Edge cross-section profile for colonoscopic object detection

Yi Wang
Iowa State University

Edge cross-section profile for colonoscopic object detection

by

Yi Wang

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

Major: Computer Science

Program of Study Committee:
Carl K. Chang, Major Professor
Wallapak Tavanapong
Johnny S. Wong
Les Miller
Alexander Stoytchev
Wensheng Zhang

Iowa State University

Ames, Iowa

2012

TABLE OF CONTENT

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGMENTS

When I was thirteen years old, my farther sent me to a summer school. It was the first time that I touched a computer – a 486 Intel machine, and I was very excited when it did what I commanded. I got to know more about computer science when I grew older, and ultimately, I was very happy and satisfied with my experiences in the study and research these years. There are many people I would like to acknowledge and say thank you to for all their supports and help.

I want to express my heartfelt thanks to Prof. Wallapak Tavanapong, my research advisor, for her efforts to teach me and advice me with infinite patience. The work presented in the dissertation is not possible without her advice and support. She taught me how to go deeper in a research topic, how to come up with new research ideas, how to write a paper with good quality, and many aspects that would influence my research, life and future.

I would like to gratefully thank Prof. Johnny Wong, my research co-advisor, for his help and support in this research. He encouraged me when I felt depressed because of the challenges in the research. He made his efforts to support us, lead us, and coordinate the work in our research group. He is a good mentor and leader.

I would like to thank Prof. Piet de Groen and his research team at Mayo Clinic Rochester for providing colonoscopy video data, for his help on software testing and deployment at Mayo Clinic Rochester, conference and journal paper revision, and other collaborative work. I would like to thank Prof. JungHwan Oh and his research group at the University of North Texas for the collaborative work on software development, paper revision, and other related work on this research. I would also like to express many thanks to my fellow graduate students in our research lab: Xuemin (Helen) Liu, Kihwan Kim, Dongho Hong, Sean Stanek for their great help on this research. I also want to thank useful discussion and inspiration from Georgi I. Batinov, Yetian Chen, and many other graduate students from the ISU Computer Science department.

I would like to thank other professors on my POS committee members: Prof. Carl K. Chang, Prof. Les Miller, Prof. Alexander Stoytchev and Prof. Wensheng Zhang, for their discussion, help and efforts put on this dissertation.

I have to express my special thanks to my wife, Yan Gu, who always helps me, encourages me in my daily life and research. I cannot finish without saying “thanks” to my parents who educated me, supported me and encourage me to pursue my Ph.D. They would be very proud of me.

This research is supported by a research assistantship offered by the Department of Computer Science funded through grants from the U.S. National Science Foundation under Grant STTR-0740596, IIS-0513777, IIS-0513809, IIS 0513582, IIP-0956847, and funding from EndoMetric Corporation and Mayo Clinic Rochester.

This dissertation is dedicated to my family and friends who love me and who I love.

ABSTRACT

Colorectal cancer is the second leading cause of cancer-related deaths, claiming close to 50,000 lives annually in the United States alone. Colonoscopy is an important screening tool that has contributed to a significant decline in colorectal cancer-related deaths. During colonoscopy, a tiny video camera at the tip of the endoscope generates a video signal of the internal mucosa of the human colon. The video data is displayed on a monitor for real-time diagnosis by the endoscopist. Despite the success of colonoscopy in lowering cancer-related deaths, a significant miss rate for detection of both large polyps and cancers is estimated around 4-12%. As a result, in recent years, many computer-aided object detection techniques have been developed with the ultimate goal to assist the endoscopist in lowering the polyp miss rate. Automatic object detection in recorded video data during colonoscopy is challenging due to the noisy nature of endoscopic images caused by camera motion, strong light reflections, the wide angle lens that cannot be automatically focused, and the location and appearance variations of objects within the colon. The unique characteristics of colonoscopy video require new image/video analysis techniques.

The dissertation presents our investigation on edge cross-section profile (ECSP), a local appearance model, for colonoscopic object detection. We propose several methods to derive new features on ECSP from its surrounding region pixels, its first-order derivative profile, and its second-order derivative profile. These ECSP features describe discriminative patterns for different types of objects in colonoscopy. The new algorithms and software using the ECSP features can effectively detect three representative types of objects and extract their corresponding semantic unit in terms of both accuracy and analysis time.

The main contributions of dissertation are summarized as follows. The dissertation presents 1) a new ECSP calculation method and feature-based ECSP method that extracts features on ECSP for object detection, 2) edgeless ECSP method that calculates ECSP without using edges, 3) part-based multi-derivative ECSP algorithm that segments ECSP, its 1st - order and its 2nd - order derivative functions into parts and models each part using the method that is suitable to that part, 4) ECSP based algorithms for detecting three representative types of colonoscopic objects including appendiceal orifices, endoscopes during retroflexion operations, and polyps and extracting videos or segmented shots containing these objects as semantic units, and 5) a software package that implements these techniques and provides meaningful visual feedback of the detected results to the endoscopist. Ideally, we would like the software to provide feedback to the endoscopist before the next video frame becomes available and to process video data at the rate in which the data are captured (typically at about 30 frames per second (fps)). This real-time requirement is difficult to achieve using today's affordable off-the-shelf workstations. We aim for achieving near real-time performance where the analysis and feedback complete at the rate of at least 1 fps.

The dissertation has the following broad impacts. Firstly, the performance study shows that our proposed ECSP based techniques are promising both in terms of the detection rate and execution time for detecting the appearance of the three aforementioned types of objects in colonoscopy video. Our ECSP based techniques can be extended to both detect other types of colonoscopic objects such as diverticula, lumen and vessel, and analyze other endoscopy procedures, such as laparoscopy, upper gastrointestinal endoscopy, wireless capsule endoscopy and EGD. Secondly, to our best knowledge, our polyp detection system is the only computer-aided system that can warn the endoscopist the appearance of polyps in near real time. Our retroflexion detection system is also the first computer-aided system that can detect retroflexion in near real-time. Retroflexion is a maneuver used by the endoscopist to inspect the colon area that is hard to reach. The use of our system in future clinical trials may contribute to the decline in the polyp miss rate during live colonoscopy. Our system may be used as a training platform for novice endoscopists. Lastly, the automatic documentation of detected semantic units of colonoscopic objects can be helpful to discover unknown patterns of colorectal cancers or new diseases and used as educational resources for endoscopic research.

CHAPTER 1 INTRODUCTION

This chapter introduces the background on colonoscopy, addresses the motivation of our research and summarizes the contribution of this dissertation.

1.1 Background on Colonoscopy

The colon is a hollow, muscular tube about *150* centimeters long. It consists of six parts: cecum with appendix, ascending colon, transverse colon, descending colon, sigmoid and rectum. See Figure 1.1. Colonoscopy is currently the preferred screening modality for prevention of colorectal cancer [1] [2]. A colonoscopic procedure consists of two phases: insertion phase and withdrawal phase. During the insertion phase, the endoscopist gradually inserts a flexible endoscope into the most proximal part of the colon (signified by the appearance of the appendiceal orifice or the terminal ileum). During the withdrawal phase, the endoscopist then gradually withdraws the scope while performing careful examination of the colon mucosa based on images generated from the tiny wide-angle lens camera at the tip of the scope. The video data is displayed on a monitor for real-time diagnosis by the endoscopist. Careful examination of colon, biopsy and therapeutic operations such as polyp removal are typically performed during the withdrawal phase [3].

