estimated edge segments and real edge segments. To assign a value for the pixel on the real edge segment, we compute the normal vector of the corresponding edge pixel. The direction of the normal vector is toward the image boundary. We calculate the difference of the maximum brightness intensity value and the minimum brightness intensity value along the normal vector direction considering only pixels within *d* distance pixels from the edge pixel. For the estimated portion of the contour, we interpolate the differences using the differences at the two end points of the estimated edge segments.

4.3. Experimental environment and results

4.3.1. Experimental environment

We implemented our image analysis techniques in OpenCV and used OpenGL for 3D rendering. We evaluated the proposed technique using 39 colonoscopy images selected from seven different colonoscopic procedures and from different segments of the colon (e.g., transverse colon, cecum). We obtained parameters from a separate training set of colonoscopy images. Manual contours of each colon fold were drawn by a trained staff and reviewed by the domain expert for correctness. We count the number of detected folds (*D*), the number of actual folds (*A*), the number of actual folds missed by the program (*M*), the number of wrongly detected folds (*W*), and the number of detected folds that are difficult to evaluate manually (*N*). For instance, the distant inner fold can be much occluded, making it difficult to manually determine the actual shape of the fold. We used the following performance metrics to evaluate the effectiveness of the proposed algorithm:

$$\text{Recall}(R) = \frac{\#\ \text{Correctly detected folds}}{\#\ \text{Actual folds}\ (A)} \times 100,$$

$$\text{Precision}(P) = \frac{\#\ \text{Correctly detected folds}}{\#\ \text{Dectected folds}\ (D) - \#\ \text{N/A}\ (N)} \times 100,$$

$$\text{Similarity}(S) = \frac{\sum_i \frac{MLA_i \cap PLA_i}{MLA_i \cup PLA_i}}{\#\ \text{Correctly detected folds}} \times 100$$

where *MLA*$_i$ and *PLA*$_i$ are manually labeled area and program labeled area for *i*th correctly detected fold, respectively. The area of a fold is the number of pixels inside the fold contour. The test was performed on a workstation with two Intel Xeon E5504 Quad-Core 2GHz Processors, 6 GB RAM, and 64-bit Windows 7 Professional as the operating system.

4.3.2 Experimental results

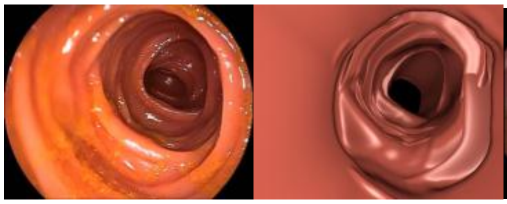
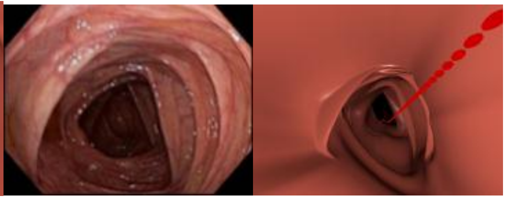
Figure 4.10 shows that the computer generated 3D images using the proposed algorithm resemble corresponding input images quite well. Table 4.1 summarizes the test results showing very good recall score which means that the program detects almost all existing folds. However, some non-existing (wrong) folds were generated, which affect precision. A wrong fold happens when the algorithm pairs

Table 4.1 Performance of our colon contour algorithm; *D*: # detected folds, *A*: # actual folds, *M*: # missed folds, *W*: # wrong folds, *N*: # N/A, *R*: recall, *P*: precision, *S*: similarity

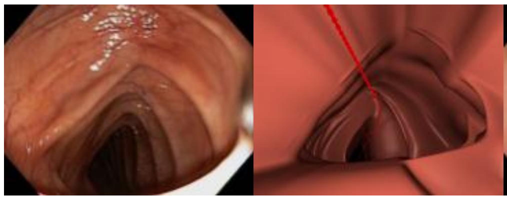
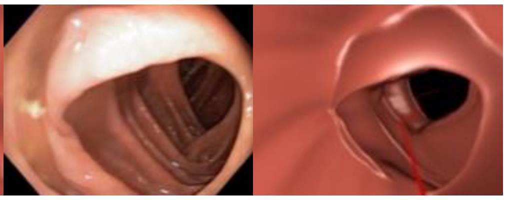
Case ID	#Detect	#Actual	#Miss	#Wrong	#N/A	Recall	Precision	Similarity ±SD (%)
1	5	6	1	0	0	0.83	1.0	87.2±7.2
2	6	8	2	0	0	0.75	1.0	74.5±16.4
3	8	7	0	1	0	1.0	0.88	83.6±3.64
4	6	6	0	0	0	1.0	1.0	70.8±17.7
08404	6	4	0	0	2	1.0	1.0	79.044.8
12239	5	4	0	0	1	1.0	1.0	82.298.3
12660	9	6	0	3	0	1.0	0.67	68.9±8.99
12808	5	5	1	1	0	0.8	0.8	74.0±8.4
14069	4	4	0	0	0	1.0	1.0	72.8±13.8
14279	5	4	0	1	0	1.0	0.8	84.5±5.9
14428	7	5	0	2	0	1.0	0.71	80.3±8.1
14507	5	3	0	1	1	1.0	0.75	83.3±9.3
14537	4	2	0	1	1	1.0	0.67	80.8±13.7
14816	6	6	0	0	0	1.0	1.0	85.9±6.3
14918	4	5	1	0	0	0.8	1.0	85.6±8.6
15007	4	5	1	0	0	0.8	1.0	81.3±9.8
15077	5	5	0	0	0	1.0	1.0	69.1±8.5
15107	4	4	0	0	0	1.0	1.0	70.7±17.2
15137	7	5	0	0	2	1.0	1.0	77.0±16.8
15268	7	5	1	3	0	0.8	0.57	85.1±5.7
15297	8	6	1	2	1	0.83	0.71	77.6±9.0
15807	7	6	0	1	0	1.0	0.86	72.1±10.4
15957	6	4	0	2	0	1.0	0.67	80.3±10.0
16526	5	4	0	0	1	1.0	1.0	80.5±13.7
16646	5	4	0	1	0	1.0	0.8	84.6±5.1
17006	9	6	0	3	0	1.0	0.67	82.9±6.2
17465	6	6	1	1	0	0.83	0.83	78.6±11.8
17615	3	5	2	0	0	0.6	1.0	90.0±4.8
17885	7	4	0	2	1	1.0	0.67	62.9±15.7
18064	6	5	0	0	1	1.0	1.0	85.4±6.5

18274	9	6	0	2	1	1.0	0.75	91.1±4.3
20102	5	4	0	0	1	1.0	1.0	77.6±9.3
20741	8	4	0	4	0	1.0	0.5	73.6±16.6
21018	5	3	0	1	1	1.0	0.75	82.5±3.2
21696	5	5	0	0	0	1.0	1.0	74.9±7.6
22959	6	5	0	0	1	1.0	1.0	84.5±7.6
23326	7	6	0	1	0	1.0	0.86	83.2±11.7
23566	6	5	0	1	0	1.0	0.83	87.7±11.7
24034	4	4	0	0	0	1.0	1.0	75.9±19.8
Overall	229	191	11	34	15	94.2%	84.1%	79.5±13.0

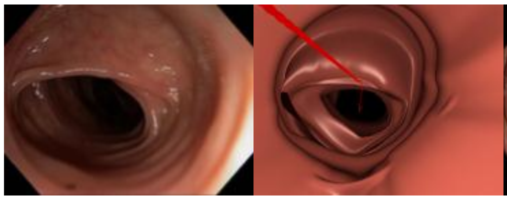
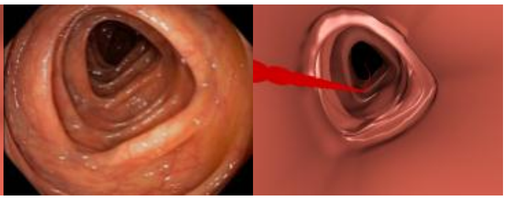
a fold edge segment with an unfiltered strong blood vessel edge or with an edge segment from another fold. If a fold edge segment is not long enough and the true pair edge segment is too far due to a large portion of flat surface, the algorithm mismatches edges from different folds. The crucial factor that reduces similarity is occlusion which happens for inner folds since a large portion of these folds are hidden behind other folds. Our test result shows less similarity for inner folds and higher similarity for outer folds as expected. The processing time for colon fold contour estimation was 0.5 s per image on average.

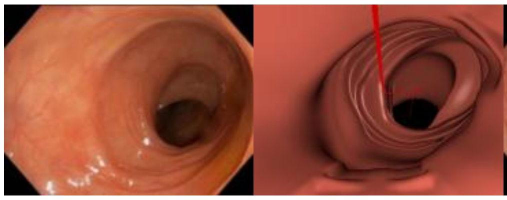
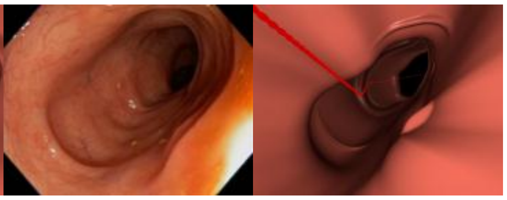
(a) $S = 83.6 \pm 14.4$ (b) $S = 79.0 \pm 4.8$

(c) $S = 82.2 \pm 8.3$ (d) $S = 85.6 \pm 8.6$

(e) $S = 75.9 \pm 7.7$ (f) $S = 83.2 \pm 11.7$

(g) $S = 89.1 \pm 5.3$ (h) $S = 78.3 \pm 11.8$

Figure 4.10 Left image of each item is input colonoscopy image and the right one is its corresponding 3D reconstruction result with the colon center axis (red line).

4.4. Conclusion

We have presented our new algorithm for automatic contour estimation for 3D reconstruction of a colon segment from a single 2D colonoscopy image. The technique gives an average recall of 94.2% and an average precision of 84.1%, which is quite good. The lower performance comes from the fact that the innermost fold may be occluded or the outer fold was behind the camera, causing incorrect estimation of the fold contours in the occluded parts of the fold.

CHAPTER 5 COLON SEGMENT RECONSTRUCTION FROM A SINGLE COLONOSCOPY IMAGE

Automatically detected fold contours of a 2D colonoscopy image are given from the previous chapter. There are three major steps in this chapter: 1) Reverse projection of 2D fold contours into 3D space; 2) Distance estimation of the fold contours in 3D space based on brightness intensity of pixels surrounding the fold contours and 3) Interpolation of the colon fold surface as well as wall surface between neighboring fold contours. We estimated fold width and protrusion in order to emulate real colon structures. Since it is not possible to obtain ground truth real colon structures, we use an Olympus synthetic colon model and measured fold distances and circumferences as ground truth. We conducted experiments on twelve images taken using an endoscope inside the synthetic colon model. We obtained twelve reconstructed virtual colons. The average estimation errors of the fold depth estimates and the fold circumference estimates compared to the ground truth are 4.1 and 12.1 mm, respectively. We also tested that the technique can handle more complex colon images from real colons. The experimental results are promising.

5.1. Reverse Projection

Reverse projection places fold contours in 3D space while correcting distortion caused by a fish-eye lens typically which is used in colonoscopes to allow for a wider field-of-view (FOV). We assume that the intrinsic parameters of the camera are given. That is, we know the projection model that the camera lens of the endoscope obeys. Given the projection model, we trace the ray projected on the image by converting each vertex u on the fold contour to a unit vector v_u in 3D space.

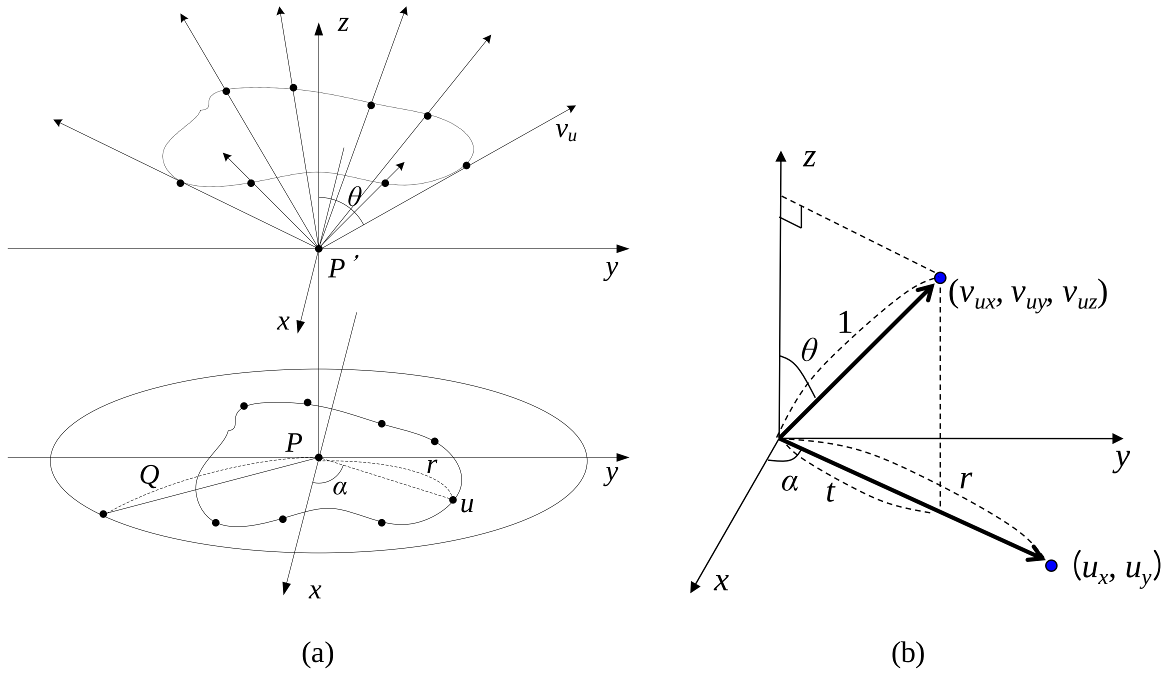


Figure 5.1 (a) and (b) Conversion of point *u* on fold contour to reverse projection vector v_u.

The bottom of Figure 5.1(a) shows a contour in the input image. Let *P* be the origin of the 2D *xy* coordinate system of the input image, which is at the center of the image. Let's consider a contour point *u* at *r* distance from the origin in this 2D coordinate. Let the angle between *x* axis and vector *Pu* be α. Therefore, the *x* and *y* coordinate of *u* (u_x, u_y) is

$$u_x = r \cos \alpha$$

$$u_y = r \sin \alpha.$$

Figure 5.1(b) shows the 3D coordinate system. Let *t* be the orthogonal projection of v_u onto the *xy* plane of this coordinate system. Thus,

$$t = \sin \theta.$$

We can now compute the *x*, *y*, and *z* coordinate of v_u (v_{ux}, v_{uy}, v_{uz}) as

$$v_{ux} = t \cos \alpha = \sin \theta \cos \alpha = u_x \sin \theta / r$$

$$v_{uy} = t \sin \alpha = \sin \theta \sin \alpha = u_y \sin \theta / r$$

$$v_{uz} = \cos \theta$$

We know the values of u_x, u_y, and r. To obtain the 3D coordinate of v_u, we need to compute the value of θ using the given projection model as follows. If the camera follows the equidistance projection and FOV is 140˚,

$$r = f \theta$$

Let Q be the image diameter (see Figure 5.1(a)) which is the maximum value of r. The maximum value of θ is 70˚ since FOV is 140˚. When the value of θ is at the maximum, the value of r is also at the maximum. Thus,

$$Q = 70 f$$

$$f = Q / 70$$

$$\theta = 70r / Q$$

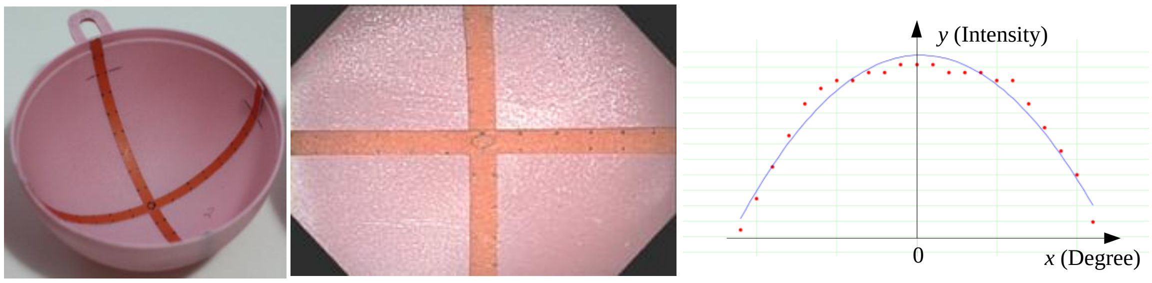
Finally, we have θ; therefore, we can calculate the unit vector v_u (v_{ux}, v_{uy}, v_{uz}) from vertex u (u_x, u_y) on a fold contour of an input image. This process is applied to all the points on each fold contour. At the end of this process, we will have vectors of rays reverse-projecting the 2D contour into 3D space.

5.2. Depth Estimation from Intensity

Our depth estimation consists of two key steps: brightness intensity calibration and estimation of depth of each non-specular pixel based on its brightness intensity.

5.2.1 Brightness intensity calibration

Brightness intensity calibration should be performed before we use pixel intensity to estimate distance of pixels from the endoscope. The rationale is that rays from the light source are more concentrated on the center of the image than the border of the image. We made a calibration object which is a hemisphere with scales inside as shown in Figure 5.2(a). The hemisphere object has a property that when a camera is placed at the center of the sphere, the distance from the camera to any point on the surface is identical. Furthermore, any point on the surface has its surface normal directing toward the camera. Thus, if the light source emits photons evenly around the surface, brightness intensity must be identical for all pixels.



(a) (b) (c)

Figure 5.2 (a) Hemisphere object marked with 18 scales at 10ightness intensity c is used for measuring intensity distribution of the camera. (b) Image of the hemisphere object captured using Olympus PCF-H180AL with 140° FOV. (c) Dots show intensity values sampled along the horizontal line from (b). The purple line was the result of the polynomial regression fitting, which yields Equation (5.1).

We performed experiments by placing several types of endoscope such as Olympus CF-H180AL scope at the center of the sphere of the calibration object. As shown in Figure 5.2(b), pixels near the center (0° in Figure 5.2(c)) are brighter than pixels near the border. We obtained Equation (5.1) that predicts the intensity value y ($0 \leq y \leq 255$) given a degree x using polynomial regression fitting (Lutus, 2011).

$$y = -0.0168x^2 + 186 \tag{5.1}$$

Based on the above experiments, before calculation of pixel depth for each pixel involved in the calculation, we adjust the brightness intensity of the pixel based on the value of x, which is the degree calculated as FoV**d/imgw*. The value of d is the Euclidean distance of the pixel coordinate from the image center and *imgw* represents the image width. The adjusted intensity i' is obtained from the original pixel intensity i using Equation (5.2) where $0 \leq i$ and $i' \leq 255$.

$$i' = i + 0.0168x^2 \tag{5.2}$$

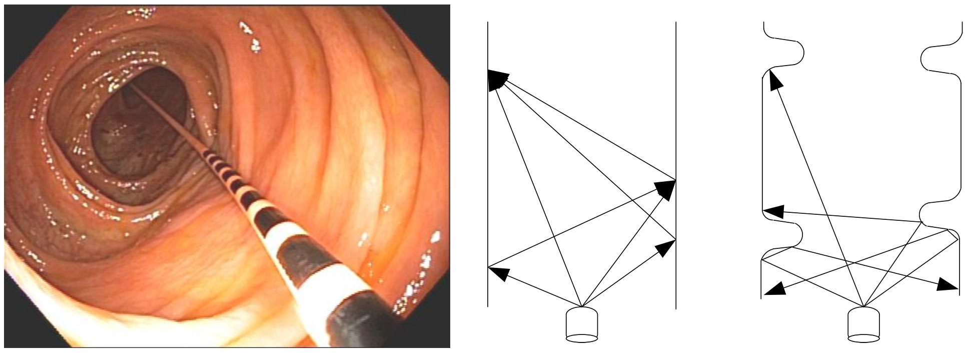
The adjusted brightness intensity is used in subsequent steps. We demonstrate the impact of the brightness intensity calibration on the reconstruction in Section 5.4.1.

5.2.2. Depth from Intensity (*DfI*) of non-specular pixels on a contour

We use brightness intensity of the surface around a fold contour to infer the distance of that surface from the camera. For estimation of depth from brightness intensity, the colon surface is assumed Lambertian except at specular spots. This assumption is also used in previous work for colon surface reconstruction for computer-aided surgery (Kaufman and Wang, 2008). Under the assumptions that (i) a point light source and the camera lens are located at the same position and (ii) there is no ambient light, the Lambertian's cosine law in Equation (5.3) holds.

$$i = \frac{C \cdot cos\theta}{d^2}, \tag{5.3}$$

where i is the intensity value; d is the distance between the camera and the surface; θ is the slant angle between the observer's line of sight and the surface normal; and C is a constant representing an intrinsic camera property. Given the same slant angle, the surface at a further distance appears darker in the image than the surface closer to the camera. For surface at the same distance from the camera, the surface appears darker if it is slant more.



(a) (b) (c)

Figure 5.3 (a) Experiment to establish the relation between the brightness intensity on a surface and the distance from the endoscope to the surface. (b) Colon surface is glossy, which reflects more light to the reflection angle and generates considerable ambient light. (c) Protruded folds blocks ambient light from deep surface.

The assumption (ii), however, does not hold in practice since the closed structure of the colon and glossy mucosa cause significant ambient light. We performed experiments to explore the relationship between the brightness intensity and the distance from the camera in the colon by inserting a special ruler designed for colonoscopy as shown in Figure 5.3(a). The deeper the surface is, the more effect the ambient light becomes as shown in Figure 5.3(b). From our observation, the ambient light causes a slower drop in brightness intensity as the distance d increases than that described by Equation (5.3). We experimented with different values of the power coefficient m (0.5, 0.75, 1, 1.25, and 2) in Equation (5.4). The detailed results are reported in Section 5.4. In summary, we found that $m=1$ gives the least estimation error from the ground truth measurements of an Olympus synthetic colon model.

$$i = \frac{C \cdot cos\theta}{d^m}, \tag{5.4}$$

The limitation of Equation (5.4) is that it does not take the amount of protrusion of colon folds into account. For example, as shown in Figure 5.3(c), tall folds block reflected lights from reaching inside, reducing the amount of ambient light on the surface whereas flat surface should result in more ambient light. For simplicity, we use Equation (5.4) with $m=1$ hereafter.

Next, we estimate the depth (distance from the camera) d_u for every reverse projection vector v_u (in 3D space) corresponding to the pixel u on each of the fold contours. Let θ_u be the angle between the normal vector of the surface where v_u is directing and the line of sight of the camera. Each pixel u has corresponding brightness intensity denoted as i_u. We rearrange Equation (5.4) and replace the parameters d and i with the parameters corresponding to the point u. Hence, we have Equation (5.5).

$$d_u = \frac{C \cos \theta_u}{i_u} . \tag{5.5}$$

To minimize the estimation error of d_u, we want to find a point close to u that best minimizes the slant angle θ_u. When $\theta_u=0°$ (the surface normal at this point faces directly to the line of sight of the camera), $\cos 0°=1$. We cannot use specular pixels since they do not obey the Lambertian's cosine law. Therefore, we find a suitable non-specular pixel close to the point u. According to the Lambertian's cosine law, for points located at the same distance from the camera, those appear brighter in the image have smaller slant angles. Hence, we search for the brightest non-specular pixel within a 1D window of size K starting from u along the normal direction of u away from the estimated center of the innermost fold (CIF). The center is obtained by the preprocessing step (Section 3.2). We denote this brightest non-specular pixel u' and its intensity as $i_{u'}$. We estimate $\theta_{u'}$ instead of θ_{u}. To detect a specular pixel, we slide a square window (of size BxB pixels) centered at each pixel in the image. If the intensity of the center pixel is T times higher than the average brightness intensity in the sliding window, the pixel is considered a specular pixel; $T > 1$. Figure 5.4 illustrates key factors involved in the estimation. The green dots in Figure 5.4(a) are examples of the brightest non-specular pixels along the fold contours.

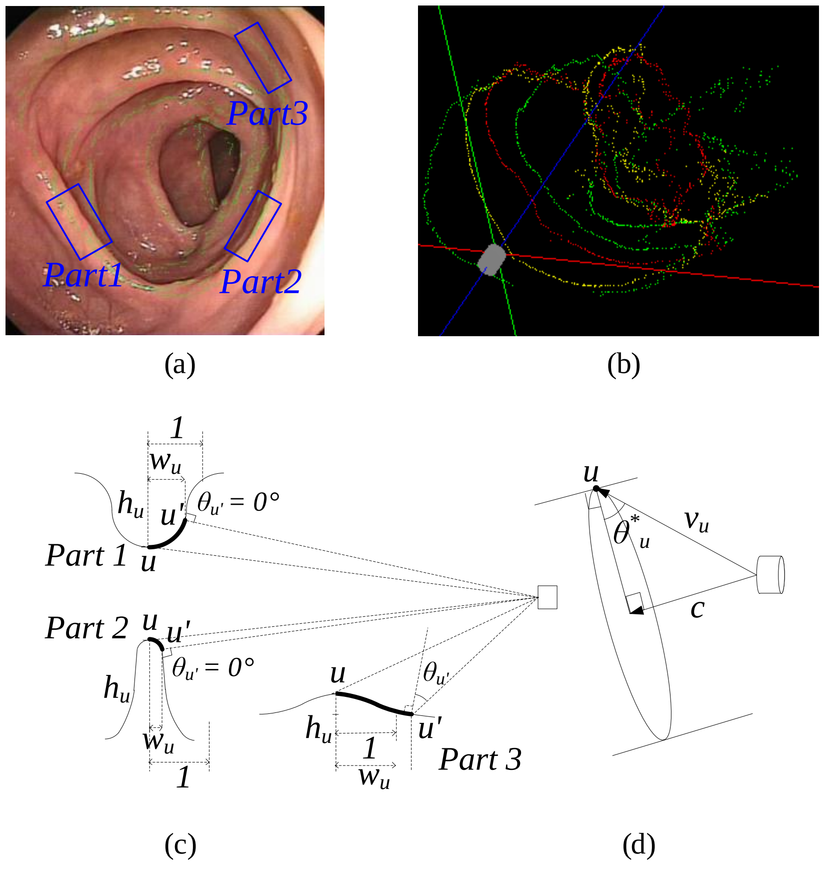


Figure 5.4 (a) Blue rectangles mark different surface properties of a fold. The green dots represent the brightest non-specular pixels (*u'*), within the 1D search window from the fold contours. (b) We estimate the value of $\theta_{u'}$ of *part 1* and *part 2* to be 0° as *u'* faces directly to the camera. However, $\theta_{u'}$ of *part 3* cannot simply be set to zero since it depends on the fold protrusion. (c) For a wide fold like *part 3*, we estimate the value of $\theta_{u'}$ proportionally to the value of θ^*_u. Vector *c* is from the endoscope to the center of the fold contour. Unit vector v_u is the reverse projection of pixel *u*. We know the angle between *c* and v_u, and thus can calculate the value of θ^*_u. (d) Scattered dots are the projections of the green dots in (a) on 3D space by *DfI*. Different color indicates different fold contours.

Let w_u be a normalized distance between the contour pixel *u* and the corresponding brightest non-specular pixel *u'*; $0<w_u\leq 1$. For the point *u* where $0 < w_u < 1$ (like Parts 1 and 2 in Figure 5.4(c)), we set $\theta_{u'}$ to $0°$ since the slant angle of the brightest non-specular pixel *u'* is closest to $0°$. For the point *u* where w_u is 1, the brightest non-specular pixel *u'* is located outside the search range or this part of the fold may be flat (e.g., Part 3 of Figure 5.4(c)). In this case, we need to consider the protrusion of the fold (height) in our depth estimation. Thus, we use Equation (5.6) to represent the two cases.

$$d_u = \frac{C(\lceil \overline{w}_u \rceil \cos 0 + \lfloor w_u \rfloor \cos \theta_{u'})}{i_{u'}}, \tag{5.6}$$

where $\overline{w}_u = 1 - w_u$. When $w_u < 1$, d_u becomes $C/ i_{u'}$. When w_u is 1, we consider the fold height at the point u. This fold height is estimated by h_u as the absolute difference of the average intensities of K pixels inside and K pixels outside of the contour along the perpendicular line across the pixel u. The more protruded fold blocks more rays of light from the light source near the camera. As a result, we see larger brightness difference between the pixels in front of the fold and behind the fold. The value of h_u is normalized between 0 and 1 inclusive.

Let θ^*_u be the slant angle when the fold at the point u is perfectly flat (i.e., h_u is zero). When the fold at the point u is at the maximum height, we set $\theta_{u'}$ is 0° since surface of tall folds faces the camera directly. The value of $\theta_{u'}$ increases as the normalized height decreases. Thus, we have Equation (5.7).

$$\theta_{u'} = \theta^*_u(1 - h_u) \tag{5.7}$$

The angle θ^*_u at the flat surface point u can be estimated using the unit vector v_u and vector c from the camera to the contour center in 3D space (see Figure 5.4(d)). Finally, we have

$$d_u = \frac{C(\lceil \overline{w}_u \rceil + \lfloor w_u \rfloor \cos(\arcsin(v_u \cdot \hat{c})(1 - h_u)))}{i_{u'}}. \tag{5.8}$$

Since we have d_u, we can obtain various points on the reverse-projected fold contours by multiplying v_u and d_u as shown in Figure 5.4(b). These points scatter due to estimation errors of d_u. Moreover, d_u is not available for a point u in an occluded portion of a fold contour. To address these issues, we first remove outlier points using statistical boxplots. We found boxplots to be sufficiently effective for most fold contours, except the innermost contour, which is difficult to estimate correctly due to occlusion. Then, we fit the remaining points to a plane using the least squares fitting algorithm. This plane becomes the plane the fold is laid on. Then, we recalculate v_u by computing the cross point between $\hat{v}_u$ and the plane for each point u. The values of constants C, T, B and K were determined experimentally as explained in Section 5.4.

5.3. Colon Surface Generation

There are two major types of surface: (i) surface of folds and (ii) surface between folds (or wall surface). We use Cubic Bezier curve [67] for interpolation of the surface. In our previous technique [51], we calculate the curves using a fixed fold width, which causes all folds to have same thickness (width), which is not the case for real colon folds. Furthermore, we did not consider smooth continuation of the surface among neighboring folds, which results in unnecessary protrusion in the reconstructed colon as shown in Figure 5.5(c). We propose a new method that uses several features extracted from an input colonoscopy image and geometrical relations between neighboring folds estimated in Section 5.2.2 to improve our previous surface generation technique.

We sample P evenly distributed points from each fold contour and assign indices to these points starting from the topmost point (the one with the biggest y coordinate) in the clockwise direction. The value of P is based on experiments shown in Section 5.4. Interpolation for surface is applied on corresponding points with the same index on different fold contours. Since the same algorithm is applied to each point, we omit the point index from the notations used in the rest of the description.

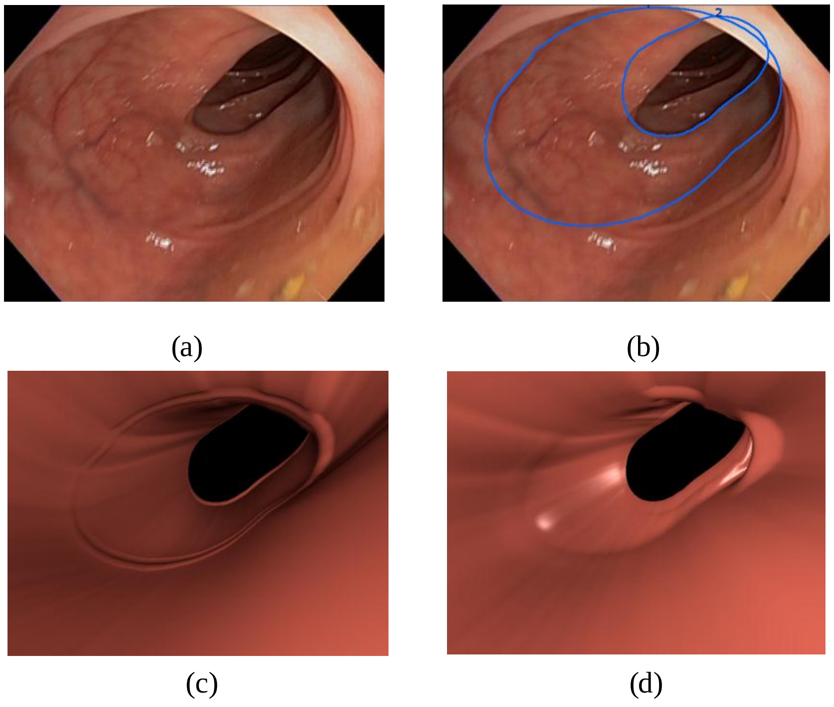


Figure 5.5 (a) Original input image with flat surface on the left side. (b) Original image superimposed with the detected fold contours shown as blue lines. (c) Front view of the reconstructed colon segment using our previous technique shows incorrect protrusion of the fold closest to the camera; however, the original image (a) shows flat surface. (d) Front view of the reconstructed colon segment using the proposed technique with a better expression for the flat part of the fold.

Figure 5.6 shows the fold surface on p_{i0}, one of the sampled points on the i^{th} fold contour. The corresponding surface generated from this point consists of a Bezier curve $p_{i0}p'_{i2}$ on its left and another Bezier curve $p_{i0}p_{i2}$ on its right. Another Bezier curve $p_{i2}p'_{(i+1)2}$ expresses the wall surface, *Wall i*, between the i^{th} fold and $(i+1)^{th}$ fold. The positions of p_{i2} and p'_{i2} are important and should be computed based on the estimated fold width and height in order to generate the surface that closely reflects the real surface. Thus, we use W_i and H_i computed as $W{\cdot}w_u$ and $H{\cdot}h_u$, respectively where w_u and h_u are measured in the previous section. Recall that $0 < w_u \leq 1$ and $0 \leq h_u \leq 1$. The constants W and H are the maximum fold width and fold height determined based on experiments in Section

5.4. Let q_i denote the center of the fold contour i in 3D space. We compute the positions of p_{i2} and p'_{i2} using Equations (5.9-5.13).

$$\overrightarrow{v_{i0}} = \overrightarrow{q_i p_{i0}} \tag{5.9}$$

$$\overrightarrow{v_{i1}} = \overrightarrow{p_{(i-1)0} p_{(i+1)0}} \tag{5.10}$$

$$p_{i1} = p_{i0} + H_i \hat{v}_{i0} \tag{5.11}$$

$$p_{i2} = p_{i1} + W_i \hat{v}_{i1} \tag{5.12}$$

$$p'_{i2} = p_{i1} - W_i \hat{v}_{i1} \tag{5.13}$$

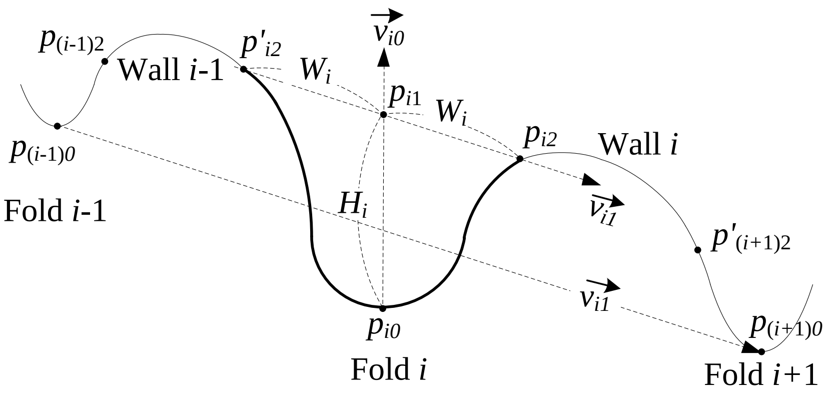


Figure 5.6 Diagram showing an interpolation of the fold surface of fold i.

Equation (5.10) gives the vector that aligns the left and right neighboring folds of the fold i. Figure 5.6 shows $\overrightarrow{v_{i1}}$ twice parallel to each other for ease of understanding of Equations (5.11-5.13). We use Equations (5.11-5.13) to position the points p_{i2} and p'_{i2} on this vector $\overrightarrow{v_{i1}}$ so that the three folds align well and the fold width and height are taken into account. For a fold i which does not have the previous (i-1) or the next (i+1) fold, we use p_i instead of inexistent $p_{(i-1)}$ or $p_{(i+1)}$ to calculate $\overrightarrow{v_{i1}}$.

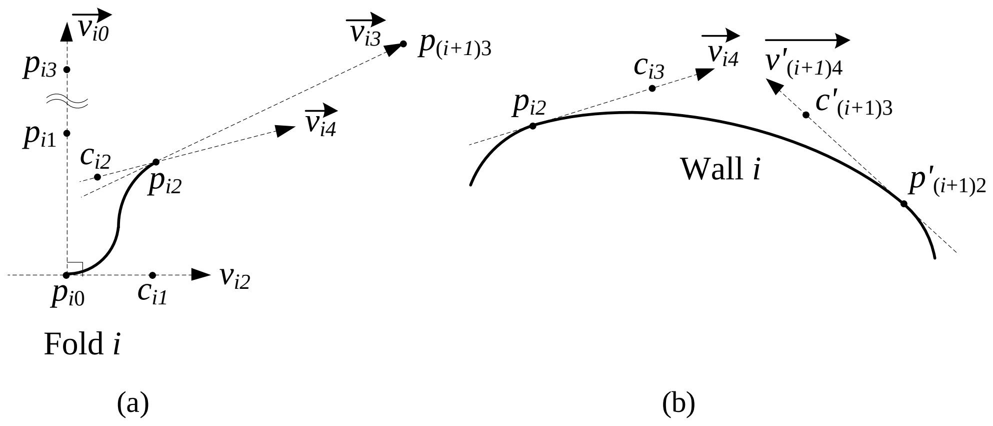


Figure 5.7 Diagram demonstrating the interpolation of the wall surface of wall *i*

Besides the two end points, each Cubic Bezier curve requires two control points to describe the shape of the curve. We need two control points, c_{i1} and c_{i2}, for the curve $p_{i0}p_{i2}$ as shown in Figure 5.7(a). For curve $p_{i2}p'_{(i+1)2}$ connecting the i^{th} and $(i+1)^{th}$ folds, we need two more control points, c_{i3} and $c'_{(i+1)3}$ as shown in Figure 5.7(b). The positions of the control points are also important as they contribute to the shape of the Bezier curve. The control points c_{i1}, c_{i2} and c_{i3} are calculated using Equations (5.14-5.19).

$$c_{i1} = p_{i0} + C_1 W_i \hat{v}_{i2} \tag{5.14}$$

$$p_{i3} = p_{i0} + C_2 H_i \hat{v}_{i0} \tag{5.15}$$

$$\overrightarrow{v_{i3}} = \overrightarrow{p_{i2}p_{(i+1)3}} \tag{5.16}$$

$$\overrightarrow{v_{i4}} = \overrightarrow{v_{i1}}(1 - h_i) + \overrightarrow{v_{i3}}h_i \tag{5.17}$$

$$c_{i2} = p_{i2} - C_3 W_i \hat{v}_{i4} \tag{5.18}$$

$$c_{i3} = p_{i2} + C_4 \left|\overrightarrow{p_{i2}p'_{(i+1)2}}\right| \hat{v}_{i4} \tag{5.19}$$

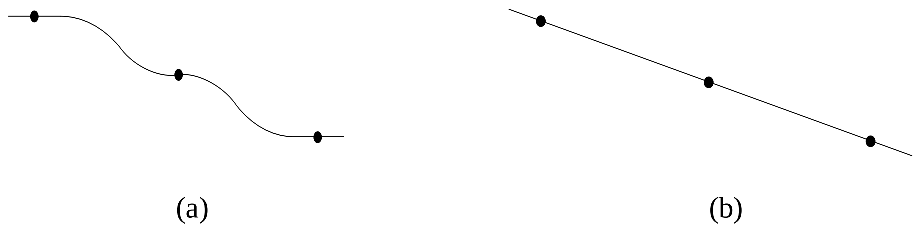
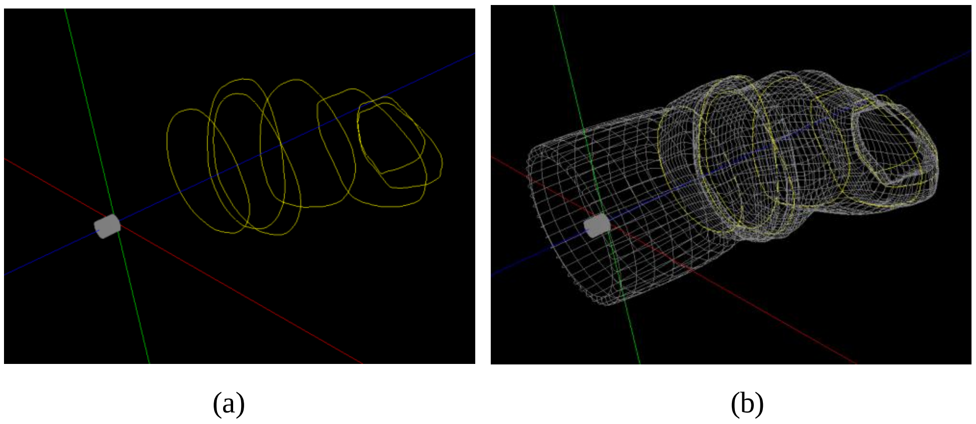


Figure 5.8 (a) Expression for a flat portion of a fold with our old surface generation algorithm; (b) Expression for a flat portion of a fold with proposed new surface generation algorithm. Dots in (a) and (b) represent three consecutive folds.