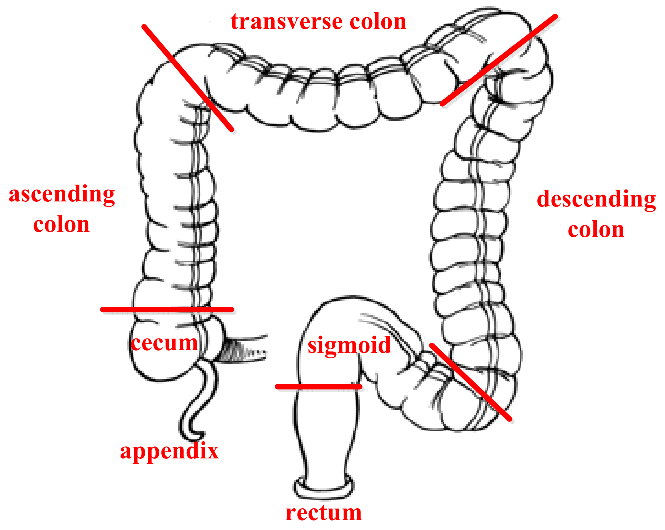


Figure 1.1: The segments of the human colon. The picture is originally from [4].

The entire colonoscopy procedure typically lasts between *20* to *40* minutes. We call the recorded video data during colonoscopy the ***colonoscopy video.*** The appearance of the appendiceal orifice (appendix in Figure 1.1) during colonoscopy indicates a complete traversal of the colon, which is an important quality indicator of the colon examination. A colonic polyp is a growth of tissue that develops in the inner lining of the colon or the rectum. Polyps are the most common seen abnormalities. Most colorectal cancers develop from adenomatous polyps [1]. Early detection and removal of adenomatous polyps are the main goals during colonoscopy. Colonoscopy is complex to master. The colon structure for each patient varies. Some colons have many more turns than others, depending on age, gender, etc. Polyps may be missed because the patient may not clean the colon well before the exam, preventing the endoscopist from traversing the entire colon or seeing the colon mucosa clearly. The endoscopist may miss inspection on one side of a colon wall or some sections of the colon or behind some colon folds in which an adenomatous polyp is located. The endoscopist may not recognize a polyp, or may forget to remove the polyp that has been recognized, or may not remove all parts of the polyp.

A significant miss rate for detection of both large polyps and cancers was reported [2] [5] [6]. The average miss rate is estimated at *4-12%*, but as high as *80%* (*4/5*) was reported in one study in 2007 [7]. A large Canadian study reported that colonoscopy offered virtually no protection against right sided colon cancer and only a *60-70%* protection against left sided colon cancer [8]. The miss rate varies among endoscopists and it may be related to the experience of the endoscopist and the location of the lesion in the colon as suggested by the Canadian study. Recent reports in [9] [10] indicate the identity of the endoscopist performing the procedure as the dominant indicator of the protective effect of colonoscopy.

1.2 Motivation

Toward reducing the polyp miss rate, many computer-aided techniques were proposed for objectively measuring the quality of the colon examination during colonoscopy. These techniques focus on detection of the appearance of polyps [11], measurement of the time taken to examine the colon mucosa while the endoscope was withdrawn [12] [13], and detection of the appearance of instruments used for tissue-sampling and therapeutic operations [3], just to name a few. Automatic detection of polyps and some other types of objects is desirable for objective quality documentation and computer-assisted colonoscopy to achieve an optimal quality inspection. In 2006, a set of guidelines for a good quality colonoscopy procedure was recommended [2]. An endoscopist should spend a minimum of *6* minutes during the withdrawal phase, reach the cecum at least *95%* of the colonoscopy procedures performed by the endoscopist, and detect polyps in 25% of his/her male patients and *15%* of his/her female patients. An operation suggested as an essential part of the examination of the colon to improve polyp yields is rectal ***Retroflexion*** [14] [15] – an endoscope maneuver where the tip of a flexible endoscope is deflected more than *90* degrees from the axial direction of the shaft of the endoscope. This results in visualization of intestinal mucosa along the shaft of the endoscope. There are many techniques proposed for detecting polyp appearance in colonoscopic images. However, existing polyp detection techniques either take a long time (*3* to *4* seconds) to analyze an image or detect too many false alarm regions (at least *1* false region per image) [16] [17] [18] [19] [20] [21] [22] [23] [24] [25] [26] [27] [28] [29] [30] [31]. None of the existing automatic polyp detection techniques is suitable for practical use. The only existing technique for detection of an image showing an appendiceal orifice is quite slow (about *5* minutes to analyze one image) [32]. There are no other computer-aided techniques for detecting retroflexion operations so far. Given a large number of colonoscopy procedures performed annually (over *14* million procedures in the United States) [33], an effective system for automatic detection of these objects in terms of high detection rates and fast analysis time is highly desirable to assist the endoscopist during the procedure to avoid missing polyps.

Similar to the existing computer-aided techniques, we face many challenging issues for effectively detecting the appearance of these objects in colonoscopy video. These challenges are the noisy nature of endoscopic images caused by camera motion, strong light reflection, the wide angle lens that cannot automatically be focused, the location and appearance variations of objects within the colon, and that the analysis is too slow to output analysis results of the current frame before the next video frame is seen by the endoscopist (i.e., the analysis time is more than *33* milliseconds for *30* frames per second video capturing rate). We aim to develop new image/video analysis techniques to solve these challenging issues.

In this dissertation, we aim to detect three types of representative objects: appendiceal orifice, endoscope, and polyp. We investigate a local appearance model – ***edge cross-section profile*** (ECSP) as a core model in our detection techniques. Edge cross-section profile and its first-order derivative (called gradient profile) are well known local appearance models for object detection (e.g., human face detection) [34] [35] [36] and tracking [37] in both medical imaging and non-medical fields. The ECSP models the local appearance of pixel values along the directions perpendicular to edge pixels. Figure 1.2 (a-b) shows an example of marked edge cross-section (white color bar) and an associated region of interest (ROI) on a polyp in colonoscopy. Figure 1.2 (c) shows the corresponding ECSP. Due to the polyp protrusion, the intensity values on ECSP of polyps increase abruptly near the edge (around pixel positions 40-60 in the Pixel#-axis in Figure 1.2 (c)) and increase at a much slower pace afterward. This is also true for sessile polyps with very little protrusion. However, most non-polyp objects do not have this pattern. Besides polyps, other commonly seen types of objects in colonoscopy also show discriminative patterns of ECSP and its derivative profiles. We will demonstrate these patterns in Chapter 3.

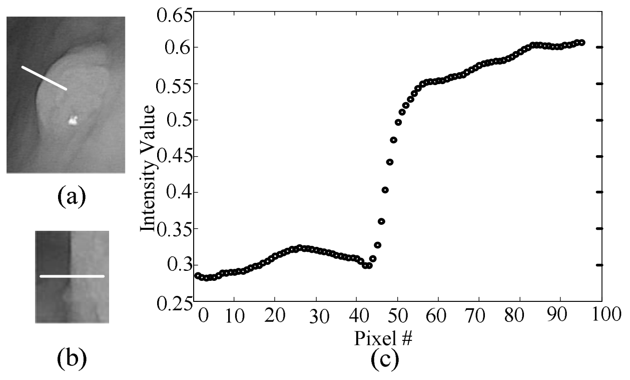


Figure 1.2: An example edge cross-section and its profile function of a polyp edge. (a) An edge cross-section of an edge pixel on a polyp marked as a white bar. (b) The corresponding region of interest extracted from (a). (c) ECSP obtained from (b). The intensity values are rescaled from [0,255] to [0,1].

In summary, our motivations to investigate ECSP for detecting these colonoscopic objects are as follows. 1) *Model robustness*: As a local appearance model, ECSP and its derivative profiles are robust to variations in object appearances, such as variations caused by the rotation and translation of the capturing camera. 2) *Pattern discrimination*: ECSP, its 1st – order and its 2nd – order derivative functions show different patterns for different objects in images captured from colonoscopy videos. The modeling of these patterns on ECSP functions can potentially be used to detect colonoscopic objects. 3) *Analysis time*: Comparing with other features such as region texture and color, extraction of edge-based features involved fewer pixels in the calculation. Therefore, it has potential to achieve a fast analysis time, such as real-time detection during live colonoscopy.

1.3 Contributions of Dissertation

Previous work using ECSP or its gradient profile for object detection/tracking either model the distribution of the value of each point on a profile by multivariate functions [34] [35] [36] [37], or fit the profile shape to pre-defined functions [38] [39] [40] [41]. This dissertation proposes the first feature-based ECSP techniques including three different approaches to extract features for detection of the three aforementioned object types. In the first approach, we present our ECSP calculation method which is invariant to object scale in image and extract ECSP features from key positions on ECSP and its gradient profile. In the second approach, we develop an edgeless ECSP that calculates ECSP and obtains features on ECSP without using edge detectors. In the third approach, we segment ECSP, its gradient profile and its 2nd – order derivative profile into non-overlapping parts. We model and extract features from each segmented part using the method which is suitable for that part.

Based on our proposed feature-based ECSP approaches, we developed several algorithms for detecting three representative types of objects in colonoscopic images including an anatomical landmark – appendiceal orifice, an out-of-patient object used in a retroflexion operation – endoscope, and an abnormality – polyp. Furthermore, we developed algorithms for extracting the semantic units of these types of objects including appendix video – the video with sufficient inspection time (at least

3 seconds) on appendiceal orifice, retroflexion video – the video containing at least one frame showing retroflexion operation, polyp sub-shot – a group of consecutive in focus frames showing polyps, and polyp shot – a sequence of consecutive nearby detected polyp sub-shots including the non-polyp images between each two sub-shots.

We developed a software package for evaluating these techniques in a clinical trial. Our proposed appendix image and video detection algorithms can effectively detect appearance of appendiceal orifice in image and appendix video tested on *23* endoscopy procedures. The average sensitivity and specificity for the detection of appendiceal orifice images with the often seen crescent appendiceal orifice shape are *96.86%* and *90.47%*, respectively. The average accuracy for the detection of appendix videos is *91.30%.* The average analysis time for detecting the quality visualization of appendiceal orifice in a colonoscopic procedure was *38* minutes and *17* seconds. The detection time is fast enough to confirm the orifice visualization quality to the patient before he/she leaves the endoscopic screening facility.

Our retroflexion detection algorithm tested on *50* colonoscopy videos gave a *100%* specificity, *100.0%* precision, and *78.0%* accuracy running on the last *45* seconds of each video, and a *72.7%* specificity, *91.7%* precision, and *82.0%* accuracy running on the entire withdrawal phases of each video for detecting retroflexion videos. Our retroflexion detection technique is the first technique that can detect the appearance of retroflexion. The average execution time of our MATLAB analysis code is *0.46* seconds per image on a modern PC (e.g., Intel Duo Core processors). The system can run in near real-time to assist the endoscopist during live colonoscopy using the frame rate of *2* fps. Our retroflexion detection system is promising for clinical practice in terms of both of the detection rate and analysis time.

We implemented our polyp image detection and shot extraction algorithms using C/C++ and Open Source Computer Vision Library. Our polyp detection system can correctly detect *97.7%* (*42* of *43*) of polyp shots in near real time under the frame analysis rate of *10* fps tested on *53* randomly selected video files captured from two endoscopy brands – FUJINON and OLYMPUS during routine screening colonoscopy. The average number of false shots per video is *36.2* and the average duration of each of these false shots is less than *1.5* seconds. Our system can run in near real-time to assist the endoscopist by providing visual feedback of a detected polyp as a warning signal on monitor during routine screening colonoscopy. To our best knowledge, our system is the first system that can detect polyp image and polyp shot in near real-time running on a modern PC. Our polyp detection and shot extraction system is promising for clinical use in terms of both of the detection rate and analysis time.

Incorporating our retroflexion detection and polyp detection system into clinical practice for routine screening colonoscopy has potential to reduce the polyp miss rate. The system can be used during training of new endoscopists. The future use of our appendix detection system for documenting the quality of colonoscopy may also contribute to the decline in the polyp miss rate. The automatic documentations of detected semantic units of these three colonoscopic object types can be helpful to discover unknown patterns of colorectal cancers or new diseases and used as educational resources for endoscopic research. In future work, our ECSP based techniques can be extended to both detect other types of colonoscopic objects such as diverticula, lumen and vessel, and analyze other endoscopy procedures, such as laparoscopy, upper gastrointestinal endoscopy, wireless capsule endoscopy and EGD.

1.4 Content Guide

The remainder of this dissertation contains the following content.

Chapter 2 introduces the current state-of-the-art of relevant work on object detection, shot segmentation and ECSP.

Chapter 3 presents the formulation and our calculation method of multi-derivative ECSP.

Chapter 4 proposes ECSP feature extraction method, and ECSP features for detecting images showing appearance of appendiceal orifice and appendix videos.

Chapter 5 introduces our edgeless ECSP method that extracts ECSP features without extracting edges. We combine features obtained from edgeless ECSP, region shape and location for retroflexion detection.

Chapter 6 presents a part-based ECSP by segmenting ECSP functions into non-overlapping parts. We investigate features on parts for polyp detection. We present a near real-time detection system for warning endoscopists of polyp appearance in live colonoscopy for clinical use.

Finally, we conclude our work and present the future work of ECSP and the improvement for detecting the three representative object types in Chapter 7.

CHAPTER 2 RELATED WORK

In this chapter, we briefly summarize the current state-of-the-art research for detection of the three aforementioned types of objects in colonoscopy, shot segmentation, and the related work on edge cross-section profile for object detection.

2.1 Object Detection in Colonoscopy

Object detection and recognition are common research topics in computer vision and pattern recognition areas. Low-level features (e.g., pixels color, region texture, and geometric shape information) or automatically detected salient features together with machine learning methods have been widely used for object detection [42] [43] [44] [45] [46] [47] [48]. Detection of objects in medical images such as polyps, blood and ulcers has been investigated in [16] [17] [19] [22] [30] [49] [50]. In this section, we briefly review the current state-of-the-art research on detection of the three representative types of colonoscopic objects – appendix, retroflexion and polyp.

2.1.1 Appendix Detection

Appendix image detection problem was previously introduced and addressed in [32]. The technique uses Random Hough Transform (RHT) to fit the appendiceal orifice shape to an ideal ellipse shape and extracts features based on the shape. *This technique was the only technique that addresses the appendiceal orifice detection problem.* The technique works as follows. First, a qualified curvilinear structure and the skeleton of the structure are identified. Ideal ellipses are estimated from the skeletons using RHT. Only qualified ellipses are kept for feature extraction. The important features include 1) ratio of edge pixels that are part of qualified curvilinear structures to the total number of edge pixels in the image; 2) sum of fractions of areas in the image that are part of the detected ellipses; and 3) sum of fractions of edge pixels in the image that are on the boundaries of the detected ellipses. A classifier was used to determine whether the image is an appendiceal orifice image. The detection performance evaluated on six videos was good for the appendiceal orifice with multiple curvilinear structures. However, the recognition time for this technique is long (about 5 minutes per image using MATLAB on an Intel Xeon 1.86 GHz processor), making it difficult to improve for real-time quality control in the future. The long recognition time is due to a large number of iterations of RHT to get good ellipse approximation. Thus far, there is no technique proposed for detecting the appendix video (video showing at least *3* seconds of appendiceal orifice).

2.1.2 Retroflexion Detection

To our best knowledge, there is no existing literature on detection of the endoscope appearance in colonoscopic images. The closest work is the detection of biopsy forceps and diathermy snare [3]. The algorithm segments an image into non-overlapping regions. The light reflected connected pixels which have a linear curvature in the segmented regions are used to estimate the insertion direction of the instrument. The instrument regions along the pre-defined insertion direction were detected by matching with pre-defined shape templates after region filtering and merging. The instrument shots in a video were identified by a shot detection step. The major drawback is the execution time of the algorithm since it relies on an image segmentation method called JSEG [51] and a hierarchy clustering for insertion direction estimation. In our experiments, processing of a *720-by-480* resolution image took *22* seconds tested on the same machine described in Chapter 5. These instruments are rigid objects with bright blue or red color. The technique in [3] relies on the linearity of the bright regions and predefined region templates to detect these instruments. However, different from instruments, the endoscope does not have a rigid body. We cannot simply match the shape of the endoscope to a predefined template or use color information alone to detect the endoscope since it is similar to the lumen seen from a distance.