Let C_1, C_2, C_3, and C_4 be constants. The control point c_{i1} is set to be at a distance proportional to the width of the fold i on $\overrightarrow{v_{i2}}$ ---the vector perpendicular to $\overrightarrow{v_{i0}}$, as shown in Equation (5.14). To set the positions of c_{i2} and c_{i3}, we calculate the position of the point $p_{(i+1)3}$ using Equation (5.15) but for the $(i+1)^{th}$ fold, instead of the i^{th} fold. This point is on the normal vector of the corresponding point on the next fold $p_{(i+1)0}$ at the distance proportional to the height of the fold. The vector $\overrightarrow{v_{i3}}$ is computed using Equation (5.16) to ensure a smooth connection of the wall surface between the two neighboring folds. Control points c_{i2} and c_{i3} are on vector $\overrightarrow{v_{i4}}$ which is linearly interpolated between vector $\overrightarrow{v_{i1}}$ an d vector $\overrightarrow{v_{i3}}$ by the height of fold i using Equation (5.17). The vector $\overrightarrow{v_{i4}}$ of a flatter fold i (h_i is closer to 0), should be closer to $\overrightarrow{v_{i1}}$ than $\overrightarrow{v_{i3}}$ and vice versa. We select $\overrightarrow{v_{i4}}$ this way to avoid unnecessarily protrusion when the fold is flat as shown in Figure 5.8(b) and Figure 5.5(d). Our previous surface generation algorithm resulted in unnecessary bumps as shown in Figure 5.8(a) and Figure 5.5(c). We place c_{i2} using Equation (5.18) on $\overrightarrow{v_{i4}}$ at the distance proportional to the width of the fold. By Equation (5.19), c_{i3} is placed on $\overrightarrow{v_{i4}}$ at the distance proportional to the distance between p_{i2} and $pa_{(i+1)2}$. Control points c_{i2} and c_{i3} on a same vector $\overrightarrow{v_{i4}}$ make the curve at p_{i2} smooth. Control points $c'_{i1,}$ c'_{i2} and c'_{i3} can be obtained similarly. Due to noise within an image and estimation errors, some portion of consecutive folds may conflict with each other. In this case, we adjust the slant of the farther of the conflicting folds as follows. We iteratively increase the depth of the invading portion of the farther contour and re-compute the plane of that contour based on the adjusted depths until the two folds no longer conflict.


(a) (b)

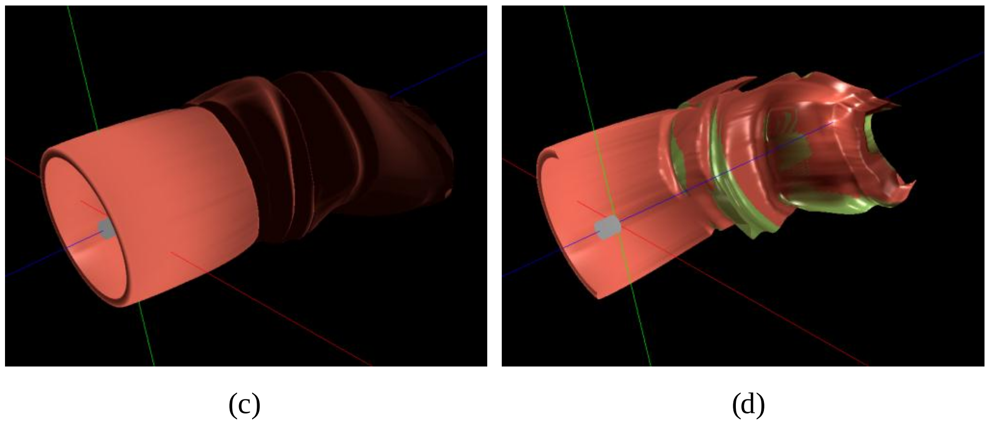
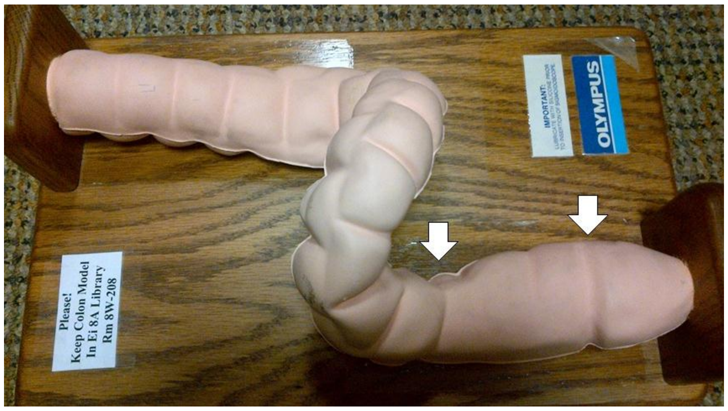
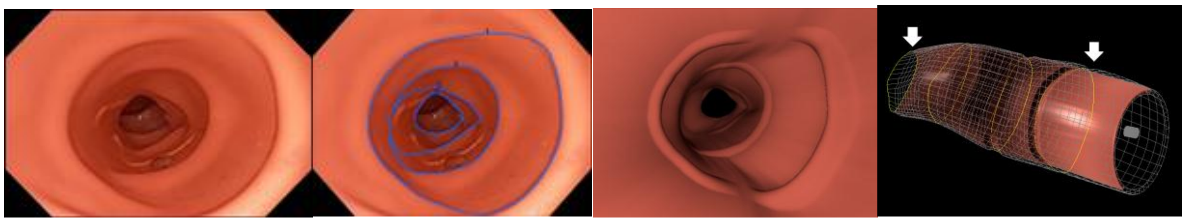

(c) (d)

Figure 5.9 (a) As a result of *DfI* algorithm in Section 5.2.2, fold contours are located in 3D space. (b) Wire-frame of a reconstructed colon segment after surface generation. (c) Final reconstructed colon segment rendered with OpenGL. (d) Half rendered virtual colon segment with unobserved areas shown in green.

Figure 5.9(a) shows an example of reconstructed colon folds in 3D space. After interpolation between these folds to generate the wire-frame of the colon structure is completed (Figure 5.9(b)), we divide the wire-frame into small polygons for smooth rendering using OpenGL (Figure 5.9(c)). We render 40,000 (200 circumferential × 100 stacks × 2 triangles) polygons in between two folds to express smooth colon surface in our experiments.



(a)



(b) (c) (d) (e)

Figure 5.10 (a) Olympus synthetic colon model. (b) Image (06990) taken by an endoscope camera inside the colon model in (a). (c) Image (b) superimposed with labeled fold contours in blue; the contours are numbered from 1 (the outermost contour) to 4 (the innermost contour); (d) Reconstruction result using the fold contours and Equation (5.4) with the brightness intensity calibration. (e) Side view of the half rendered reconstructed colon with the brightness intensity calibration. The section between the two white arrows in (e) correspond to the section of the colon model between the two white arrows in (a). Overall shape of the two sections is similar and the left parts of these sections are narrower than the right parts.

5.4. Experimental Environment and Results

We conducted quantitative evaluation of our technique using an Olympus synthetic colon model (Figure 5.10(a)) and computed average deviations of fold depths and fold circumferences of the reconstructed models from the synthetic colon model. For qualitative evaluation, we selected thirty-eight images from different segments of the real colon from six screening colonoscopy procedures to show that our technique works on real colonoscopy images as well. The input image resolution was 720x480 pixels. For displaying the virtual colon, we used the OpenGL default projection model (perspective projection) because OpenGL does not natively support the equidistance projection used in fish-eye lenses. Thus, a slight distortion and different depth perception is noticeable for some images.

Table 5.1 shows parameter values used in our implementation. The values were determined based on experiments using a set of images not included in the image set for the evaluation in Section 5.4.1 and 5.4.2. FoV was set to 110 since images were cropped and the actual coverage was found to be 110 as observed during our experiments to determine the formula for brightness intensity calibration.

Table 5.1 Parameters and default values

Step	Parameter and value
Section 5.2.2	C = 150
	K = 10
	T = 20
	B =15
	FoV =110
	m =1
Section 5.3.	P =200
	C1 = 1/2
	C2 = 15
	C3 = 1/2
	C4 = 1/5
	W=0.005; H=0.25

5.4.1. Tests on images from an Olympus synthetic colon model

The picture of the synthetic colon model is shown in Figure 5.10(a). The model has three sections: the straight beginning with the narrow end; the curve section in the middle, and the ending straight section. With the model, we measured outside distances between folds and outside fold circumference as measuring the inside circumference of the fold is difficult. We used the information as ground truth for objective evaluation of the reconstruction algorithms. We picked four images from each section of the colon model. We selected these images since the fold contours in these images were detected well so that we can focus on evaluating the reconstruction results.

Table 5.2 Depth represents the distance from the camera. Circf. represents the circumference. MAD is the mean absolute difference (error) between measurements taken from the ground truth and the reconstructed model using the default setting. The lower the MAD, the better the result.

	Ground truth (mm)		Reconstructed model		Absolute Difference	
	Depth	Circf.	Depth	Circf.	Depth	Circf.
Fold 1	28	200	42	202	14	2
Fold 2	65	184	73	184	8	0
Fold 3	108	180	111	190	3	10
Fold 4	128	160	139	129	11	31
MAD					9	11

Performance evaluation on the synthetic colon model

Figure 5.10(a) is the picture of the model. Figure 5.10(b) shows an image taken from the inside of the colon model. The detected contours are automatically labelled and shown in Figure 5.10(c). Figure 5.10(d) is the reconstruction result with the parameter values in Table 5.1. Comparing Figure 5.10(b) and (d), the right part of the outermost fold, fold 1, is shown the generated colon with the highest amount of protrusion as shown in the input image (Figure 5.10(b)). The left part of fold 1 in the virtual colon model is less protruded than the right part of fold 1. The upper and lower parts are flat as in the input image. Table 5.2 shows the ground truth of each fold as well as measurements taken from the virtual colon generated by our algorithm using Figure 5.10(b). We scaled the reconstructed circumferences and depths using the same scale factor. The scale factor was determined by matching the circumference of fold 2 of the virtual colon with the ground truth of fold 2. In this case, we used fold 2 to determine the scale factor since its shape is most similar to the real fold 2. The mean absolute difference (error) of the depth and the circumference are 9 and 11, respectively.

Table 5.3 shows MAD for each image in different sections of the colon model. The average MAD for depth and circumference are computed using MADs of all twelve images. The average depth error is around 4 mm with the minimum depth error of only 1 mm. We observed that the endoscope can see folds within the range of 20-150 mm from the camera. The average depth error of 4 mm is small compared to this range. The average circumference error is around 12 mm, which is less than 1/10 of the circumference of folds in the colon model (140-200 mm).

Table 5.3 MAD in depth and MAD in circumferences of 4 images for each section

	Image ID	Depth MAD	Circf. MAD
Section 1 Straight	06990	9	11
	00080	5	7
	13000	4	12
	07230	5	23
Section 2 Curve	00170	2	4
	00180	1	0
	00270	5	0
	00370	1	30
Section 3 Straight	15010	3	9
	16220	8	16
	15950	4	26
	02870	2	7
Average MAD		4.1	12.1

Determination of suitable value of m in Equation (5.4)

Table 5.4 shows that the value m=1 gives the least error for both depth and circumference while *m*=2 (the original Lambert for both depth ives the highest error.

Table 5.4 Average estimation errors over twelve input images in Table 5.3 using different values of m in Equation (5.4)

Average MAD of	0.5	0.75	1.0	1.25	2.0
Depth	13.3	8.5	4.1	8.2	30
Circf.	29.5	18.8	12.1	14.3	38.2

Effectiveness of brightness intensity calibration

Table 5.5 shows that with intensity calibration, the error is lower for both depth and circumference estimates.

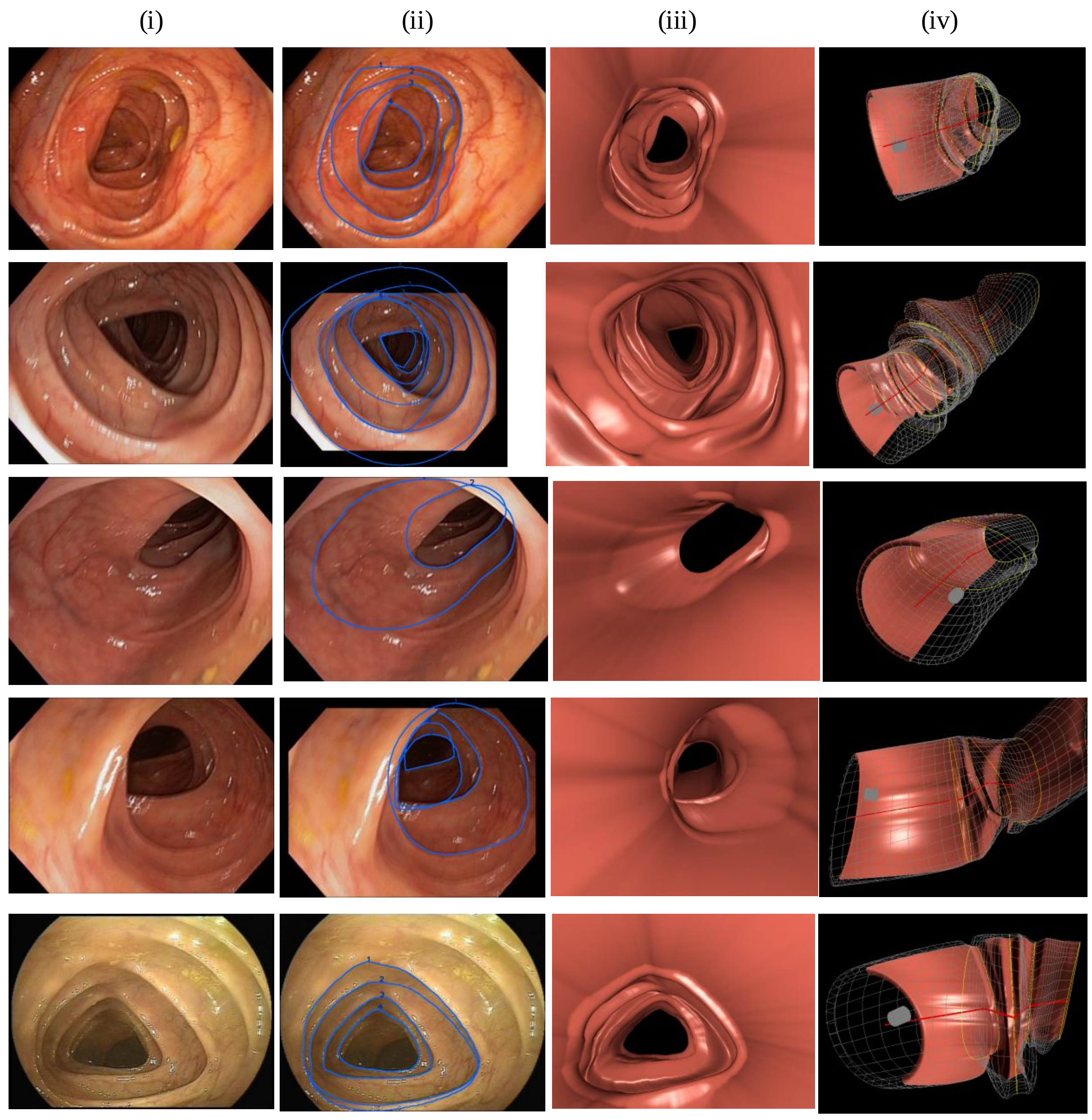
Table 5.5 Effectiveness of intensity calibration on reconstructed results over twelve input images in Table 5.3

Average MAD of	w/ intensity calibration	w/o intensity calibration
Depth	4.1	6.8
Circf.	12.1	15.2

5.4.2. Tests on images of real colons

We evaluated proposed reconstruction technique objectively in terms of overall shape of reconstructed models with the synthetic colon. In this section, we evaluate our technique qualitatively about how well fold shapes are expressed with real colons since we could not measure fold protrusion (height) and thickness (width) inside the synthetic colon. Moreover, we cannot obtain the ground truth shape of real colons even with CT data since the positions of the patient during CT colonography and colonoscopy are different, and the colon is not rigid. However, we can feel the amount of protrusion and thickness of folds when looking at them in 2D images. We can compare how similar protrusion and thickness by looking at the images of the inside of the real colon and the rendered reconstructed models under the same view direction and distance.

Six cases are shown in Figure 5.11. The original input images are shown in the first column of Figure 5.11. The second column shows the corresponding input images overlaid with automatically detected fold contours. In the third column, front views of virtual colons reconstructed by the proposed algorithm are presented. Side views of the half rendered virtual colons are displayed in the fourth column.



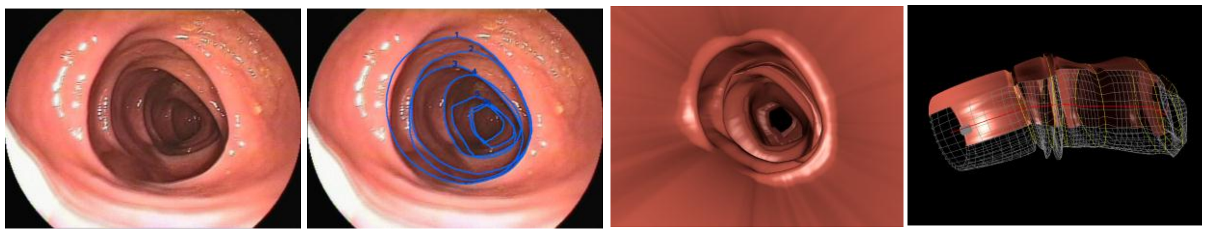


Figure 5.11 (i) Original input image. (ii) Automatically detected fold contours superimposed on the input image. (iii) Front view of the virtual colon. (iv) Side view of the half rendered virtual colon. (a) 16646: rendered fold shape looks closely similar to that in the input image (b) 17885: parts of the folds are beyond the original image; contours of the third and fourth outermost folds were wrong causing the overall shape of the colon to look abnormal as shown in the side view; but the protrusion and thickness of the rendered folds look close to the input image (c) 20102: smooth surface is rendered correctly (d) 21696: the innermost fold is placed too far from the camera (e) 27006: triangular shape folds are rendered correctly (f) 59214: the rendered fold shape and overall shape look similar to that of the input image.

Table 5.6 Execution time per module

Step	Execution time (in seconds)
Depth estimation	0.008
Surface generation	0.016
Polygon construction	0.075
Unobserved area calculation	0.128
Total	0.227

5.4.3. Average execution time per image

We ran our reconstruction software given the fold contours on a 64-bit Windows 7 workstation with Intel Xeon E5504 CPU (two 2.0 GHz processors) and 6 GB of RAM. Table 5.6 shows the time taken for each key step of the colon reconstruction techniques. The calculation of the unobserved area and the polygon construction is the most time consuming.

5.4.4. Discussion

We have shown the effectiveness of our intensity calibration in lowering the error of fold depth and circumference. Our modification of the Lambertian cosine law is effective. Our method for estimating fold protrusion and thickness is promising since it expresses flat surface and protruded surface effectively for most images we observed in our data set. The quality of the 3D reconstruction result largely depends on the result of contour estimation. Good contour estimation gives good 3D reconstruction results with low depth and circumference estimation errors as demonstrated in Tables 5.2 and 5.3. We observed more errors for folds far from the camera because of occlusion. The 3D reconstruction technique itself is promising, but needs further improvement to overcome the aforementioned drawbacks. In the next chapter, we improve our contour estimation technique by tracking fold edges between consecutive frames. Using multiple images is advantageous in contour completion since partially occluded folds in one frame may appear unconcluded in another frame.

5.5. Conclusion

In this chapter, we present our new depth from intensity 3D reconstruction technique of a colon structure. We use intensity calibration to handle uneven distribution of light rays on colon surface. We found that ambient light affects the relationship between brightness intensity and distance of colon surface from the camera described by the Lambertian cosine law. We proposed a modification of the Lambertian cosine law and showed the effectiveness of our modification. The technique is the first fully automated technique that is able to reconstruct a colon structure from a 2D colonoscopy image. Most other existing work focuses on reconstructing the colon surface. The proposed 3D reconstruction technique is promising as shown in our experimental results using measurements of an Olympus synthetic colon model as ground truth. The average depth estimation error and circumference estimation error are 4.1 mm and 12.1 mm, respectively. Our reconstructed results on images of real colons are quite good in terms of expressing fold protrusion and fold thickness based on our observation.

CHAPTER 6 3D COLON SEGMENT AND COLONOSCOPE MOTION RECONSTRUCTION FROM SEQUENTIAL COLONOSCOPY IMAGES

This chapter introduces a technique that reconstructs the motion of a colonoscope as well as a 3D virtual colon segment using sequential colonoscopy images. The first step is tracking fold edges across the sequential input colonoscopy images. In the second step, we build a model of a 3D colon segment for each input frame in a local coordinate system and register the local model into the reference /global coordinate system using the tracking information in the first step. As a result of the registration, we have an extended 3D colon segment. The origins of the local coordinate systems in the global coordinate system form the trajectory of the colonoscope. The third step is colon surface generation and rendering of the model for visualization. This step can be done when visual feedback is required. We complete fold contours by pairing fold edge segments and / or adopting the fold shape from the neighboring fold edges. We use the same algorithm for surface generation as in Section 5.3 after the all closed fold contours are obtained.

6.1. Fold Edge Tracking

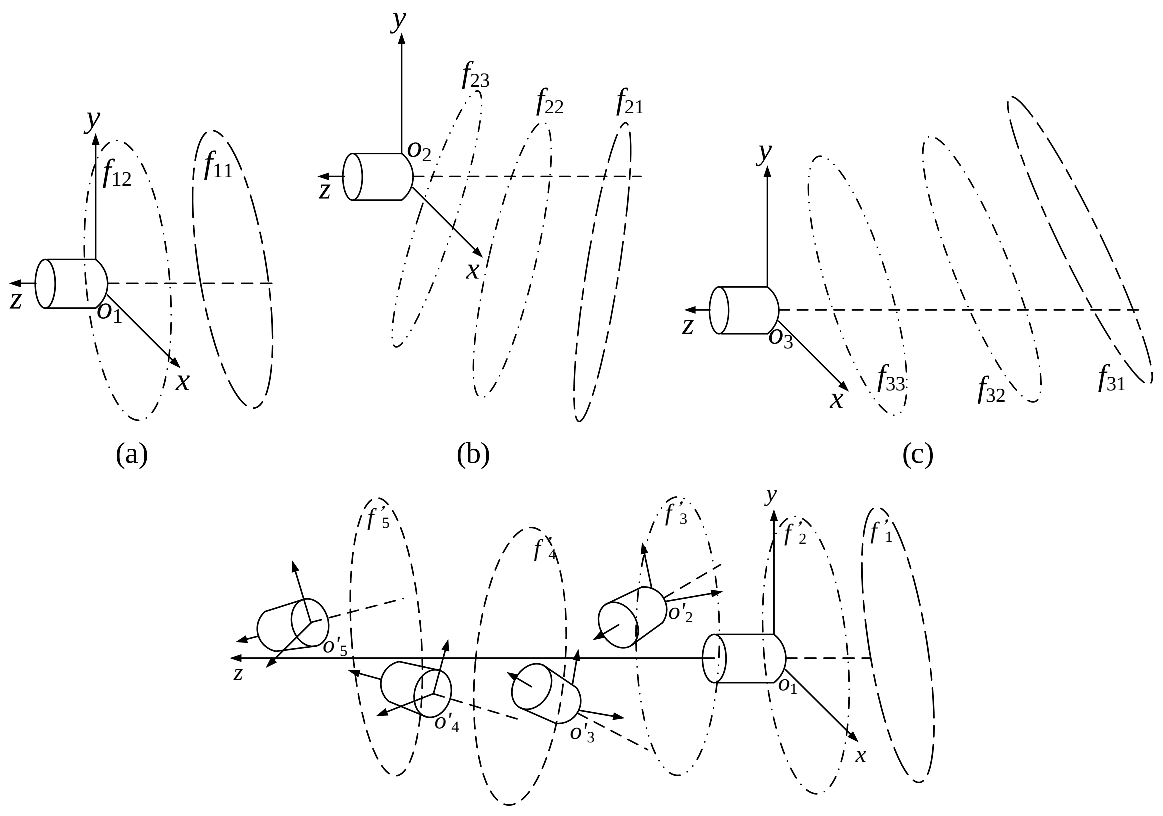
We present our technique to reconstruct a 3D colon segment from a single colonoscopy image in Chapter 5. To establish correspondence among 3D colon segments reconstructed from different images, we find correspondence across fold edges among these images since the models are reconstructed based on the fold edges.

To track a fold edge across frames, we sample a number of points along each fold edge of the current frame and find the corresponding tracked points in the next frame using the Lucas-Kanade (LK) tracking algorithm [68]. Other optical-flow tracking algorithms that do not incur significant processing overhead may be used. A good feature point to track for LK tracking is a corner point with strong derivatives in two orthogonal directions. Since our feature points are along the edges, they are not necessary the best feature points for LK tracking. Furthermore, the corresponding tracked points in the next frame may not locate along any fold edge of the frame. Therefore, we select new feature points to track every frame and use them to locate corresponding points (tracked points) in the next frame only. In other words, feature points in frame *i* are used to find tracked points in frame *i+1*; after that new feature points on the fold edges of frame *i+1* are selected and used for tracking with frame *i+2*, and so on. To select feature points of an edge, we sample them at a fixed number of pixels apart starting from the first pixel of the edge. An alternative to this simple technique is to find feature points closest to the tracked points and use them for tracking instead of resampling new feature points every frame. However, this alternative incurs more computation overhead and is sensitive to the initial selection of good feature points to track.

We use the tracking results to assign a global edge ID to each edge, depending on whether the edge in the current frame corresponds to an edge in its previous frame as follows. We first label each tracked point and its 8 connected neighbors in frame *i* with the global edge ID of the corresponding edge in frame *i*-1 (i.e., global edge ID of the edge used to find these tracked points). If any edge pixel in frame *i* is in the same pixel location as one of the labels, the edge pixel is assigned that label. An edge is either assigned the lowest value (i.e., lowest global edge ID) among the labels assigned to its edge pixels or is given a new unique global edge ID if its edge pixels are not assigned any labels (i.e., not overlapping with any edges in the previous frame).

6.2. Model Registration

We have one 3D colon segment reconstructed from one input colonoscopy image in its own local coordinate system. A colon model is a set of points reconstructed based on fold contours. From fold edge tracking, we also have correspondence among points between neighboring frames. In other words, we have multiple models and we know the correspondence among the parts of the models.



(d)

Figure 6.1 (a-c) Three virtual colon models, each in its own local coordinate system; small cylinder represents the camera location; the camera is withdrawn; (d) Global virtual colon on a global coordinate system generated from merging the virtual colons in (a)-(c); the origin o_1 is used as the origin of the global coordinate system, the origin o_2 in (b) becomes o_2' and the origin o_3 in (c) becomes o3' in the global coordinate system; folds f_{11}, f_{21} and f_{31} in the three local coordinate systems correspond to the same global fold f'_1 in the global coordinate system. Folds f_{12}, f_{22} and f_{32} in the local coordinate systems correspond to the same global fold f'_2. Folds f_{23} and f_{33} in the local coordinate systems correspond to the same global fold f'_3. Folds f'_3, f'_4, and f'_5 are reconstructed from subsequent images not shown here. The camera motions of the colonoscope are obtained.

Thus, we can superimpose all the models in the same global coordinate system to create a longer virtual colon segment as well as the trajectory of the endoscope. This concept is illustrated in Figure 6.1. We call this process registration. A model in a local coordinate system before registration is called a local model and the model registered in a global coordinate system is called a global model.

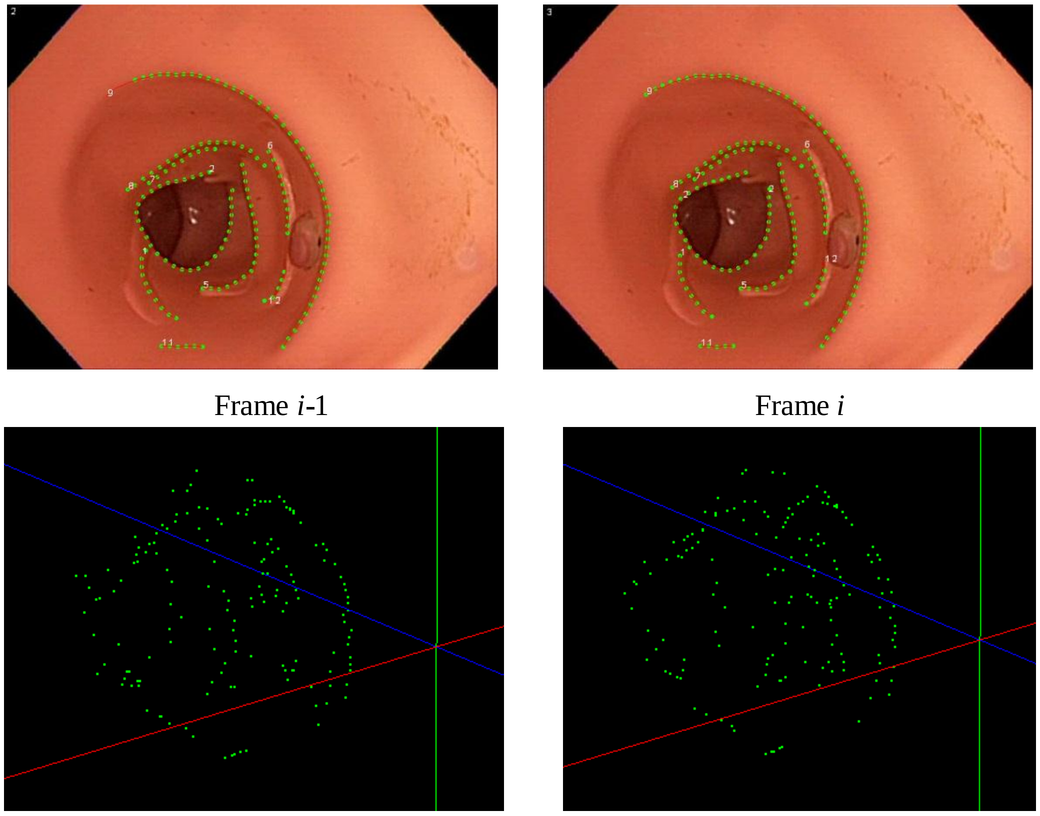
However, the local models do not have identical shape even for the same part of the colon due to noises in input images, estimation errors of our *Depth from Intensity* (*DfI*) algorithm, and colon mobility. In addition, each local model may cover slightly different folds. Furthermore, different illumination among input frames due to a fold blocking the light source causes different scales of local models. We use the least-squares approach by Umeyama [59] to estimate transformation parameters from one local model to another local model such that the transformation error is minimized.

In this section, we first describe a method to build a local model in its own local coordinate system for each input image in Section 6.2.1. Second, in Section 6.2.2, we describe an algorithm that effectively registers local models into a global model in the global coordinate system. The coordinate system of the model derived from the first input frame is used as the global coordinate system. The local model of the subsequent frame is registered with the global model reconstructed from all its prior frames and then merged into the global model. In Section 6.2.3, we derive a formula that calculates camera trajectory with camera view direction. Next, we present the technique for pairing fold edges and completing fold contours in 3D space in Section 6.2.4. Then, we use the same algorithm discussed in Section 5.3 for rendering the virtual colon.

6.2.1. Construction of a Model in a Local Coordinate System

First, we reversely project the points used to track fold edges in 2D space described in Section 6.1

in 3D space using *DfI* and the reverse projection algorithm presented in Sections 5.2 and 5.1, respectively. Thus, for each input frame, we have a local 3D coordinate system where the colonoscope is at the origin and the model is located relatively from the colonoscope. The points are scattered and form a rough shape of a colon segment as shown in Figure 6.2. The pseudo code for building a local model for a given frame is shown in Figure 6.3.



Frame *i*-1 Frame *i*

Model $M_{i\text{-}1}$ built from frame *i*-1 Model M_i built from frame *i*

Figure 6.2 Top row: images from the colonoscope taken inside the Olympus synthetic colon shown in Figure 2.1(b). Green dots show the sample points used for tracking in frame *i*-1 and the corresponding tracked points in frame *i*. Bottom row: corresponding local models. Red, green and blue axes represents *x*, *y* and *z* axes, respectively.

```
buildLocalModel(f){
Input. f: frame
Output. M: Set of 3D points reconstructed from this frame
    $M \leftarrow \{\}$
    for ( $i \leftarrow 1; i \leq$ number of edges in $f$; $i$++) {
        $E_i \leftarrow \{\}$ // $E_i$: Set of points assigned to edge $i$
        for ( $1 \leq j; j \leq$ the number of points in the edge $i$ in $f$ ; $j$++){
            // obtain a reverse projection vector $v_{ij}$ for each $p_{ij}$ (point $j$ on edge $i$)
            $v_{ij} \leftarrow$ reverseProjection($p_{ij}$)
            // obtain the corresponding point in 3D space
            // DfI() is the function that returns the depth using the DfI algorithm
            // unitVector($v$) returns the unit vector of the input vector $v$
            $p'_{ij} \leftarrow$ unitVector($v_{ij}$)* DfI($p_{ij}$)
            $E_i \leftarrow E_i \cup \{p'_{ij}\}$
        }
        $M \leftarrow M \cup E_i$
    }
    return $M$
}
```

Figure 6.3 Pseudo code for building a local model of the input frame

6.2.2. Transformation Matrix Calculation

We use two steps: (1) calculating transformation parameters which registers a local model of frame i (M_i) to its previous local model (M_{i-1}) and (2) computing transformation parameters which registers M_i to the global model by accumulating all the transformation parameters from the first frame up to frame i. In this way, we can obtain the camera motions between consecutive frames all the way up to frame i.

The problem definition for the first step is shown below.

- Given $M_i = \{p_{ij}\}$; $i = 1, 2, \ldots, m$; $j = 1, 2, \ldots, n$; (The model M_i is reconstructed from frame i.)
- Calculate T_i, R_i and S_i to transform M_i with the least mean squared error between the transformed M_i and M_{i-1} where $i > 1$, T_i is a 3x1 translation matrix, R_i is a 3x3 rotation matrix,

and S_i is a scalar scaling factor.

We employ Umeyama's solution[59] to solve this problem. As a result, calculated S_i, R_i and T_i register M_i into the local coordinate system of M_{i-1}. In other words, $S_iR_iM_i + T_i$ is registered in the local coordinate system of M_{i-1} ($i > 1$) and $S_1R_1M_1 + T_1$ is registered in the global coordinate system where $S_1 = 1$, $R_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$, and $T_1 = [0\ 0\ 0]^{\mathrm{T}}$ which represent no scaling, no rotation, and no translation, respectively.

In the second step, we register M_i into the global coordinate system. To transform the local model M_i to $M^G{}_i$ which is a part of the global model in the global coordinate system, we can repeatedly apply the scaling factors, the rotation matrices, and the translation matrices from frame 1 to frame i as shown in Equation (7.1).

$$M^G{}_i = S_1R_1(\cdots (S_{i-2}R_{i-2}(S_{i-1}R_{i-1}(S_iR_iM_i + T_i) + T_{i-1}) + T_{i-2}) \cdots) + T_1 \tag{7.1}$$

$$M^G{}_i = S_1S_2 \cdots S_iR_1R_2 \cdots R_iM_i + S_1S_2 \cdots S_{i-1}R_1R_2 \cdots R_{i-1}\, T_i + \cdots + S_1R_1T_2 + T_1 \tag{7.2}$$

$$M^G{}_i = S^G{}_iR^G{}_iM_i + T^G{}_i, \tag{7.3}$$

$$S^G{}_i = \prod_{k=1}^{i} S_k, \tag{7.4}$$

$$R^G{}_i = \prod_{k=1}^{i} R_k, \tag{7.5}$$

$$T^G{}_i = T^G{}_{i-1} + S^G{}_{i-1}R^G{}_{i-1}T_i, \tag{7.6}$$

Equation (7.3) transforms the local model M_i to $M^G{}_i$ in the global coordinate system, directly.

Note that a different problem formulation is possible. For instance, instead of finding the best transformation matrix between adjacent frames, one can consider a range of previous or future frames to find the best matching model with the least transformation error. This approach should produce better registration results, but incur more processing overhead to process additional frames.

6.2.3. Colonoscope Motion Calculation

Transformation matrices provide camera motions as shown in Equations (7.7-7.9). We obtain the camera position at frame i ($C^L{}_i$) from repeated applications of the translation matrices obtained from registering adjacent frames starting from the first frame to the current frame i. We obtain the camera view vector at frame i ($C^V{}_i$) where the camera is directing and the camera up vector at frame i ($C^U{}_i$) which is perpendicular to the camera view vector. The camera up vector can express camera torque.

$$\text{Camera location } C^L{}_i = T^G{}_i \tag{7.7}$$

$$\text{Camera view vector } C^V{}_i = \prod_{k=1}^{i}(R_i)\, C^V{}_1 \tag{7.8}$$

$$\text{Camera up vector } C^U{}_i = \prod_{k=1}^{i}(R_i)\, C^U{}_1, \tag{7.9}$$

where $C^V{}_1$ = (0, 0, -1) is the initial camera view vector; $C^U{}_1$=(0, 1, 0) is the initial camera up vector; and $C^L{}_1$ = (0, 0, 0) is the initial camera location in the local coordinate system.