2.1.3 Polyp Detection

Existing computer-aided polyp detection techniques can be grouped into four categories based on acquisition modality: 1) CT images from 3D computed tomography [7] [52] [53], 2) images from wireless capsule endoscopes [54], 3) colonoscopic images from optical colonoscopy using standard endoscopes [11], and 4) colonoscopic images using magnifying endoscopes or Narrow-Band Imaging (NBI) [55] [56]. We focus on the detection of polyp appearance in colonoscopic images from optical colonoscopy using standard endoscopes in routine day-to-day practice. Pit patterns [56] of polyps are not necessarily seen using these scopes.

Recent techniques for polyp detection in colonoscopic images from optical colonoscopy employ machine learning classifiers on one of the following major feature categories: 1) geometric shape of edges [16], 2) region color [22], 3) region texture [30], and 4) the combinations of color, texture and edges, such as the combination of region color and geometric shape of edges [19] or the combination of region color and Local Binary Pattern (LBP) texture [17] [50]. Most of polyp detection techniques using the features in the aforementioned four categories are summarized in [11]. We summarize these techniques in Table 2.1, which is an extended version from Table 1 in [11].

Table 2.1: Overview of Polyp Detection Techniques
(This table is an extended version from Table 1 in [11])

Authors	Year	Method	Feature Class	Classifier	Datasets (#images, image resolution)	Literature
Krishnan et al.	1998	Curvature analysis	Shape	-	(2 normal and 4 abnormal images, -)	[18]
Karkanis et al.	2001	Co-occurrence matrix, Wavelet transform	Texture	NN	(8 images, 512 x 512 pixels)	[26]
Wang et al.	2001	Local Binary Pattern	Texture	NN	(3 images, -)	[21]
Magoulas et al.	2001	Co-occurrence matrix	Texture	NN	-	[28]
Kang et al.	2003	Area, color and shape of segments	Shape, color	-	-	[20]
Karkanis et al.	2003	Color Wavelet Covariance	Texture, color	LDA, SVM	(1,380 images, 1000 x 1000 pixels)	[22]
Tjoa et al.	2003	Texture spectrum histogram, color histogram	Texture, color	SVM	(12 normal and 54 abnormal images, -)	[25]
Magoulas et al.	2004	Co-occurrence matrix	Texture	NN	-	[27] [29]
Li et al.	2005	Wavelet coefficients, histogram of CIE-Lab color space	Texture, color	SVM	(12 normal and 46 abnormal images, 256 x 256 pixels)	[24]
Dhandra et al.	2006	Watershed region segmentation in HIS color plane	Shape, color	-	(50 normal and 50 abnormal images, -)	[19]
Iakovidis et al.	2006	Local Binary Pattern	Texture	SVM	(15,000 images, 320 x 240 pixels)	[17]
		Wavelet Energy	Texture			
		Opponent Color – Local Binary Pattern	Texture, color			
		Color Wavelet Energy	Texture, color			
		Wavelet Correlation Signatures	Texture, color			
		Color Wavelet Covariance	Texture, color			

NN = Neural Network, SVM = Support Vector Machine, LDA = Linear Discriminant Analysis

Table 2.1: Overview of Polyp Detection Techniques (Continued)
(This table is an extended version from Table 1 in [11])

Authors	Year	Method	Feature Class	Classifier	Datasets (#images, image resolution)	Literature
Hwang et al.	2007	Curve direction, curvature, Watershed region segmentation, RGB pixel color	Shape, color	-	(815 polyp frames and 7,806 normal f-rames, 390 x 370 pixels)	[16]
Alexandre et al.	2007	RGB pixel color	Color	SVM	(35 images, 514 x 469 pixels)	[23]
Cheng et al.	2008	Co-occurrence matrix, color texture	Texture , color	SVM	(37 training and 37 testing images, 378 x 254 pixels)	[30]
Ameling et al.	2009	Co-occurrence matrix	Texture	SVM	(1,736 images, 1920 x 1080 pixels)	[31]
		Local Binary Pattern	Texture			
		Opponent Color – Local Binary Pattern	Texture , color			

NN = Neural Network, SVM = Support Vector Machine, LDA = Linear Discriminant Analysis

Geometric shape: In [18], the edge curvature is calculated as the feature of the geometric shape of an edge. The algorithm is designed to select the edges with “S” like shape. The edge with “S” like shape generally has a large change of curvature values along the tracked edge pixels. However, this algorithm can only be applied to detect polyps connected with lumen fold with “S” like shape. Many polyps do not have this particular type of shape. The algorithm was only evaluated by testing *2* normal and *4* abnormal images.

Geometric shape and color: In [19], watershed segmentation algorithm is applied to segment an input image in HIS color space. A regional maxima morphological operation is applied to select regions. The recent technique in [16] extracts features of the geometric shape of edges rely on shape approximation of polyp edges to the ideal shape template. The technique is consisted of two major steps – candidate edge calculation and template matching. For candidate edge calculation, a gradient image is calculated by selected the maximum gradient value among three RGB color channels for each image pixel. A marker-controlled watershed segmentation algorithm is used to segment the gradient image. A regional maxima morphological operation is applied to select the initial seeds as markers for watershed segmentation. That is, the pixels which have large local intensity values are used as initial seeds. The segmented region boundaries are considered as candidate polyp edges. In template matching, ellipse and parabola are considered as template shape models. The polyp edges are determined based on coefficients of template models that fit the edges using the least squares fitting method. The technique misses polyps that do not approximate well with the ideal shape template [16]. Furthermore, only partial edges of a polyp may be detected and may be insufficient to approximate the ideal shape template. Both techniques run slowly. The time-consuming watershed segmentation and morphological operations make it difficult to improve to achieve the real-time performance. In [20], a simple method is presented to pre-process an image by detecting Canny edges, dilating them into regions, and thresholding the regions based on the region size, color and shape to identify polyp regions. Although the paper claims that the technique does real-time detection, the algorithm can only process *1* image per second [20]. In addition, the technique in [20] is very roughly described without sufficient details and without any performance evaluation to support their claim. Therefore, the technique and results are not convincing. The technique is also questioned in [11].

Region color: In [23], an image is divided into non-overlapping blocks. The pixel color in RGB color plane and the coordinates of each pixel is concatenated into a high-dimensional feature vector per block. As a result, there are total *8000* features obtained for each block of *40 x 40* pixels, and this involves the calculation for the features of *132* blocks on each image with the resolution of *514 x 469* pixels. The calculation time of this method is expensive, although the analysis time is not clearly reported in the paper. Finally, SVM is used to classify block into polyp block and non-polyp block. The paper clarifies that they do not provide the criteria to consider an image as a polyp image. The evaluation of features is only based on classifying blocks using *2*-fold cross validation on *35* images.