6.2.4. Merging $M^G{}_i$ to the global model G

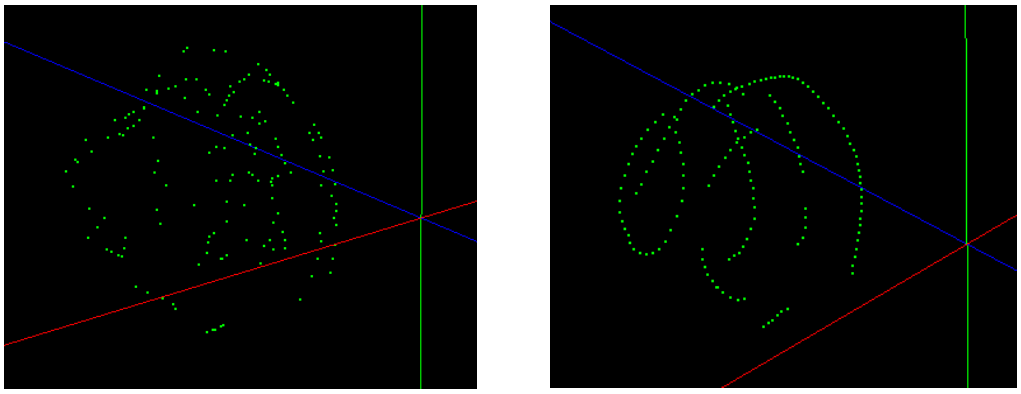
First, we find a representative curve (RC) in 3D space for each edge in a given $M^G{}_i$. Each RC of $M^G{}_i$ is assigned the global edge ID given from the fold edge tracking step. Points on the same edge may be scattered at different depths from the camera as shown in Figure 6.5(a). Therefore, we place these points on the same average depth by repeatedly calculating an average curve from the average curve in a previous iteration. The number of iterations is equal to the number of points on the RC. To compute the average curve of an RC, we start from the first point on the edge to the last point on that edge. For each point, we replace its depth with the average depth of itself and its neighboring point on the same edge. Figure 6.5(b) shows nine RCs obtained from the model $M^G{}_i$ as a result of this step applied to the points in Figure 6.5(a).

Second, we merge the RCs of $M^G{}_i$ with those of the global model G. The pseudo code is shown in

Figure 6.4. As a result of this step, some existing RCs of *G* may become longer and the positions of points on these RCs may be updated.

```
mergeMtoG($M^G_i$, $G$){
Input. $M^G_i$: Local model transformed to global coordinate system
         $G$: Global model
Output. $G$: Global model after merging with $M^G_i$
    For each $RC_j$ of $M^G_i$,
        For each $RC_k$ of $G$ with the same global edge ID as $RC_j$,
            if $RC_k$ is longer than $RC_j$        // the number of points of $RC_k$ is more than that of $RC_j$
                For each point $p_{jl}$ on $RC_j$,
                    Compute the Euclidean distance between $p_{jl}$ and the nearest point $p_{km}$ on $RC_k$
                    if the distance < THRSD_DIST,
                        Update the value of $p_{km}$ to be at the average location of $p_{jl}$ and $p_{km}$ in $RC_k$
            else
                For each point $p_{km}$ on $RC_k$,
                    Compute the Euclidean distance between $p_{km}$ and the nearest point $p_{jl}$ on $RC_j$
                    if the distance < THRSD_DIST,
                        Update the value of $p_{jl}$ to be at the average location of $p_{jl}$ and $p_{km}$ in $RC_j$
                Replace $RC_k$ with $RC_j$ from $G$
    return $G$
}
```

Figure 6.4 Pseudo code for merging M^G_i to the global model *G*



(a) Model M^G_i (b) RCs in Model M^G_i

Figure 6.5 Example of deriving representative curves (RCs)

6.3. Fold Contour Completion for Rendering

The previous steps are performed for every frame to update the global model *G*. However, the fold contour completion and rendering step only run when the endoscopist wants to visualize the global model G with the unobserved area labeled and the motion of the endoscope. In this step, we group RCs of G and connect the grouped RCs to make a closed fold contour. In order to group these RCs, we need to find planes in which these RCs are laid on. The difficulty is that some RCs may be too short or RCs are close to straight. In these cases, existing linear plane fitting techniques could result in incorrect planes for those RCs. We design a technique to find a representative plane (RP) for each RC first and group nearby RPs to find an average RP for the group. To generate a smooth connection between a pair of RCs, we use Cubic Bezier curve when the gap between the two tips of the RCs in the pair is smaller than a threshold. Otherwise, we adopt the shape from the nearest RC covering the gap. If no nearest RC can cover this gap, we do not use this RC to create a fold contour.

1. Filtering noise RCs

- Remove short RCs: We filter short RCs out as short ones are likely mis-detected blood vessel edges or wrong tracking. We compute the sum of the Euclidean distance of the x, y, z locations of the neighboring pairs of points with the same global edge ID. We discard this RC when the sum is smaller than a threshold, THRSD_VALID_RC_LENGTH.

- Discard non-key RCs: This step is to pick one key RC out of closely located RCs and discards non-key RCs. First, we sort RCs in the decreasing order of their length (the sum of the Euclidean distance between the neighboring points). Then, for each RC_i, we find an RC_j (where $j \neq i$) closer to RC_i than a threshold, THRSD_DIST_BTW_RC, and discard this RC_j. To calculate the distance between RC_i and RC_j, for each point in RC_i, we find the nearest point in the RC_j based on the Euclidean distance among them. The average of these distances is the distance between RC_i and RC_j.

2. Grouping the remaining RCs

- To group nearby RCs together, we need distance among RCs. Thus, we find a representative plane (RP) for each RC and the distance among RPs is used for grouping. RP is perpendicular to the vector directing to the center of the innermost fold (CIF) projected in 3D space. Each RP is centered at the mid-point of its RC and has the reverse projection of CIF as its normal vector as shown in Figure 6.6(a). Once we find the corresponding RP of an RC, we project all the points on the RC to the RP along their reverse projection as illustrated in Figure 6.6(b).

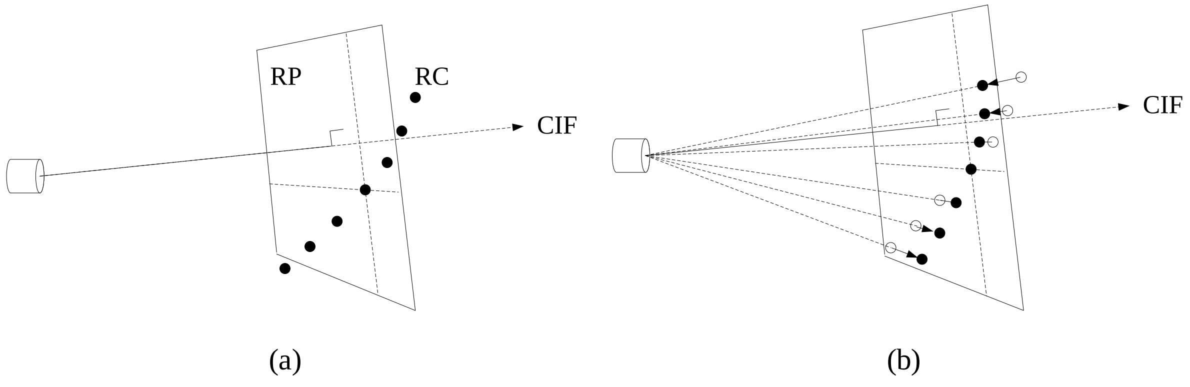


Figure 6.6 Diagram showing a representative plane (RP) of a given RC

- Starting from the RP with the lowest global ID as the current RP, we group RPs whose Euclidean distance from the current RP is within a threshold, THRSD_DIST_BTW_RP. If there is at least two RPs in the group, we compute the average RP among the RPs in the group. Next, we project all the points on the RPs in the group to this average RP along the reverse projection vector of each point. The RPs in the group is excluded from further grouping. We consider the RP with the lowest global ID among the remaining RPs as the current RP and repeat the same process until all RPs are considered.

3. Fold contour completion for each RP

Each RC in an RP has the start tip and the end tip. We determine the center c of the fold as the average location of all the end points of all the RCs in the RP. Choose an RC_i and compute the angle ($\angle e_i c\ s_j$) between the end tip (e_i) of the RC_i, c, and the start tip (s_j) of the $\underline{RC_j}$. We pair RC_j with RC_i which has the smallest $\angle e_i c\ e_j$ using the RC connection technique described below. If there is no RC_j that can be paired with, we use the shape adoption technique described below. We repeat this process until we can no longer pair any remaining RC in this RP.

RC connection: To connect e_i and s_j, we first compute the tangent lines: one for e_i and the other for s_j using the tip and the k^{th} point from that tip where k is determined based on experiments. We select a control point on each tangent line at the distance d from the tip where d = *control_prop* * Euclidean distance between the e_i and s_j. We do not connect s_j of RC_j and e_i of RC_i if the distance between the two control points is greater than the distance between s_j and e_i since, in this case, the two tangential lines do not meet where the connection may not be smooth.

Shape adoption: We calculate the range (polar angle [0, 360)) uncovered by RC_i which is [θ_i, θ_j]

where θ_i is the polar angle of e_i and θ_j is the polar angle of s_j as shown in Figure 6.7. We find RCs in other RPs that overlap with the range $[\theta_i+\alpha, \theta_j-\alpha]$. If more than one RC satisfies the condition, select the RC of the closest RP in the Euclidean distance. Then, project the points on the overlapping segment of the selected RC on this RP on their reverse projection vector. The margin α is to allow a smooth connection between the RC_i and the adopted segment where $\alpha = (\theta_j - \theta_i) * margin\_prop$. Next, connect the adopted segment and the two tips of the RC_i using the RC connection technique. If there is no RC in any other RPs to adopt the shape from or the uncovered polar angle after the shape adoption is still at least the threshold angle, we discard this RP from fold contour completion. The values of the parameters *control_prop* and *margin_prop* and thresholds were determined experimentally and will be described in Section 6.4.

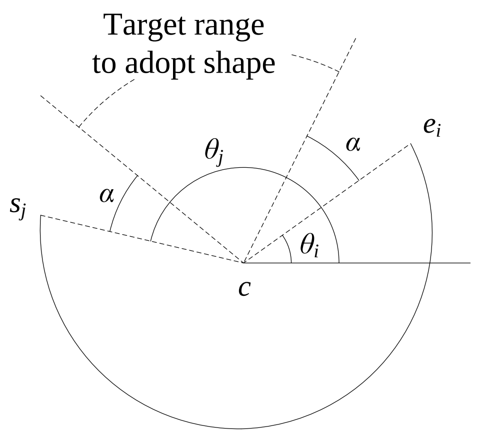


Figure 6.7 Shape adoption; s_j denotes the start tip of RC_j and e_i is the end tip of RC_i

4. Finding the plane for each completed contour

The completed contours are forced on RPs which has different slant from a real plane that the contour is lying on. Thus, we find a natural plane for each completed fold contour using the same method in Section 5.2 except that we use only the points on the real RCs but not the points created from Bezier Curve or the shape adoption technique.

5. Colon surface generation

We generate colon fold and wall surface using the same method in Section 5.3.

6.4. Experimental Setup and Results

We developed our software in C++ language with OpenCV and OpenGL. We evaluated the effectiveness of our technique on the reconstructed camera motion and the reconstructed colon structure. We used the same synthetic colon as in Section 5.4.1 and the same ground truth. The depth of folds and fold circumference ground truth were measured from the outside of the synthetic colon by our domain expert. The domain expert inspected the inside of the synthetic colon by repeatedly performing a full insertion (from the beginning to the end of the synthetic colon) followed by a full withdrawal (from the end to the beginning of the synthetic colon). Note that the measurements taken from the outside of the synthetic colon may be different from those taken from the inside of the synthetic colon during the inspection since the synthetic colon can change its shape depending on the amount of air/CO_2 used during the inspection and endoscope positions. However, the domain expert is not able to measure the circumferences of the colon folds from the inside of the synthetic colon.

Although Figure 6.8 shows that the curvy section of the colon (section 2) has several colon folds, most images of the inside of this section is either wall or blurry or has distorted fold shape due to the endoscope movement into this section. For the following experiments, we selected a number of image sequences and used our technique to reconstruct the corresponding colon segment and the camera motion from these sequences. We used a total of 898 images for evaluation of camera motion direction and used a total of 456 images for evaluation of reconstruction of the colon structure. These images were originally captured at the capturing rate of 29.97 fps at the resolution of 720x480 pixels.

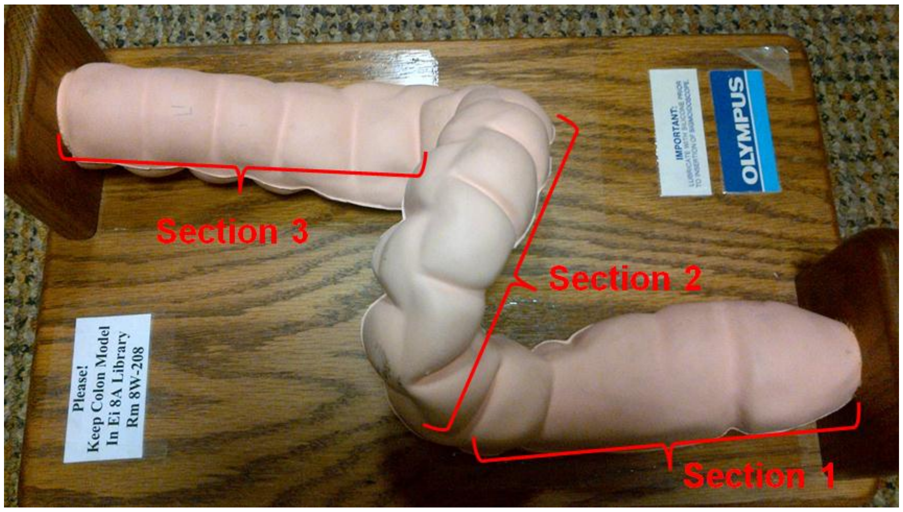


Figure 6.8 Olympus synthetic colon used in our experiments

Table 6.1 shows parameter values used in our implementation. The values were determined based on experiment using a set of images not included in the image set for evaluation in Section 6.4.1 and 6.4.2.

Table 6.1 Parameters and default values

Step	Parameter and value
Section 6.2.4	THRSD_DIST = 0.005
Section 6.3	THRSD_VALID_RC_LENGTH = 0.4
	THRSD_DIST_BTW_RC = 0.07
	THRSD_DIST_BTW_RP = 0.1
	k = 5
	control_prop = 0.4
	margin_prop = 0.25

6.4.1. Results on camera motion estimation

Camera motion consists of translation, rotation, and scaling. Camera translation and rotation are obvious during colonoscopy. Scaling is caused by our reconstructions, not the actual change in the scale of the colon. With *Depth-from-Intensity*, the depth of the reconstructed colon folds depends on the brightness intensity. The brightness can change significantly in subsequent frames. For instance, since camera lens and the light source are not exactly on the same location a protruding fold near the light source may block significant amount of light and cause less illumination for the folds behind it. We evaluate each aspect of the camera motion separately.

Evaluation of Translation Direction

While we cannot measure the exact amount of camera translation, we can observe the direction of the translation in each axis. For the z-axis, the camera direction is either backward (B/+), forward (F/-), or no motion (N). For the y-axis, the direction is either up (U/+), down (D/-), or no motion (N). For the x-axis, the direction is right (R/+), left (L/-), or no motion (N) Table 6.2 shows the ground truth camera direction denoted as GT, the calculated amount of camera motion and direction, the normalized camera motion in the x-axis, y-axis, and z-axis, and the score. The Case ID indicates the start frame of the image sequence used in that case. The # frames indicates the number of frames used in that case. We originally wanted to use at least 30 frames per case; however, in some cases, 30 frames include more than one camera motion. Therefore, we only selected a sequence that shows a single camera motion for evaluation. The score is the number of correct camera motion directions including no motion the software determines. The score is between 0 and 3 because there are three axes. A higher score is desirable. When the absolute normalized camera motion is less than 0.2 (20% of the distance the camera moved), it is difficult to observe the motion direction in that axis from watching the corresponding image sequence. Therefore, we treat such a motion as no motion. Table 6.2 shows the results. In Case 13253, the camera direction in each of the three axes is correct. The score is therefore 3. The absolute normalized score in the x-axis is 0.03 less than 0.2; the software determines that there is no clear motion, which is also correct according to the ground truth. Only in the z-axis, the software found a significant camera forward movement, which is the same as the ground truth. Five out of ten cases show correct camera directions in all axes. Table 6.2 shows that the technique gives the correct camera translation directions for 73% (22/(3*10)) of the translation directions.

Table 6.2 Results of camera motion evaluation: GT is the ground truth direction observed manually. Cal. x, y, z are calculated amount and direction of motion in x, y, and z direction, respectively. Cal. Abs. travel dist. is the absolute distance the camera translates. Nor. x, y, and z capture the normalized amount (Cal. /Cal. Abs travel distance) together with the direction of the translation in x-axis, y-axis, and z-axis, respectively. The camera direction in the x-axis is denoted with right(+)/left(-), in the y-axis with up(+)/down(-) , and in the z-axis with backward(+)/forward(-)

Case ID	# frames	GT (x/y/z)	Cal. x	Cal. y	Cal. z	Cal. Abs. travel dist.	Nor. x	Nor. y	Nor. z	Score
13253	30	N/N/F	-3.3	5.8	-101.3	101.5	**-0.03 (N)**	**0.06 (N)**	**-1.00 (F)**	**3**
13284	30	N/N/F	-10.6	-15.7	-91.2	93.2	**-0.11 (N)**	**-0.17 (N)**	**-0.98 (F)**	**3**
13437	13	R/N/F	56.4	38.2	-117.4	135.7	**0.42 (R)**	0.28 (U)	**-0.87 (F)**	2
4984	17	R/U/N	3.6	3.3	-14.5	15.3	**0.24 (R)**	**0.22 (U)**	-0.95 (F)	2
6165	30	L/N/B	-12.0	0.2	22.6	25.6	**-0.47 (L)**	**0.01 (N)**	**0.88 (B)**	**3**
6846	20	N/N/B	-2.2	9.2	-8.2	12.5	**-0.18 (N)**	0.74 (U)	-0.66 (F)	1
12915	30	N/N/B	8.3	2.8	4.3	9.8	0.85 (R)	0.29 (U)	**0.44 (B)**	1
12521	18	N/U/F	-0.5	22.3	-27.8	35.7	**-0.01 (N)**	**0.62 (U)**	**-0.78 (F)**	**3**
17355	12	N/D/B	-3.7	-4.3	19.5	20.3	**-0.18 (N)**	**-0.21 (D)**	**0.96 (B)**	**3**
16074	29	L/D/B	-6.9	20.9	-21.4	30.7	**-0.22 (L)**	0.68 (U)	-0.70 (F)	1

Evaluation of Rotation Direction

In order to evaluate the rotation direction, we decompose the rotation matrix into an individual rotation matrix for each axis. The rotation matrix for each axis in the right-handed coordinate system is shown below.

$$R_x(\alpha) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix}$$

$$R_y(\beta) = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix}$$

$$R_z(\gamma) = \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

The combined rotation about the x-axis, the y-axis, and the z-axis, respectively is

$$R_z(\gamma)R_y(\beta)R_x(\alpha) = \begin{bmatrix} \cos\beta\cos\gamma & \sin\alpha\sin\beta\cos\gamma - \cos\alpha\sin\gamma & \cos\alpha\sin\beta\cos\gamma + \sin\alpha\sin\gamma \\ \cos\beta\sin\gamma & \sin\alpha\sin\beta\sin\gamma + \cos\alpha\cos\gamma & \cos\alpha\sin\beta\sin\gamma - \sin\alpha\cos\gamma \\ -\sin\beta & \sin\alpha\cos\beta & \cos\alpha\cos\beta \end{bmatrix}.$$

Thus, we decompose the rotation matrix R computed in Section 6.2.2 as below.

$$\beta = \text{asin}(-R_{31})$$

$$\alpha = \text{atan}(R_{32}/R_{33})$$

$$\gamma = \text{atan}(R_{21}/R_{11})$$

For this evaluation, the GT of the rotation about the x-axis (Up, Down), the y-axis (Left, Right, and the z-axis (Counter Clockwise, or Clockwise), respectively are determined for each image sequence. When the camera rotation about an axis is less than about 7°, it is difficult to observe the rotation about that axis from watching the corresponding image sequence. Therefore, when the absolute computer gernerated score is less than 7°, we treat such a rotation as no rotation. The score is the number of correct rotation directions in the three axes.