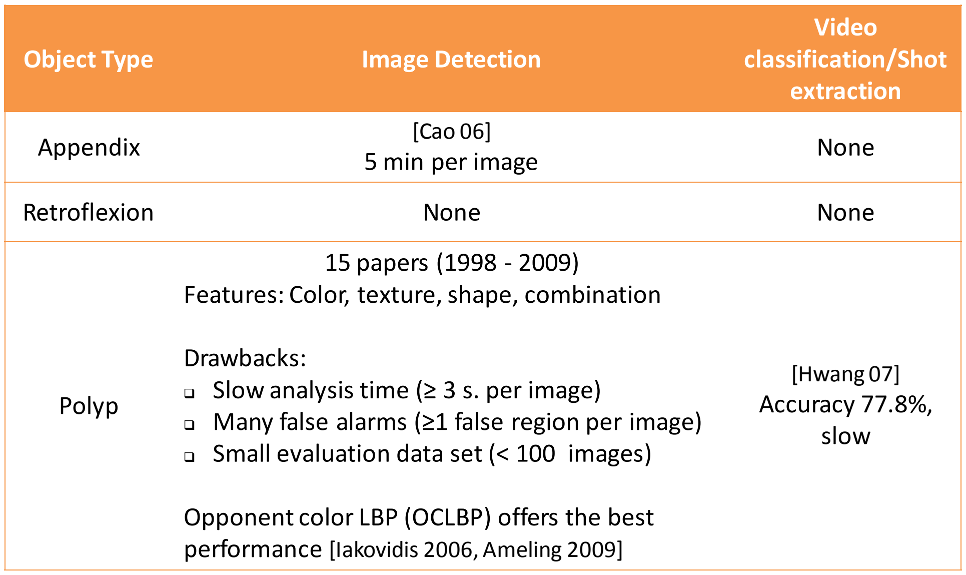
Region texture: Local Binary Pattern (LBP) is a popular texture feature for polyp detection [17] [31]. An LBP pattern of a pixel is represented by a binary sequence of length L. The l^{th} position in the binary sequence is assigned a *0* if the pixel value is smaller than the value of its l^{th} neighboring pixel, or assigned a *1* otherwise. An LBP pattern is mapped to a unique LBP value. A histogram of LBP values is used as an LBP feature of an image. Another region texture feature used in [17] is Wavelet Energy. Wavelet Energy feature encodes the content of the input image in variable width spatial frequency bands [17]. Given a grayscale image, after convoluting it with low pass and high pass filters, its wavelet textures are represented by sub-sampled images at different scales. These multi-scale images include both low-resolution and detail images. The Wavelet Energy feature is the sum squares of all wavelet coefficients that are obtained from each of these multi-scale images.

Region texture and color: An Opponent Color-LBP (OCLBP) feature [17] [31] is a concatenation of three intra-channel LBP histograms and six inter-channel LBP histograms. Each of the three intra-channel LBP histograms (C_i where the channel $i=1$, 2, and 3) is obtained from pixels in each of the RGB color channels, respectively. The six inter-channel LBP histograms (C_1-C_2, C_1-C_3, C_2-C_1, C_2-C_3, C_3-C_1, and C_3-C_2) are obtained from every combination of two different color channels. For example, the LBP pattern of C_1-C_2 is obtained from comparing the pixel value in the red color channel (C_1) with values of its neighboring pixels in the green color channel (C_2). Another combination of region texture and color feature is Color Wavelet Energy. Color Wavelet Energy feature is calculated by summing the squares of all wavelet coefficients obtained from all color channels of these multi-scale images [17]. Wavelet Correlation Signatures is an extend feature based on the Color Wavelet Energy by taking account into the correlations of wavelet coefficients between color channels [17]. Comparing to Color Wavelet Energy feature, Wavelet Correlation Signatures add the correlations of wavelet coefficients obtained from inter-channels (C_1-C_2, C_1-C_3, C_2-C_1, C_2-C_3, C_3-C_1, and C_3-C_2) as additional features. Another two texture features encoding spatial dependence of pixel color information are co-occurrence matrix [26] [27] [28] [29], and its combination with Wavelet Transform named Color Wavelet Covariance [22]. Co-occurrence matrix encodes the spatial dependence of pixel values by estimating the second-order joint conditional probability density function $f\ (i, j, d, a)$. The function $f\ (i, j, d, a)$ is calculated by counting all pairs of pixels i and j at distance d at a given direction a. The angular displacement a is usually in the range of the values $\{0, \pi/4, \pi/2, 3\pi/4\}$. Color Wavelet Covariance features are covariance estimation of different color channels overall four statistical features including angular second motion, correlation, inverse difference motion and entropy [22]. These statistical features are derived from co-occurrence matrices. Another set of combination of texture and color features are coefficients of Wavelet Transform and histogram bin values in CIE-Lab color space [24]. However, the paper [24] focuses on the investigation of a good classifier - ensemble SVM for feature classification on non-overlapping blocks in image instead of the investigation of the goodness of features. In [25], an alternative representation of the combined features was presented by histogram bins. The combined features are the bins values of texture spectrum histogram and color histogram. In [30], color features are derived from co-occurrence matrix as a feature vector. The extraction of color texture features took about *13* seconds per image at the resolution of *378 x 254* pixels on a PC with a 1.83GHz Intel Centrino Duo CPU and 2GB RAM. The technique was evaluated on *37* training images and *37* testing images.

In 2006, most features from the categories of region color, region texture and the combination of region texture and color were compared in [17], including LBP, OCLBP, Wavelet Energy, Color Wavelet Energy, Wavelet Correlation Signatures and Color Wavelet Covariance features. Among above compared features, OCLBP offered the best performance [17]. The test set has *15,000* images with the resolution of *320 x 240* pixels.

In 2009, co-occurrence matrix, LBP and OCLBP were compared in [31]. Among above compared features, OCLBP also offered the best performance presented in [31]. The test set has *1,736* images with the resolution of *1920 x 1080* pixels obtained from a high definition capturing device.

As also indicated in [11], most newer techniques (except technique [16]) developed since the comparison made in [17] and [31] were either evaluated on a data set that was too small (less than 100 combined normal and abnormal images) to clearly demonstrate the effectiveness of these techniques, or took a long time to analyze an image. Figure 2.1 summarizes existing work related to automatic detection of the three representative types of objects.



Object Type	Image Detection	Video classification/Shot extraction
Appendix	[Cao 06] 5 min per image	None
Retroflexion	None	None
Polyp	15 papers (1998 - 2009) Features: Color, texture, shape, combination Drawbacks: ❑ Slow analysis time (≥ 3 s. per image) ❑ Many false alarms (≥1 false region per image) ❑ Small evaluation data set (< 100 images) Opponent color LBP (OCLBP) offers the best performance [Iakovidis 2006, Ameling 2009]	[Hwang 07] Accuracy 77.8%, slow

Figure 2.1: Summary of the related work of automatic detection of the three representative object types

2.2 Shot Segmentation

A shot consists of sequential frames with similar visual properties. Shot segmentation cuts a long video file into smaller shots. The criterion for grouping frames into a shot is typically based on low-level image features (e.g. color or intensity histograms, changes in edges) and temporal information [57] [58] [59] [60] [61] [62] [63] [64] [65] [66] [67] [68] [69].

Shot segmentation techniques have been proposed for uncompressed video and compressed MPEG video [62]. The majority of shot segmentation algorithms are designed to process uncompressed videos as low level features such as pixels in decoded frames contain more content information. Shot segmentation on compressed MPEG video reduces computation time since video decoding is not performed. We focus on processing uncompressed video extracted from compressed MPEG video in our research.

The key to segment an uncompressed video into shots is to find either cuts [62] or gradual transitions [62] of shots in the video. Shot segmentation algorithms in uncompressed video can be categorized into three major categories: 1) thresholding-based algorithms, 2) clustering-based algorithms and 3) model-based algorithms.