Table 6.3 Results of rotation evaluation on synthetic colon model images in the x-axis: up (+)/ down (-), in the y-axis: left (+)/right (-), and in the z- axis: counter-clockwise (+)/clockwise (-); *N* denoting no camera motion; we show the start frame and end frame for each sequence.

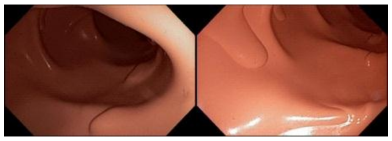
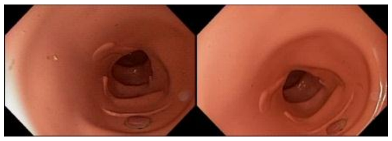
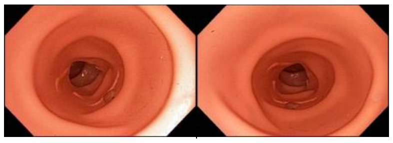
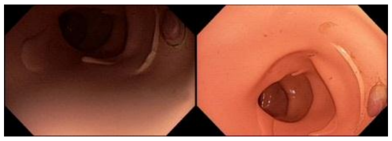
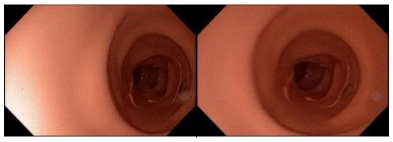
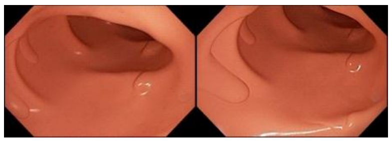
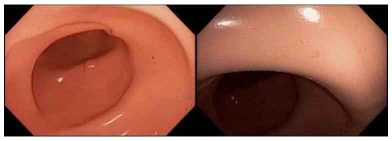
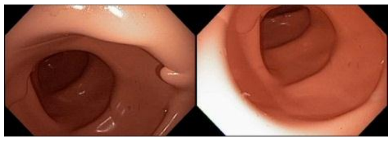
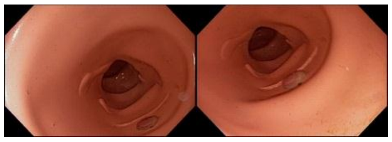
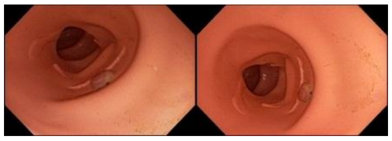
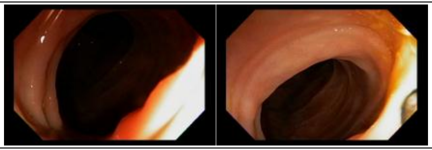
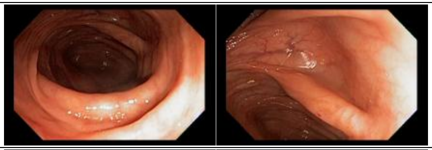
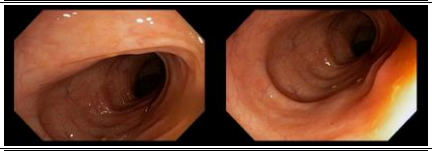
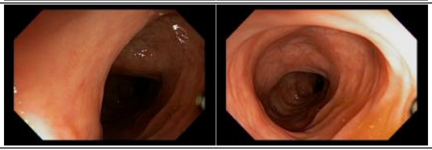
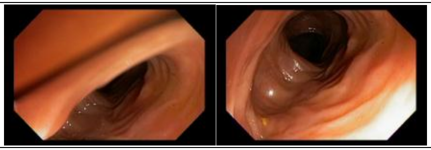
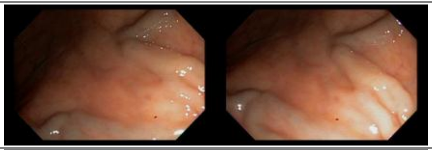
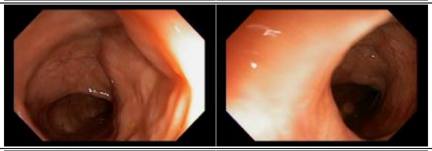
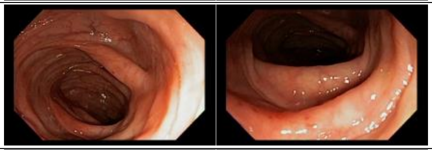
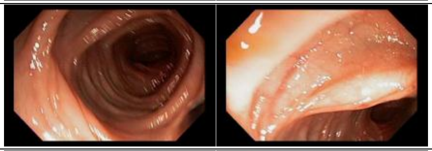
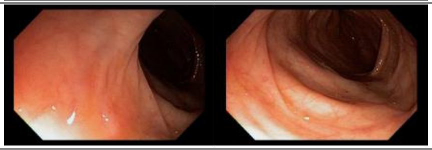
Case ID	# frames	GT	Start frame / End frame	x	y	z	score
6080	30	D/L/CC		**-45.7 (D)**	**36.3 (L)**	-17.8 (C)	2
12805	30	U/R/C		**16.9 (U)**	**-10.4 (R)**	**-17.1 (C)**	**3**
6973	27	N/L/CC		**2.4 (N)**	**11.7 (L)**	**16.6 (CC)**	**3**
12521	18	U/L/N		**25.74 (U)**	**8.7 (L)**	**0.1 (N)**	**3**
6937	8	N/R/N		**3.5 (N)**	**-26.5 (R)**	**-2.4 (N)**	**3**
12017	10	D/L/N		**-7.8 (D)**	**9.8 (L)**	**1.4 (N)**	**3**
12143	7	U/R/N		**20.6 (U)**	**-28.5 (R)**	-64.3 (C)	2
12230	15	D/L/N		**-31.2 (D)**	**27.5 (L)**	**2.5 (N)**	**3**
12863	10	D/R/N		**-28.8 (D)**	**-45.9 (R)**	11.2 (CC)	2
12872	25	U/N/C		**10.3 (U)**	**2.6 (N)**	**-26.9 (C)**	**3**

Table 6.4 Results of rotation evaluation on colonoscopy images of real colons in the x-axis: up (+)/ down(-), in the y-axis: left(+)/right(-), and in the z- axis: counter-clockwise(+)/clockwise(-); *N* denoting no camera motion; we show the start frame and end frame for each sequence.

Case ID	# frames	GT	Start frame / End frame	x	y	z	score
28815	15	U/R/N		**27.7** **(U)**	**-22.4** **(R)**	**3.7** **(N)**	**3**
22670	20	U/N/N		**29.7** **(U)**	**-5.7** **(N)**	**-2.1** **(N)**	**3**
30521	20	D/N/N		**-18.4** **(D)**	-13.8 (R)	**1.0** **(N)**	2
30897	39	N/R/N		19.0 (U)	**-16.5** **(R)**	**-5.7** **(N)**	2
31293	9	D/N/N		**-11.7** **(D)**	11.5 (L)	**-5.0** **(N)**	2
35967	29	N/N/CC		**-2.2** **(N)**	-11.1 (R)	**44.3** **(CC)**	2
32286	14	D/L/N		**-19.4** **(D)**	**27.4** **(L)**	**-1.4** **(N)**	**3**
22781	12	D/N/N		**-31.4** **(D)**	**-2.4** **(N)**	**6.9** **(N)**	**3**
12179	11	U/N/CC		**38.7** **(U)**	**2.0** **(N)**	**8.4** **(CC)**	**3**
19462	32	N/R/N		**-6.6** **(N)**	**-16.2** **(R)**	**-3.8** **(N)**	**3**

Tables 6.3 and 6.4 show that reconstructed rotation directions on the synthetic colon (in Figure 2.1(b)) and real colons, respectively are quite good. Tables 6.3-6.4 show that the technique gives the correct camera rotation directions for 88% (53/(3*20)) of the directions.

Evaluation of scaling

When an image has low illumination, the reconstructed colon model is large since the distance from the camera is far and the reverse projection vectors expand. For a brightly illuminated image, the reconstructed model is small. Figure 6.9 illustrates the same 3D model, but seen with different levels of illumination. When we have an image sequence showing images with lower illumination to brighter illumination, we have a small reconstructed colon model (reconstructed from the brighter frame) to register with a large reconstructed colon model (reconstructed from the darker frame); the scale increases in this case. We denote this case as Bright-to-Dark (B-D). Thus, to fit the model B in Figure 6.9 to model A, we need to scale down (scale<1). For a similar reason, we need to scale up (scale>1) when the image sequence showing images changing from bright to dark, denoted as (D-B).

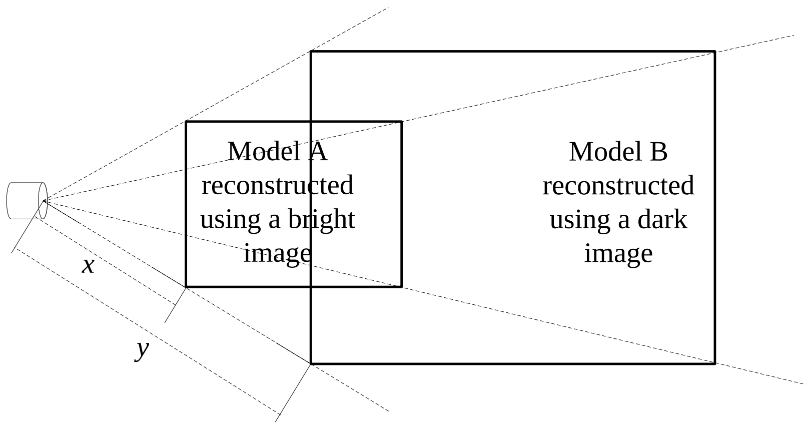

Model A reconstructed using a bright image
Model B reconstructed using a dark image
x
y

Figure 6.9 Bright-to-Dark: scaling of the reconstructed model B down to fit the reconstructed model A (scale: x/y < 1); Dark-to-Bright: scaling of the reconstructed model A up to fit the reconstructed model B (scale: y/x > 1)

Table 6.5 Evaluation of effectiveness of scaling using the synthetic colon images: Bright to dark (B-D): the model grows larger. The calculated scale (Cal. Scale) is less than 1 in this case. Dark to bright (D-B): the model shrinks. Cal. Scale is greater than 1 in this case. N denotes no scaling or scale is around 1 ([0.9, 1.1]).

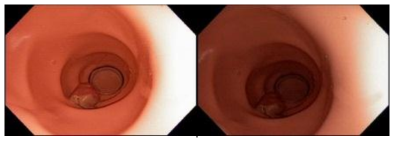
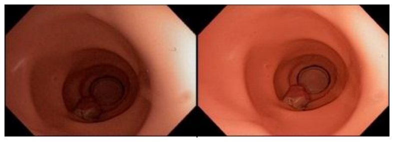
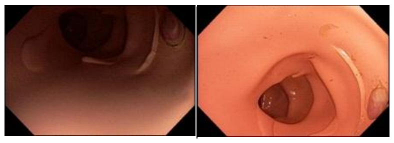
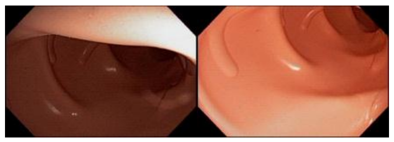
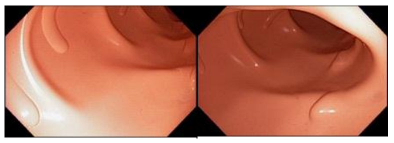
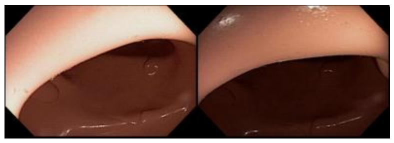
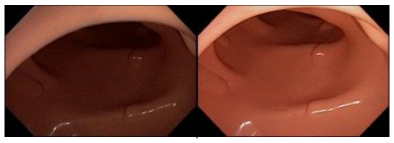
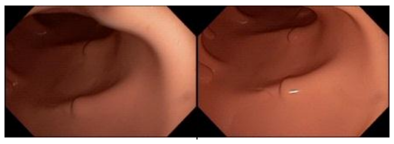
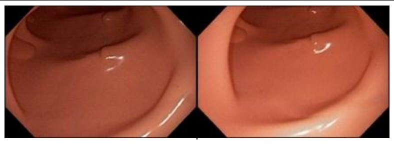
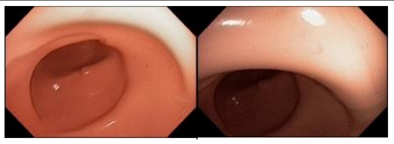
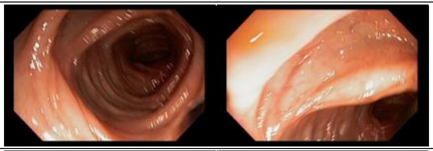
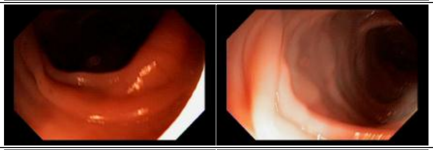
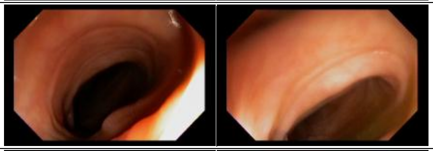
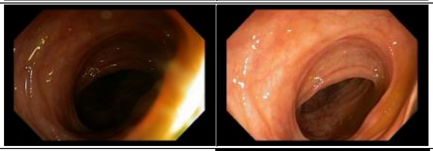
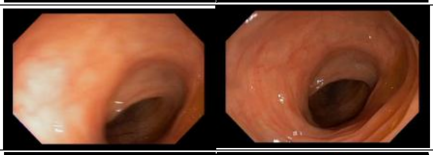
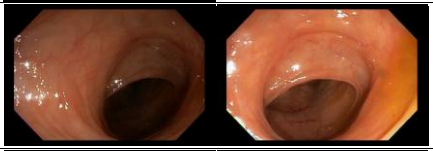
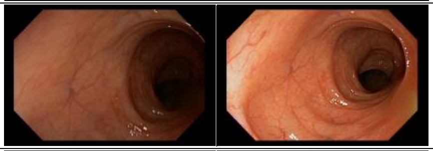
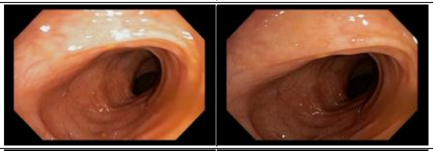
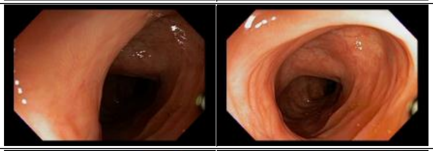
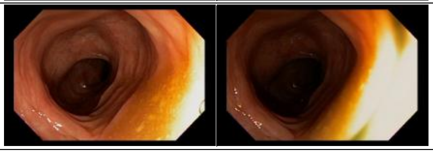
Case ID	# frames	GT	Start frame / End frame	Cal. Scale
4815	30	B-D		**0.43 (B-D)**
4885	30	D-B		1.09 (N)
12521	18	D-B		**2.16 (D-B)**
11937	18	D-B		**1.74 (D-B)**
11963	21	B-D		**0.66 (B-D)**
12038	7	B-D		**0.65 (B-D)**
12054	8	D-B		**1.59 (D-B)**
12075	7	B-D		**0.75 (B-D)**
12110	7	D-B		**1.07 (D-B)**
12145	7	B-D		**0.57 (B-D)**

Table 6.6 Evaluation of effectiveness of scaling on real colons using images from colonoscopy: Bright to dark (B-D): the model grows larger. The calculated scale (Cal. Scale) is less than 1 in this case. Dark to bright (D-B): the model shrinks. Cal. Scale is greater than 1 in this case. N denotes no scaling or scale is around 1 ([0.9, 1.1]).

Case ID	# frames	GT	Start frame/End frame	Cal. Scale
12179	10	D-B		**1.44** **(D-B)**
12226	13	D-B		1.03 (N)
29074	12	D-B		**1.11** (D-B)
29337	14	D-B		**1.40** **(D-B)**
29862	13	B-D		**0.32** **(B-D)**
30132	13	D-B		**1.31** **(D-B)**
30268	14	D-B		**1.50** **(D-B)**
30503	16	B-D		**0.64** **(B-D)**
30898	20	D-B		**1.10** **(D-B)**
30954	10	B-D		**0.56** **(B-D)**

Tables 6.5 and 6.6 show the results of scaling direction test on the synthetic colon (in Figure 2.1(b)) and real colons, respectively. For both of the tests, our software calculated scaling direction reliably for 90% (18/20) of the cases. Only one case out of ten cases in each of the tests has a wrong scale direction.

Table 6.7 Case 12900 (section 1 - straight); Circf represents circumferences. MAD represents the mean absolute difference considering all the folds seen in the image sequence. The reconstructed model denotes after scaling.

	Ground truth (mm)		Reconstructed model		Absolute Difference	
	Depth	Circf.	Depth	Circf.	Depth	Circf.
Fold 1	25	166	41	216	16	50
Fold 2	51	190	51	190	0	0
Fold 3	75	184	65	198	10	14
Fold 4	118	160	96	158	22	2
Fold 5	138	160	124	132	14	28
MAD					13	19

6.4.2. Evaluation of the colon structure

The following tables show the evaluation results on the shape of the reconstructed colons by comparing the depths and circumferences with the ground truth depths and circumferences for different sections of the synthetic colon shown in Figure 6.8. Tables 6.7-6.9 show detail results per case. The circumference of fold 2 was used to derive the scaling factor to scale the virtual colon to the synthetic colon model in Table 6.7-6.8. The scaling is needed since the originally reconstructed model is in its own unit (not in mm). Fold 2 was chosen to determine the scaling factor since it is less sensitive to illumination and occlusion because it is not too close or too far from the camera. Furthermore, it gave the least MAD after applying the scaling factor to the original reconstructed model. The circumference of in Table 6.9, fold 1 was used to scale the originally reconstructed colon model since it gave the least MAD.

Table 6.8 Case 12320 (section 2 - curve); Circf represents circumferences. MAD represents the mean absolute difference considering all the folds seen in the image sequence. The reconstructed model denotes after scaling.

	Ground truth (mm)		Reconstructed model		Absolute Difference	
	Depth	Circf.	Depth	Circf.	Depth	Circf.
Fold 1	53	160	55	184	2	24
Fold 2	85	160	87	160	2	0
MAD					2	12

Table 6.9 Case 16203 (section 3 - straight); Circf represents circumferences. MAD represents the mean absolute difference considering all the folds seen in the image sequence. The reconstructed model denotes after scaling.

	Ground truth (mm)		Reconstructed model		Absolute Difference	
	Depth	Circf.	Depth	Circf.	Depth	Circf.
Fold 1	41	180	50	180	9	0
Fold 2	67	190	61	155	6	35
Fold 3	93	185	99	147	6	39
Fold 4	112	168	109	166	4	2
Fold 5	140	100	121	145	19	45
MAD					9	24

The average depth MAD and circumference MAD on the synthetic colon are 8.5 mm and 17 mm, respectively. Table 6.10 shows that 94.7% of the folds seen in colonoscopy can be reconstructed. During the curvy section of the colon (section 2), only a few folds are shown in the corresponding image sequence, resulting in a small depth MAD and circumference MAD. Although the recall of the folds in this section is 100%, the picture of the synthetic colon indicates that there are more folds in this section. Those folds are not seen continuously in one long image sequence. Hence, the recall of folds cannot reflect this fact well. The average percentage of the depth error is about 10% (8.5/85.8) of the average depth of all the folds seen in all the image sequences. The average percentage of the circumference error is about 10% (17/178.2) of the average circumference of all the folds seen in all the image sequences. We cannot quantitatively show the performance of the technique using endoscopy images of real colons since there is no available ground truth as mentioned in the introduction. Figure 6.10 shows the results of intermediate step and final reconstruction result from colonoscopy images of a real colon.

Table 6.10 Overall results; MAD represents the mean absolute difference considering all the folds seen in each image sequence; recall of folds represents the ratio of the number of reconstructed folds and the number of actual folds seen in each image sequence

Case ID	Depth MAD	Circf. MAD	Recall of folds	# Frames
6955 (section 1)	13	33	5/5	34
6990 (section 1)	10	30	5/5	37
12900 (section 1)	13	19	5/5	144
20399 (section 1)	9	25	3/5	76
19024 (section 1)	5	12	4/4	57
12320 (section 2)	2	12	2/2	21
12117 (section 2)	8	0	1/1	28
18753 (section 2)	12	11	3/3	19
16203 (section 3)	9	24	5/5	17
15940 (section 3)	4	4	3/3	23
Average	8.5	17	36/38 (94.7%)	

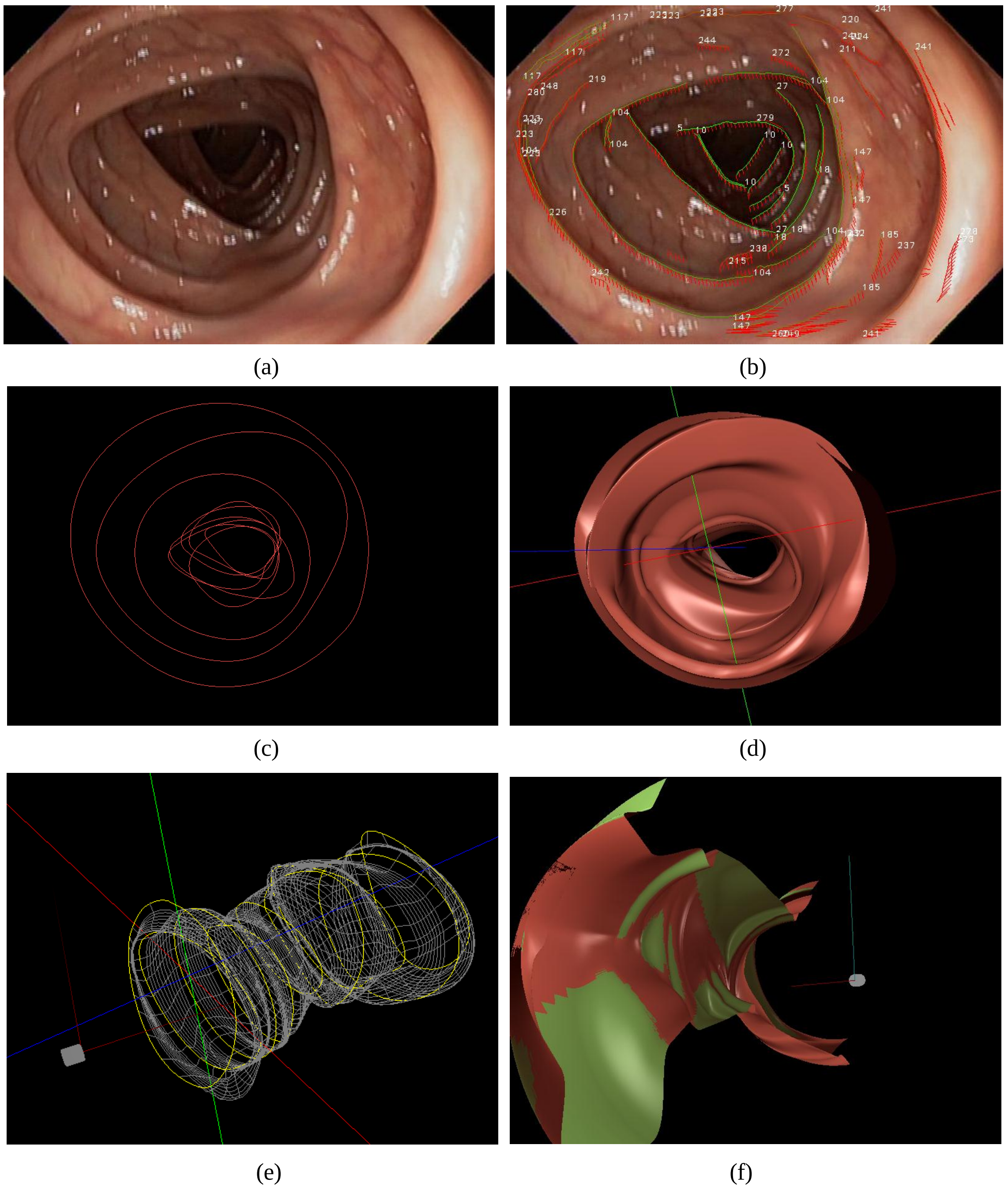


Figure 6.10 (a) Input image; (b) Green edges indicate that the fold edge is tracked through several previous frames. Short red lines indicate the motion of the tracked points. White numbers are global edge ID assigned to each edge. (c) Completed contours of G in the global coordinate system. (d) Rendered G; red, green and blue lines represents x, y and z axis of the global coordinate system. (e) Wireframe of G; fold contours are marked with yellow color. Gray cylinder represents the endoscope. (f) Green area represents unobserved area under the reconstructed camera motion on the reconstructed colon model; red and blue lines from the endoscope indicate the line of sight and the up vector, respectively.

6.5. Conclusion

We evaluated our proposed technique in two aspects: endoscope motion reconstruction and colon structure reconstruction. For the evaluation of motion reconstruction, we compared the software generated camera translations and rotation directions with ground truth motion directions. Since it is not feasible to obtain the precise amount of motion, we only evaluated the direction of motion. Overall, the test result shows that the proposed technique produces directions of endoscope motion in at least 73% in agreement with the ground truth. Further improvement is needed. Accurate camera motion directions are useful for evaluating mucosa inspection pattern in detail and for detection of the boundary between the insertion phase and the withdrawal phase of a procedure [69]. The boundary is important for calculating a number of other quality metrics.

For colon structure reconstruction evaluation, we tested our software with a number of sequences of images with overall illumination changes across the images in the sequence. The illumination changes cause the change in the scale of the local reconstructed colon models that form the global colon model. Our experiments show that, when considering the directions of scaling (up or down), we can scale the models correctly in 90% of the studied cases. The scale factors affect the overall shape of the colon structure reconstruction. The results suggest that improvement is needed to further reduce the circumferential error or depth errors which are currently around 10%. We observed a better recall of fold edges using this technique than our technique in Chapter 5. That means, more folds can be reconstructed in the straight parts of the colon.

CHAPTER 7 QUALITY METRICS FOR COLON MUCOSA INSPECTION DURING COLONOSCOPY

This chapter describes our algorithms and implementation design of a real-time colonoscopy analysis system to derive spiral score as an objective quality metric. The CIF detection technique introduced in Section 3.2 is used as a core algorithm for spiral counting. The algorithm is fast enough to achieve real-time analysis while the existing work by Liu et al. [20] is not. We found a strong correlation between the spiral score measured during the withdrawal phase and the visualization quality score given by domain experts on a data set of 159 colonoscopy videos. Our finding confirms the conclusion of a similar study on 21 colonoscopy videos by Liu et al. [20]. Also, we describe our new method for calculating unobserved mucosa area given camera motion and a structure of a colon segment. We believe that our unobserved area calculation provides a good and detail measurement of quality of colon mucosa inspection during colonoscopy procedures.

7.1. Coarse quality measurement for spiral withdrawal motion

In [20], Liu et al. proposed *spiral score* defined as *the number of times all four quadrants of the colon were consecutively visualized* as a new metric of quality of the mucosa circumferential inspection. They introduced *PCurve*, an image analysis method for CIF estimation. PCurve extends the previous technique [9, 70] by utilizing quasi-parallel curves/edges of nested colon folds to estimate CIF for the types of images that the previous technique could not handle well. Given the detected CIF in each image, the corresponding inspected quadrant is computed. Next, the spiral score is computed based on the above definition. In the same study, the spiral score of the withdrawal phase was found strongly correlated (with Pearson’s correlation coefficient of 0.68) with the ground truth visualization quality on 21 de-identified colonoscopy videos captured using Fujinon endoscopes. PCurve, however, is very time consuming, which is not suitable for real-time analysis and feedback. Instead PCurve, we use our light and fast CIF detection algorithm described in Section 3.2 and study whether our algorithm still produces satisfactory results for counting the spiral score.

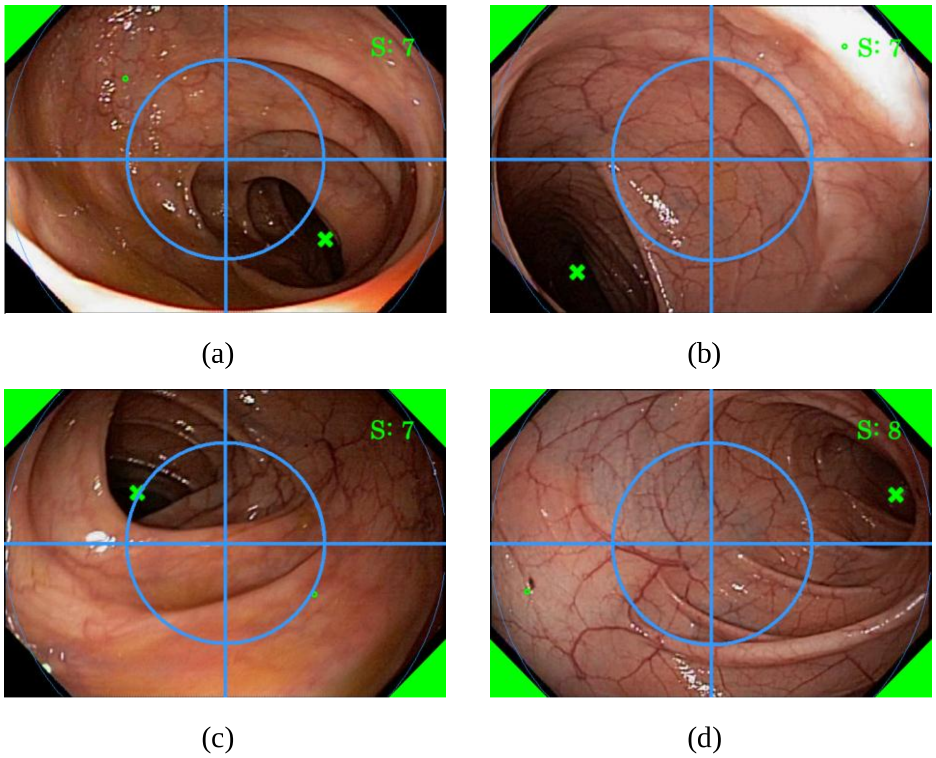


Figure 7.1 Four sequential images form a ‘spiral’; each image is divided into four fixed quadrants using the axes originated at the center of the image; a green marker at the corner indicated that the colon wall in that quadrant was inspected, which is 180° opposite to the quadrant of the detected CIF (marked by an *x*); the top-left, top-right, bottom-right, and bottom-left quadrants were detected, respectively in (a), (b), (c), and (d). The spiral score shown after the letter S increments from 7 in (a) to 8 in (d) after one additional spiral is found.

We followed the approach proposed by Liu et al. to assign the quadrant given CIF [20]. Liu et al. divided each image into four fixed and equal quadrants: top-left quadrant (Q1), top-right quadrant (Q2), bottom right quadrant (Q3), and bottom left quadrant (Q4). The inspected quadrant is diagonally opposite from the quadrant of the detected CIF if the detected CIF is in between the two concentric circles centered at the image center (shown in light blue in Figure 7.1). Figure 7.1(a-d) shows a series of inspection of the quadrants: Q1, Q2, Q3, and Q4. The rationale is that if the detected CIF is too close to the image center, the endoscopist is looking straight down the lumen and does not focus on any particular quadrant of the colon wall. Therefore, we should not mark any particular wall as inspected. To incorporate this domain knowledge, *a laterality threshold* was introduced by Liu et al. [20]. The radius of the inner circle is the multiplication result of a laterality threshold and half image height. Figure 7.1 shows a lateral threshold of 0.66. In our study, we also use the outer circle to filter out wall images (defined in Section 2.1) since we set the detected CIF at the top-left corner of the image if the image is a wall image.

Table 7.1 Parameters and values obtained based on experiments on the Fujinon data set; parameters for the CIF calculation are mentioned in Section 3.2.

Step	Parameter and value
Preprocessing	W=25; S_{smooth}= 3x3; lenThres=80; cannyLow=10; cannyHigh=15; window size for handling thick folds = 32 pixels;
CIF estimation	A=0.002; IT=15; *dist*=50
Case 1	*lenRatio*=0.4; *len*=160
Case 2	*lenRatio*=0.66; *len*=400
Spiral counting	Laterality threshold = 0.66; maxSpiral = 15

Since endoscopists may have individual inspection preference, e.g. clockwise or counter-clockwise order, Liu et al. do not consider the order of the quadrant numbers as important. As soon as all four quadrants are inspected consecutively, one spiral is counted. The spiral score is the number of spirals found so far. We limit the spiral score to be no more than *maxSpiral* to handle the case of long and complex colons causing an unusually high spiral score. The value of maxSpiral was determined empirically and is shown in Table 7.1.

We implemented the aforementioned algorithms in C++ as a Windows dynamic linked library (DLL). We used OpenCV for basic image processing such as smoothing, edge detection, color to gray scale transformations, and circle and line drawing as well as our SAPPHIRE middleware and Software Development Kit (SDK) [71] for routing images from input, analysis module, output and feedback. SAPPHIRE provides ease of coding, maintenance, collaboration among multiple developers, and efficient execution of modules with multithreading. SAPPHIRE takes a text file that lists the names of DLL of required modules and parameters of these modules. SAPPHIRE loads and runs each of the listed modules in their own thread. SAPPHIRE routes data among modules according to the detected dependency and data timestamps. Interested readers are referred to [71] for details.

Our spiral counting module skips processing blurry frames (out-of-focus frames) determined by another module [71]. It outputs the spiral score up to a current frame and a feedback image for the frame. We provide two levels of feedback. The primary feedback includes a green marker at the corner of the detected quadrant and the spiral score. The secondary feedback includes the two blue concentric circles centered at the image center as mentioned previously. The detected CIF location is marked as *x* in green. During routine screening colonoscopy, only the primary feedback should be used since it does not obstruct the viewing area. In practice, the spiral score is shown inside the black corner area. Both types of feedback may be used during teaching or training of endoscopists.

7.2. Performance evaluation and experimental results

We used two video data sets for this study: Fujinon data set and Olympus data set. The capturing rate for the videos in these data sets was 29.97 fps at the resolution of 720x480 pixels. No patient information is included in these videos. The Fujinon data set consists of 21 videos of procedures performed using Fujinon endoscopes. These procedures have excellent bowel preparation (i.e. colon wall has been cleaned carefully and there is little stool inside the colon). For each video, four endoscopists provided the ground truth mucosa visualization score from 0 (poor quality) to 100 (best quality) and the frame number when maximum intubation was reached for the first time. This frame number roughly estimates the start of the withdrawal phase. The ground truth visualization score not only includes the circumferential inspection quality, but also the quality of examination of hepatic and splenic flexures, rectal valves, and the ileocecal valve. The Fujinon data set was used by Liu et al. [20]. It is used in this study to first obtain program parameter values and compare our current work with the previous work [20]. The Fujinon data set is small. Therefore, we also conducted our study on a larger data set. The Olympus data set contains 159 de-identified videos of 159 randomly selected routine screening colonoscopies (one video per procedure) performed in a period of 1.5 years. Olympus endoscopes were used. Two domain experts viewed and annotated these videos as done in the Fujinon data set. We first verified that the assumptions (linearity, normality, independence, and constant variance) required for Pearson's correlation analysis were met. The Pearson’s correlation coefficients obtained from the two data sets are shown in Table 7.2.

Table 7.2 Pearson’s correlation coefficients

Data Set	#videos	P-Curve	Proposed technique	Proposed technique*
Fujinon set	21	0.68	0.72	0.73
Olympus set	159	N/A	0.49	0.58

*With the maximum spiral score limited to 15.

Our analysis shows the Pearson’s correlation coefficient of 0.72 between our spiral score computed during the withdrawal phase and the ground truth visualization score averaged over those given by the four endoscopists. The correlation of 0.72 is considered a very strong correlation since the ground truth score also considers quality of examination of the flexures and values that are not taken into account in the calculation of the spiral score. The correlation coefficient also indicates that our new method performs better than the PCurve method that gave Pearson's correlation coefficient of 0.68 on the same data set.

The Pearson's correlation coefficient between the average of the ground truth visualization quality scores by the two domain experts and the spiral scores on the Olympus set is 0.49. When inspecting the spiral score and the ground truth, we found videos with good visualization quality, but with low spiral score because the colon is easy to navigate. We also found videos with good (not excellent) visualization quality, but with overly high spiral scores due to the complexity of the colon with many turns. This finding suggests that we should consider the colon anatomy when computing the spiral score. For instance, when we set the spiral score to 15 for any videos with more than 15 raw spiral scores (overly high spiral score case), the correlation coefficient significantly improves to 0.58. This is a significant improvement over the non-adjusted spiral score. The maximum spiral score of 15 was observed from the Fujinon set. The Olympus set is much larger than the Fujinon set with more patients and more endoscopists performing these procedures. We believe this causes lower Pearson's correlation in the Olympus set.

The spiral counting module took on average about 13 ms per frame, which is less than 33 ms required for real-time processing. The time was measured on Dell Vostro 460 with Sandy Bridge Core i7, 2600 3.4GHz with 4 GB of RAM running 64 bit Windows 7.

7.3. Unobserved area calculation and visualization

Another quality metric of interest to domain experts is locations of unobserved areas of the colon mucosa and the percentage of these areas. It is not difficult to mark the unobserved area using

OpenGL given a 3D reconstructed colon and the endoscope motion. Since camera lens and light source are at the same location, we just can render shadow on the virtual colon surface using stencil buffer. However, in this way, we cannot calculate the percentage of the unobserved area because OpenGL does not provide any method to tell which polygon the shadow is rendered. Thus, we calculate the unobserved area as follows. For each polygon, we check three conditions: (i) if the polygon is facing the endoscope (the origin of the coordinate system), (ii) if the polygon is within the camera field-of-view, and (iii) if there are no obstacles from the polygon to the endoscope. The first condition is satisfied if the angle difference (in the range of [0, 180]) between the normal of the polygon and the vector from the camera to the center of the polygon is greater than 90°. We call this vector to this polygon *the line of sight of the polygon*. The second condition is satisfied if the angle difference between the line of sight of the polygon and the camera direction is at most half of FOV of the camera. The last condition is satisfied if the distance of the polygon from the camera is same as the OpenGL-provided depth (Z-value) of the corresponding pixel in the viewport. Every pixel in a viewport has its own depth value indicating the distance of the closest object from the camera in that direction. When any of the three conditions is not satisfied, we count that polygon as an unobserved polygon. Therefore, we can calculate the percentage of the unobserved mucosa area as the percentage of unobserved polygons.