For thresholding-based algorithms, a similarity score is calculated between two consecutive frames to measure the similarity of the frames. A frame is detected as a cut if the similarity score between this frame and its previous frame is lower than a pre-defined threshold. The similarity score is generally obtained from low-level image features such as pixel intensity values, average intensity values of block-based regions, and color histogram. Pixel comparison methods compute the difference of intensity value of each pixel between two images [63]. The average difference of intensity values over all pixels is calculated as a similarity score. A cut is detected if the similarity score is larger than a threshold. Many alternative approaches can be used to compute the similarity score. A popular approach sets a score to *1* if the intensity difference for a corresponding pixel between two frames is large than a pre-defined threshold, or to *0* otherwise. The average score of all pixels is used as the similarity score. Pixel comparison methods can be extended to process a color image by computing the similarity score of pixels in each color channel separately. These algorithms are the simplest to obtain the similarity between frames. However, they are usually sensitive to camera motion and light condition. For example, a small translation of camera or a small change in illumination could affect the vast majority of pixels. Therefore, a single threshold is not robust to determine a shot cut. Block-based comparison methods [64] [65] divide an image into overlapping or non-overlapping blocks and calculate similarity scores among corresponding blocks. The average score of these similarity scores of corresponding blocks between two images is used to determine a shot cut. Comparing to the pixel comparison, the block-based comparison considers the global content between images and it is more robust to the small camera or object motion. However, the block-based comparison methods are sensitivity to large motion of camera or objects. Histogram comparison computes the distribution of intensity values represented by histogram bins. The differences of histogram bin values between two frames are used as a similarity score. The global histogram comparison computes the intensity value distribution of the entire image, while the local histogram comparison computes the intensity value distribution in a local region such as a block. Histogram comparison is less sensitive to motion of camera or objects.

Clustering-based algorithms [66] use unsupervised learning algorithms to cluster frames into shots. Many features such as intensity or color histogram can be used together with unsupervised learning models such as K-means to cluster frame similarities. Clustering-based algorithms overcome shortcoming of thresholding-based algorithms which are sensitive to the types of videos. For example, the thresholds for different types of videos are different. However, they do not perform well for detecting gradual transitions [62].

Model-based algorithms either define or train underlying mathematical and statistical models for classifying consecutive video frames into shot frames and non-shot frames. Some example models are a mathematical model in [67], the model of probabilistic distribution of pixel intensity values in [68], and the hidden Markov model for modeling states of frames in shots [69]. Thresholds are not required in the model-based algorithms. However, these models need to be pre-trained and many shots may not have an obvious pattern to obtain a good model for some types of videos.

Shot segmentation techniques for colonoscopic video were presented in [70] and [4]. In [70], a colonoscopic video is segmented into shots using the global histogram comparison in the threshold-based approach. The technique in [4] extracts shots containing the appearance of instrument in colonoscopic operation – operation shot. The paper proposed an algorithm to detect images showing appearance of instrument as instrument image and count the detected instrument images in a time window. If a time window contains any detected instrument image, the window is considered as a window with instrument pattern. A sequence consisted of such consecutive windows with instrument pattern longer than *4* seconds are extracted as operation shots.

Traditional shot detection techniques are not suitable for colonoscopy video because of the following reasons. 1) Color information is not as useful since colonoscopic images have limited color ranges of mostly red; with stool, the red color is mixed with green or yellow. 2) There is no significant movement of a specific object of interest in the image, but small movement of the camera can cause rotation, translation, and scaling of the whole image. 3) Strong light reflection, low illumination and blurriness cause significant changes in low-level features, which makes it difficult to group them into one shot. Some robust shot segmentation methods that account for monotone image pixel color property and the effect of camera motion such as strong light reflection and partially blurred images are desirable for segmenting colonoscopy video.

2.3 Edge Cross-Section Profile

In non-medical fields, edge cross-section profile (ECSP) and its first-order derivative (called gradient profile) are well known local appearance models for object detection (e.g., human face detection) [34] [35] [36] and tracking [37].

In the medical imaging field, ECSP is used for blood vessel extraction in retinal images [38] [39] [40], and skin cancer detection [41]. Note that the ECSP profile calculation of the existing work was not explicitly mentioned in the reference and may be slightly different from our ECSP calculation. Furthermore, none of the calculated ECSP in the existing work is invariant to the object scales in image and no investigations of 2nd – order derivative ECSP function and region of interest surrounding the edge were presented.

We group existing object detection techniques using ECSP into the following two categories.

(1) Modeling ECSP with multivariate distributions [34] [35] [36]: Each point on an ECSP is treated as a random variable. The mean and variance-covariance matrices of all random variables are calculated and potential correlations among these variables are exploited. Active Shape Model (ASM) [34] [35] models the gradient profile of ECSP using one of generative probabilistic frameworks – multivariate Gaussian distribution. However, this framework is unreliable when some points have multi-modal distributions. To address this issue, *k-means* clustering is used to preprocess the gradient profiles during training and a separate model is obtained for each cluster [36]. These underlying multivariate probability models are not robust to large variations in sizes of edges and corresponding ECSP.

(2) Modeling ECSP with pre-defined functions [38] [39] [40] [41]: Multi-scale convolution

functions or regression functions are used as shape models. In [38], multi-scale 2D Gaussian-like matched filters are used as convolution functions for extracting blood vessels in retinal images. In [39] [40], a regression function defined as a linear combination of several inverse Gaussian functions is used. To handle different scales, optimal model parameters are obtained by minimizing non-linear fitting errors. However, ECSP shapes of colonoscopic objects do not strictly follow the shapes of pre-defined functions. Modeling the entire shape of an ECSP is not sufficient to capture the local properties of the object.

In this dissertation, we present our own ECSP calculation method that is invariant to the object scales in image and propose three new approaches using our proposed multi-derivative ECSP for colonoscopic object detection. In the first approach, we extract features at key positions from ECSP and its first-order derivative profile for appendiceal orifice detection. In the second approach, we propose an edgeless ECSP technique that obtains ECSP and its features without detecting edges. The edgeless ECSP is useful for detecting objects without clear edges due to the noisy nature of colonoscopic images such as image blur or low illumination. In the third approach, we propose a part-based ECSP method by segmenting ECSP, its 1st – order and its 2nd – order derivative profiles into non-overlapping parts. The segmentation of ECSP gives flexibility for modeling each part or extracting features for a part using a suitable method for that part. Based on our experiments on real colonoscopy data sets, our ECSP features are robust to camera motion and the location and appearance variations of objects.

CHAPTER 3 EDGE CROSS-SECTION PROFILE

This chapter introduces our algorithm for calculating ECSP of an edge pixel, ECSP of an edge, the multi-derivative ECSP functions, and its corresponding region of interest (ROI). We first present the descriptor ECSP and its ROI. We then present a profile extractor for the calculation method of ECSP and its ROI.

3.1 Profile Descriptor

ECSP function for an edge pixel: Let $I(\overrightarrow{u_i})$ be the intensity value of an edge pixel $\overrightarrow{u_i}$ for $i \in 1, \dots, M$, where M is the total number of pixels on this edge. The vector $\overrightarrow{u_i}$ denotes the pixel at the coordinate (x_i, y_i) on a 2D image. We define a unit vector $\overrightarrow{v_i}$ as an *edge tangent* passing $\overrightarrow{u_i}$ and a unit vector $\overrightarrow{n_i}$ as an *edge normal* perpendicular to $\overrightarrow{v_i}$ pointing to the concave side of the edge. See Figure 3.1. The ECSP function for the edge pixel $\overrightarrow{u_i}$ is defined as

$$f_i(r) = I(\overrightarrow{u_i} + r \cdot \overrightarrow{n_i}), r \in \left[-\frac{\lambda}{\kappa}, \frac{\lambda}{\kappa}\right], r \in \mathbb{Z}, \tag{3.1}$$

where *r* is the pixel distance (called radius) in an integer set $\mathbb{Z}$ from the edge pixel $\overrightarrow{u_i}$ along its edge normal $\overrightarrow{n_i}$. The maximum value of *r* is inversely proportional to the edge curvature κ and scaled by a positive constant λ. This maximum radius *r* value depends on edge curvature. Therefore, object scales are reflected by radius *r* on ECSP functions. That is a large radius reflects a large size object relative to the image size and a small radius reflects a small size object relative to the image size.