7.4. Conclusion

In this chapter, we introduced a method to count spiral motions of the endoscope as a quality metric for colonoscopy procedures. We developed the software that provides feedback of inspected quadrant as well as the number of the spirals seen so far by detecting CIF per frame. Furthermore, we show that the CIF location can be used to compute the spiral score---the number of times all sides of the colon wall is inspected as the quality metric, as effective as the PCurve method studied previously We found that the spiral score metric has a strong correlation with visualization quality given by domain experts. Hence, it is promising to be used to determine one aspect of quality inspection of colonoscopy. The average processing time is 13 ms per image on a modern workstation, which makes this technique clinically and practically more useful for real-time processing and real-time feedback of quality of mucosa inspection during colonoscopy. The unobserved colon mucosa area with the location of the endoscope calculated by our proposed technique provides fine detail measurements of quality of the mucosa inspection. Furthermore, it potentially can enable the endoscopist to go back and review these uninspected areas. This second metric is yet to be evaluated more thoroughly. Other metrics such as bowel preparation quality, efforts of the endoscopist in cleaning the colon should also be taken into account. Last, we can potentially use 3D endoscope motion for better colonoscopy quality measurement such as more accurate spiral motion counting and end-of-insertion-phase detection.

CHAPTER 8 CONCLUSION AND FUTURE WORK

We have described in great detail about our technique for automated reconstruction of a 3D virtual colon from 2D colonoscopy images and our evaluation of the technique. Our technique is the first fully automated technique of its kind. Most other existing work focuses on reconstructing the colon surface. We have overcome several major challenges. We use pixel intensity calibration to handle an uneven distribution of light on the colon surface. We found that ambient light affects the relationship between brightness intensity and distance of the colon surface from the camera described by the Lambertian's cosine law. We proposed a modification of the Lambertian's cosine law and showed the effectiveness of our modification. We also extended Umeyama's registration method for two point sets to register models reconstructed from sequential frames to create a global virtual colon. Nevertheless, several challenges remain. Since we use *Depth-from-Intensity* to estimate the depth of a point on a colon fold contour in 3D space, significant illumination changes between neighboring frames has a considerable impact on the scale of the corresponding virtual colons. Errors in fold edge tracking can lead to wrong contour shape. Our technique is reasonably good for reconstruction of the shape of the straight segments of the colon, but less effective for the curvy segments. The experimental results show that reconstruction of the directions of camera rotation is quite accurate. The calculation of the camera translation direction should be further improved in order to correctly estimate the boundary between the insertion phase and the withdrawal phase of colonoscopy in which a number of objective quality metrics during the withdrawal phase have been proposed.

We studied the effectiveness of the spiral score metric---the number of times all sides of the colon wall is inspected. We developed software that provides feedback of the inspected quadrants as well the number of the spirals seen so far. We found that the spiral score metric has a strong correlation with the ground truth visualization quality of the colon mucosa. Furthermore, we have our analysis and feedback software running on two PCs in two educational rooms at Mayo Clinic, Rochester, MN. Whether the computer generated feedback help improving quality of colonoscopy of endoscopists in these rooms are being studied by a separate research team. We introduced the method to estimate locations of unobserved areas in the colon during colonoscopy as well as a percentage of these unseen areas given a reconstructed virtual colon and camera motion. These metrics are of great interest to endoscopists to pinpoint these areas as feedback for a second round inspection during colonoscopy. Nevertheless, other metrics for measuring bowel preparation quality, efforts of the endoscopist in cleaning the colon should also be taken into account to fully capture quality of colonoscopy.

Future work includes an improvement on selecting feature points to track since a subtle difference using two different sets of feature points for tracking can generate accumulated errors when registering many sequential models. Furthermore, instead of registering local models reconstructed using adjacent frames, we can register the local models to the global model directly to reduce accumulated errors and improve the accuracy of the reconstructed model shape for both straight and curvy segments of the colon. This will improve the accuracy of the unseen area calculation method as well. A better design for contour completion able to handle curvy segments of the colon is needed. Because of frequent blurry frames that appear during colonoscopy and frames with a close up inspection of the colon mucosa without the lumen in view, we cannot generate local colon models from these frames. Hence, during the withdrawal phase of a colonoscopy procedure, a number of global colon models corresponding to non-blurry frame and non-wall segments are expected. We need an effective technique to register these global colon models together for the proposed technique to be useful for clinical trials and practice.

REFERENCES

[1] American Cancer Society. Colorectal Cancer Facts & Figures, 2010

[2] T. R. Levin, "Colonoscopy Capacity," *Gastroenterology,* pp. 1841-1849, December 2004.

[3] D. Lieberman, "Quality and colonoscopy: a new imperative," *Gastrointestinal Endoscopy,* vol. 61, pp. 392-394, Mar 2005.

[4] A. Pabby, R. E. Schoen, J. L. Weissfield, R. Burt, J. W. Kikendall, P. Lance, E. Lanza, and A. Schatzkin, "Analysis of Colorectal Cancer Occurrence During Surveillance Colonoscopy in the Dietary Prevention Trial," *Gastroenterology Endoscopy,* vol. 61, pp. 385-391, 2005.

[5] D. K. Rex, J. H. Bond, S. Winawer, T. R. Levin, R. W. Burt, D. A. J., and e. al, "Quality in the Technical Performance of Colonoscopy and the Continuous Quality Improvement Process for Colonoscopy: Recommendations of the US Multi-Society Task Force on Colorectal Cancer," *American Journal Gastroenterol.,* vol. 97, pp. 1296-1308, 2002.

[6] C. D. Johnson, J. G. Fletcher, R. L. MacCarty, J. N. Mandrekar, W. S. Harmsen, P. J. Limburg, and L. A. Wilson, "Effect of slice thickness and primary 2D versus 3D virtual dissection on colorectal lesion detection at CT Colonography in 452 asymptomatic adults," *American Journal of Roentgenology,* vol. 189, pp. 672-680, Sep 2007.

[7] D. K. Rex, J. L. Petrini, T. H. Baron, A. Chak, J. Cohen, S. E. Deal, B. Hoffman, B. C. Jacobson, K. Mergener, B. Pertersen, M. A. Safdi, D. O. Faigel, and I. M. Pike, "Quality indicators for colonoscopy," *Gastrointestinal Endoscopy,* vol. 63, pp. S16-S26, 2006.

[8] J. Oh, S. Hwang, Y. Cao, W. Tavanapong, D. Y. Liu, J. Wong, and P. C. de Groen, "Measuring Objective Quality of Colonoscopy," *IEEE Transactions on Biomedical Engineering,* vol. 56, pp. 2190-2196, Sep 2009.

[9] D. Y. Liu, Y. Cao, W. Tavanapong, J. Wong, J. Oh, and P. C. de Groen, "Quadrant coverage histogram: A new method for measuring quality of colonoscopic procedures," in *Proc. of Annual International Conference of the IEEE Engineering in Medicine and Biology Society, Vols 1-16,* pp. 3470-3473, 2007.

[10] S. Hwang, J. Oh, W. Tavanapong, J. Wong, J. Oh, and P. C. de Groen, "Stool Detection in Colonoscopy Videos," in *Proc. of Int'l Conf. of the IEEE Engineering in Medicine and Biology Society (EMBC)*, Vancouver, BC, Canada, August 2008, pp. 3004-3007.

[11] Y. Cao, D. Liu, W. Tavanapong, J. Wong, J. Oh, and P. C. de Groen, "Computer-aided Detection of Diagnostic and Therapeutic Operations in Colonoscopy Videos," *IEEE Transactions on Biomedical Engineering,* vol. 54, pp. 1268-1279, 2007.

[12] Y. Wang, "Edge cross-section profile for colonoscopic object detection," Ph. D., Computer Science, Iowa State University, 2012.

[13] S. Ameling, S. Wirth, and D. Paulus, "Methods for Polyp Detection in Colonoscopy Video: A Review," Technical Report, University of Koblenz-Landau2008.

[14] Y. Wang, W. Tavanapong, J. Wong, J. Oh, and P. C. de Groen, "Edge Cross-Section Features for Detection of Appendiceal Orifice Appearance in Colonoscopy Videos," in *Proc. of Int'l Conf. of the IEEE Engineering in Medicine and Biology Society (EMBC)*, Vancouver, British Columbia, Canada, Aug, 2008, pp. 3000-3003.

[15] J. Zhou, A. Das, F. Li, and B. X. Li, "Circular Generalized Cylinder Fitting for 3D Reconstruction in Endoscopic Imaging Based on MRF," *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, Vols 1-3,* pp. 448-455, 2008.

[16] A. Kaufman and J. N. Wang, "3D surface reconstruction from endoscopic videos," *Visualization in Medicine and Life Sciences,* pp. 61-+, 2008.

[17] D. Koppel, C. L. Chen, Y. F. Wang, H. Lee, J. Cu, A. Poirson, and R. Wolters, "Toward automated model building from video in computer-assisted diagnoses in colonoscopy - art. no. 65091L," *Medical Imaging 2007: Visualization and Image-Guided Procedures, Pts 1 and 2,* vol. 6509, pp. L5091-L5091, 2007.

[18] G. N. Khan and D. F. Gillies, "Vision based navigation system for an endoscope," *Image and Vision Computing,* vol. 14, pp. 763-772, Dec 1996.

[19] K. Deguchi, "Shape reconstruction from endoscope image by its shadings," *Mf '96 - 1996 IEEE/Sice/Rsj International Conference on Multisensor Fusion and Integration for Intelligent Systems,* pp. 321-328, 1996.

[20] X. Liu, W. Tavanapong, J. Wong, J. Oh, and P. C. de Groen, "Automated Measurement of Quality of Mucosa Inspection for Colonoscopy," in *Proc. of Int'l Conf. on Computational Science*, Amsterdam, Netherlands, 2010.

[21] H. Bakstein and T.Pajdla., "Panoramic Mosaicing with a 180˚ Field of View Lens," in *Proc. of IEEE Workshop on Omnidirectional Vision*, Washington, DC, USA, 2002, pp. 60-68.

[22] J. Kannala and S. Brandt, "A generic camera calibration method for fish-eye lenses," in *Proc. of the 17th International Conference on Pattern Recognition, Vol 1,* pp. 10-13, 2004.

[23] T. Stehle, D. Truhn, T. Aach, C. Trautwein, and J. Tischendorf, "Camera calibration for fish-eye lenses in endoscopy with an application to 3D reconstruction," in *Proc. of IEEE International Symposium on Biomedical Imaging : Macro to Nano, Vols 1-3,* pp. 1176-1179, 2007.

[24] E. Prados and O. Faugeras, "Shape from shading: a well-posed problem?," in *Proc. of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol 2,* pp. 870-877, 2005.

[25] R. Zhang, P. S. Tsai, J. E. Cryer, and M. Shah, "Shape from shading: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 21, pp. 690-706, Aug 1999.

[26] B. K. P. Horn and M. J. Brooks, "The Variational Approach to Shape from Shading," in *Proc. of Computer Vision Graphics and Image Processing,* vol. 33, pp. 174-208, Feb 1986.

[27] B. K. P. Horn, "Height and Gradient from Shading," *International Journal of Computer Vision,* vol. 5, pp. 37-75, Aug 1990.

[28] B. K. P. Horn, R. S. Szeliski, and A. L. Yuille, "Impossible Shaded Images," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 15, pp. 166-170, Feb 1993.

[29] J. Oliensis and P. Dupuis, "A Global Algorithm for Shape from Shading," in *Proc. of International Conference on Computer Vision*, pp. 692-701, 1993.

[30] M. Oren and S. K. Nayar, "Diffuse-Reflectance from Rough Surfaces," *1993 IEEE Computer Society Conference on Computer Vision and Pattern Recognition : Proceedings*, pp. 763-764, 1993.

[31] B. T. Phong, "Illumination for Computer Generated Pictures," *Communications of the Acm*, vol. 18, pp. 311-317, 1975.

[32] Q. F. Zheng and R. Chellappa, "Estimation of Illuminant Direction, Albedo, and Shape from Shading," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, pp. 680-702, Jul 1991.

[33] L. Cohen, L. Vinet, P. T. Sander, and A. Gagalowicz, "Hierarchical region based stereo matching," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, San Diego, CA, USA, 1989, pp. 416-421.

[34] A. Bigand, T. Bouwmans, and J. P. Dubus, "A new stereomatching algorithm based on linear features and the fuzzy integral," *Pattern Recognition Letters*, vol. 22, pp. 133-146, Feb 2001.

[35] J. Kostkova, J. Cech, and R. Sara, "Dense stereomatching algorithm performance for view prediction and structure reconstruction," in *Proc. of Image Analysis*, vol. 2749, pp. 101-107, 2003.

[36] J. Sun, Y. Li, S. B. Kang, and H. Y. Shum, "Symmetric stereo matching for occlusion handling," *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol 2, Proceedings*, pp. 399-406, 2005.

[37] J. Sun, N. N. Zheng, and H. Y. Shum, "Stereo matching using belief propagation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, pp. 787-800, Jul 2003.

[38] S. Srivastava, S. J. Ha, S. H. Lee, N. I. Cho, and S. U. Lee, "Stereo Matching Using Hierarchical Belief Propagation Along Ambiguity Gradient," in *Proc. of IEEE International Conference on Image Processing, Vols 1-6*, pp. 2061-2064, 2009.

[39] D. A. Forsyth, "Shape from texture without boundaries," *Computer Vision - ECCV 2002*, vol. 2352, pp. 225-239, 2002.

[40] J. Malik and R. Rosenholtz, "Computing local surface orientation and shape from texture for curved surfaces," *International Journal of Computer Vision*, vol. 23, pp. 149-168, Jun 1997.

[41] C. Tomasi and T. Kanade, "Shape and Motion from Image Streams - a Factorization Method," in *Proc. of the National Academy of Sciences of the United States of America*, vol. 90, pp. 9795-9802, Nov 1 1993.

[42] L. Torresani, D. B. Yang, E. J. Alexander, and C. Bregler, "Tracking and modeling non-rigid objects with rank constraints," in *Proc. of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol 1*, pp. 493-500, 2001.

[43] C. J. Taylor and D. J. Kriegman, "Structure and Motion from Line Segments in Multiple Images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, pp. 1021-1032, Nov 1995.

[44] L. Zhang, B. Curless, A. Hertzmann, and S. M. Seitz, "Shape and motion under varying illumination: Unifying structure from motion, photometric stereo, and multi-view stereo," in *Proc. of IEEE International Conference on Computer Vision, Vols I and II*, pp. 618-625, 2003.

[45] L. Hajder, D. Chetverikov, and I. Vajk, "Robust structure from motion under weak perspective," in *Proc. of International Symposium on 3d Data Processing, Visualization, and Transmission*, pp. 828-835, 2004.

[46] *Lambert's cosine law*. Available: http://en.wikipedia.org/wiki/Lambert%27s_cosine_law

[47] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, pp. 91-110, Nov 2004.

[48] M. A. Fischler and R. C. Bolles. (1981) Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of ACM*. 381–395.

[49] D. Nister, "An efficient solution to the five-point relative pose problem," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, pp. 756-770, Jun 2004.

[50] R. I. Hartley, "In defense of the eight-point algorithm," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, pp. 580-593, Jun 1997.

[51] D. Hong, W. Tavanapong, J. Wong, J. Oh, and P. C. de Groen, "3D Reconstruction of Colon Segments from Colonoscopy Images," in *Proc. of IEEE Int'l Conf. on Bioinformatics and Bioengineering*, Taiwan, 2009, pp. 53-60.

[52] A. Gruen and D. Akca, "Least squares 3D surface and curve matching," *Isprs Journal of Photogrammetry and Remote Sensing*, vol. 59, pp. 151-174, May 2005.

[53] P. Dalal, L. L. Ju, M. McLaughlin, X. R. Zhou, H. Fujita, and S. Wang, "3D Open-Surface Shape Correspondence for Statistical Shape Modeling: Identifying Topologically Consistent Landmarks," in *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pp. 1857-1864, 2009.

[54] P. Dalal, B. C. Munsell, S. Wang, J. J. Tang, K. Oliver, H. Ninomiya, X. G. Zhou, and H. Fujita, "A fast 3D correspondence method for statistical shape Modeling," *2007 IEEE Conference on Computer Vision and Pattern Recognition, Vols 1-8*, pp. 1322-1329, 2007.

[55] O. D. Faugeras and M. Hebert, "The Representation, Recognition, and Locating of 3-D Objects," *International Journal of Robotics Research*, vol. 5, pp. 27-52, Fal 1986.

[56] P. J. Besl and N. D. Mckay, "A Method for Registration of 3-D Shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, pp. 239-256, Feb 1992.

[57] K. S. Arun, T. S. Huang, and S. D. Blostein, "Least-squares fitting of two 3-d point sets," *IEEE Trans Pattern Anal Mach Intell*, vol. 9, pp. 698-700, May 1987.

[58] B. K. P. Horn, "Closed-Form Solution of Absolute Orientation Using Unit Quaternions," *Journal of the Optical Society of America a-Optics Image Science and Vision*, vol. 4, pp. 629-642, Apr 1987.

[59] S. Umeyama, "Least-Squares Estimation of Transformation Parameters between 2 Point Patterns," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, pp. 376-380, Apr 1991.

[60] G. Bradski and A. Kaehler, *Learning OpenCV Computer Vision with the OpenCV Library*: O'Reilly, 2008.

[61] Y. N. Deng and B. S. Manjunath, "Unsupervised segmentation of color-texture regions in images and video," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 23, pp. 800-810, Aug 2001.

[62] J. Malik, S. Belongie, T. Leung, and J. B. Shi, "Contour and texture analysis for image segmentation," *International Journal of Computer Vision,* vol. 43, pp. 7-27, 2001.

[63] Z. W. Tu and S. C. Zhu, "Image segmentation by data-driven Markov Chain Monte Carlo," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 24, pp. 657-673, May 2002.

[64] L. A. Vese and T. F. Chan, "A multiphase level set framework for image segmentation using the Mumford and Shah model," *International Journal of Computer Vision*, vol. 50, pp. 271-293, Dec 2002.

[65] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *International Journal of Computer Vision,* vol. 59, pp. 167-181, Sep 2004.

[66] Y. Boykov and G. Funka-Lea, "Graph cuts and efficient N-D image segmentation," *International Journal of Computer Vision,* vol. 70, pp. 109-131, Nov 2006.

[67] T. Sederberg. *BYU Bézier curves*.

[68] B. D. Lucas and T. Kanade, "An Iterative Image Registration Technique with an Application to Stereo Vision," in *Proc. of International joint conference on Artificial intelligence*, San Francisco, CA, USA, 1981, pp. 674 - 679.

[69] J. Oh, S. Hwang, Y. Cao, W. Tavanapong, and P. C. de Groen, "Measuring Objective Quality of Colonoscopy," *IEEE Transactions on Biomedical Engineering,* vol. 56, pp. 2190-2196, September 2009.

[70] D. Liu, Y. Cao, W. Tavanapong, J. Wong, J. Oh, and P. C. de Groen, "Mining Colonoscopy Videos to Measure Quality of Colonoscopic Procedures," in *Proc. of IASTED Int'l Conf. on Biomedical Engineering (BioMed)*, Innsbruck, Austria, 2007, pp. 409-414.

[71] S. R. Stanek, W. Tavanapong, J. Wong, J. Oh, R. D. Nawarathna, J. Muthukudage, and P. C. de Groen, "SAPPHIRE Middleware and Software Development Kit for Medical Video Analysis," in *Proc. of International Symposium on Computer-Based Medical Systems (Cbms),* 2011.