ECSP region of interest (ROI): Let a row vector $\overrightarrow{f_i}$ represent all points on function $f_i(r)$ starting from $r = -\lambda/\kappa$ to $r = \lambda/\kappa$. We define a matrix

$$ROI = \begin{bmatrix} \overrightarrow{f_1} \\ \dots \\ \overrightarrow{f_M} \end{bmatrix}.$$

Figure 3.1 (b) shows the plot of the intensity values in matrix *ROI* obtained from the corresponding edge in Figure 3.1 (a). The matrix *ROI* is divided into two equal size sub-matrices: the left-half sub-matrix ROI_{convex} and the right-half sub-matrix $ROI_{concave}$. ROI_{convex} consists of pixels taken from the outer side of the edge/contour and $ROI_{concave}$ consists of pixels taken from the inner side of the edge. We call the matrix *ROI* as the region of interest of ECSP.

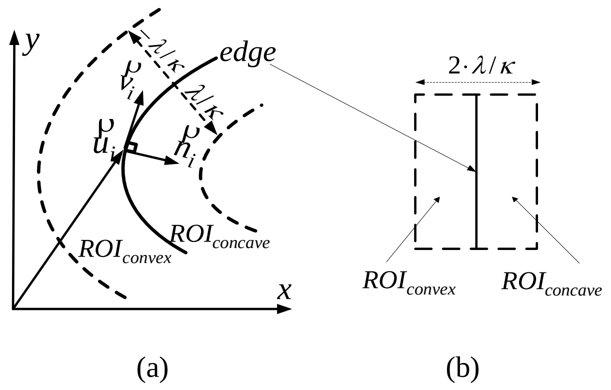


Figure 3.1: ECSP extraction of an edge in (a) to the corresponding ROI in (b). The vector $\vec{\boldsymbol{u}_i}$ indicates the position of an edge pixel with its tangent vector $\vec{\boldsymbol{v}_i}$ and normal vector $\vec{\boldsymbol{n}_i}$. The edge has its surrounding region of interest labeled as $ROI_{concave}$ and ROI_{convex} with the dashed lines indicating the edge pixels as region boundaries.

ECSP function for an entire edge: We define the ECSP function for an entire edge *j* as

$$F_j(r) = median\big(f_1(r), \dots, f_M(r)\big), r \in \left[-\frac{\lambda}{\kappa}, \frac{\lambda}{\kappa}\right], r \in \mathbb{Z}. \tag{3.2}$$

A median value for the same *r* values on these functions is selected because of its robustness to speckle and specular noises caused by light reflection.

Multi-derivative ECSP functions: The multi-derivative ECSP functions are represented as a set of functions $\{F(r), \nabla F(r), \nabla^2 F(r), \dots\}$, where each element $\nabla^k F(r)$ is the k^{th}-order derivative of the function $F(r)$. As the calculation of derivative function amplifies noise, the maximum order of derivative *K* needs to be empirically found in the training data. We select *K* based on the criteria that the patterns caused by noises and real object are discriminative on the derivative functions. To resist the noise, a function is smoothed using the moving average smoothing method [71] before the calculation of its derivative.

3.2 Profile Extractor

The key to extract ECSP is to obtain the edge normal $\vec{n_i}$ for edge pixel $\vec{u_i}$ and the edge curvature κ . We calculate edge normal and edge curvature as follows.

Calculation of edge normal $\vec{n_i}$: To calculate $\vec{n_i}$, we first obtain the edge tangent vector $\vec{v_i}$ by fitting all m_i edge pixels inside a square window D_i centered at $\vec{u_i}$ using the least squares linear regression model. We normalize x_i as $\hat{x_i} = (x_i - mean(x_{i_1,\ldots i_{m_i}}))/std(x_{i_1,\ldots i_{m_i}})$ where *mean* and *std* represent the mean and the standard deviation of x-coordinates of all pixels (denoted as$x_{i_1,\ldots i_{m_i}}$) in D_i, respectively. This normalization is important to correctly calculate $\vec{n_i}$ as it improves the numerical properties for the linear regression model [71] [72]. Using $\hat{x_i}$ as the predictor, and the corresponding y_i as the response in the regression model, we obtain its *1st* degree coefficient β_i. The edge tangent can be derived as $\vec{v_i} = [\frac{\beta_i}{\sqrt{1+\beta_i^2}}, \frac{1}{\sqrt{1+\beta_i^2}}]^T$. We finally obtain edge normal using $\vec{n_i} = R(\alpha) \cdot \vec{v_i}$, where the rotation matrix R uses the rotation angle α of either $-\pi/2$ or $\pi/2$ depending on which angle results in the calculated $\vec{n_i}$ pointing to the concave side of the edge. We will discuss our method to determinate the concave side of an edge in a later paragraph. The outlier norm vectors are detected and replaced by the median value of the norm vectors in a square window A_i. A norm vector is declared as an outlier if the angle difference between the norm vector and the median value is larger than a threshold $Thld_{degree}$. The value of $Thld_{degree}$ depends on the size of A_i. A larger A_i requires a larger value of $Thld_{degree}$. After this step, we obtain refined norm vectors of edge pixels pointing toward the concave side of the edge. Figure 3.2 shows some results of calculated edge normal for some types of objects.

Calculation of edge curvature κ: To calculate the curvature κ of an entire edge, we compute the median value of local curvatures for all M edge pixels, written as $\kappa = median\{\kappa_i\}_1^M$. Each local pixel curvature is represented by $\kappa_i = \theta_i/m_i$, where the angle θ_i is the maximum angle among every pair of edge normal vectors in the window D_i.

We use the following method to determinate the concave side and convex side of an edge. We compute the minimum bounding rectangle that encloses the edge and find two regions, $R_{concave}$ and

R_{convex}, separated by the edge as follows. We use region growing taking the center of the minimum bounding rectangle as the seed point to group pixels starting from the seed point. The resulting region is called $R_{concave}$. The rest of the pixels in the minimum bounding rectangle that do not belong to $R_{concave}$ are assigned to R_{convex}. As a result, for most edges, $R_{concave}$ is the region at the concave side of the edge and R_{convex} is the region at the convex side of the edge.

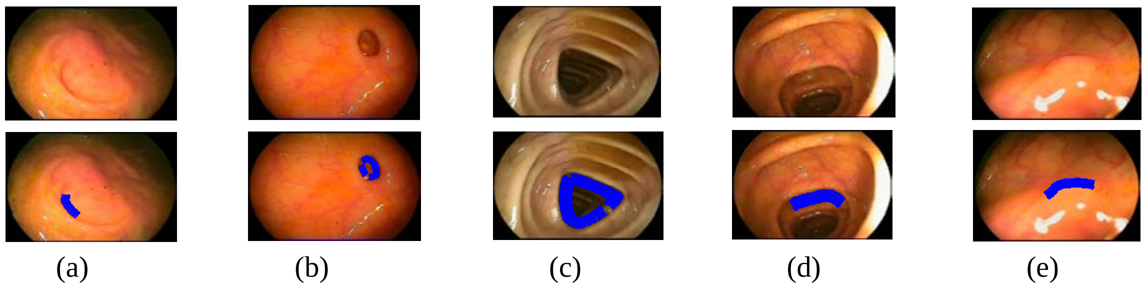


Figure 3.2: Top row: Original image; Bottom row: Image labeled with calculated edge normals (blue arrows). (a) Appendiceal orifice image; (b) Diverticulum image; (c) Colon lumen image with the lumen appearing as the dark region in the middle of the image; (d) Colon lumen image with the lumen appearing as the non-dark region in the middle of the image and darkest region at right side, and with nested multiple folds; (e) Flat polyp image

We plot multi-derivative ECSP functions of some commonly seen objects in Figure 3.3 including the three representative objects. The profiles of different objects show different patterns. For different objects whose ECSP functions appear similar, their function values are generally in different ranges (e.g., Figure 3.3 (d) and (e)).

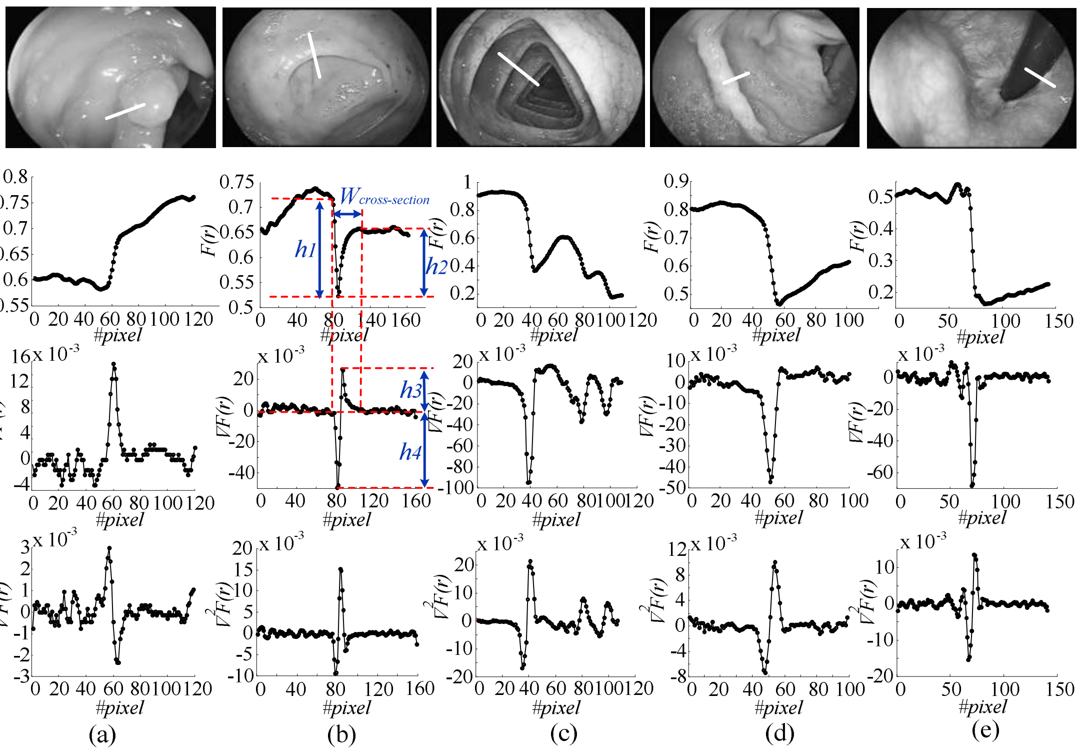


Figure 3.3: Multi-derivative ECSP of different types of colonoscopic objects. First row: Original gray scale image marked with a white bar representing the edge cross-section of an edge pixel; Second row: the corresponding intensity edge cross-section profile; Third row: the gradient profile; Fourth row: the second-order derivative edge profile. (a) Polyp image; (b) Appendiceal orifice image; (c) Lumen image with the lumen in the middle of the image and with multiple nested folds; (d) Colon fold; (e) Endoscope seen during retroflexion

CHAPTER 4 DETECTION OF APPENDIX IMAGE AND VIDEO

This chapter introduces our algorithms for detecting the quality visualization of appendiceal orifice including 1) the image showing the clearly seen appendiceal orifice and 2) the video showing at least three seconds of the appendiceal orifice inspection based on ECSP features.

4.1 Introduction

In 2006, a set of guidelines for a good quality colonoscopy was recommended [2]. This includes visualization of the appendieal orifice, which is one indicator that the cecum of the colon is reached during the procedure. The rationale for this recommendation is based on the fact that a substantial fraction of neoplasms are found in the proximal colon. Therefore, the appearance of the appendiceal orifice during colonoscopy indicates a complete traversal of the colon, which is an important quality indicator of the colon examination.

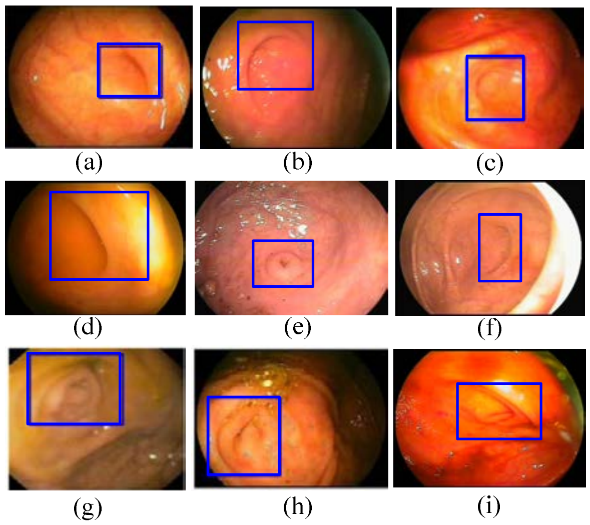


Figure 4.1: Variation in appendiceal orifice appearance; the appendiceal orifice is shown inside a rectangle: (a-f) a single crescent/circle slit seen at varying distance, (g-i) appendiceal orifice with multiple curvilinear structures with a sharp turn

The detection of the appearance of appendiceal orifice in image is very challenging for many reasons. The appendiceal orifice is often seen as a *crescent slit* because the appendix is folded around the cecum of the colon and is not maximally insufflated [73]. The shape of the appendiceal orifice may vary due to colon contractile activity or air insufflation during endoscopy. Furthermore, the appendiceal orifice can be seen at different locations, scales, view angles, and illumination. Figure 4.1 shows some examples illustrating these challenges. In addition, the orifice may be deformed or partially seen in an image or occluded; the appendiceal orifice may be partially or completely removed during appendectomy with little or no scar remaining; stool may occlude the appendiceal orifice; strong light reflected spots may cover parts of the orifice. Lastly, different types of endoscope brands have different image color and light source settings, creating image color and illumination variations.

In this chapter, we focus on identifying high quality visualization of the usual appearance of the appendiceal orifice in a clean normal colon (without appendectomy and without stools covering the orifice). A high quality video file test bed was created that contained for every colonoscopy at least *3* seconds of clear appendiceal orifice images and the location of the orifice was if possible the focus of the visualization (i.e., close to the center of the image). During the 3 second inspection of the appendiceal orifice, the endoscopist may not necessarily pause (stop moving the camera). Small camera movements, partial strong light reflection, poor illumination, and blur can occur in a few frames. To measure the quality of the visualization of the appendiceal orifice, we define an *appendix video* as a video that shows at least *3* seconds of appendiceal orifice images. If a colonoscopy video is determined to be an appendix video, this colonoscopy is considered of high quality in terms of visualization of the appendiceal orifice. Videos of procedures with low quality inspection should be reviewed by other endoscopists for quality control.

We present a hierarchical detection approach that determines whether a video is an appendix video. The approach consists of two-level analysis: image level and video level. At the image level, we propose an ECSP feature based appendiceal orifice image detection algorithm to detect appendiceal orifice images showing the commonly seen crescent slit of the appendiceal orifice. The crux of the algorithm is new local features that represent geometric shape, illumination difference and intensity changes of pixels along the norm direction (cross-section) of an edge. At the video level, we propose an appendiceal orifice shot detection algorithm that first detects near pause video segments of appendiceal orifice inspection of at least *3* seconds. If a sufficient number of images of such segments are found, the video is considered an appendix video. The hierarchical detection analysis at the image level and the video level together addresses the following challenges: 1) large scale and rotation variance due to different view angles and camera distances from the appendiceal orifice; 2) moderate illumination variance and strong light reflection; and 3) moderate amount of weak edges.

4.2 Edge Profile Based Appendiceal Orifice Image Detection

This section introduces our ECSP feature based algorithm for appendix image detection. Thresholds and parameter values used in our algorithms were obtained from experiments with our training data described in Section 4.4. We specify these values in the algorithm description and provide a summary of these values in Table 4.1.

4.2.1 Image Preprocessing

This step is to identify qualified edges in an image used as ECSP edges. We first extract edges by applying *Canny* edge detector (with a low threshold of 0.05, a high threshold of 0.1, and a standard deviation of 3) to the grayscale image converted from the input color image. Next, we apply a dilation operation twice on the edge image using a square shape structuring element (with a 5 pixel radius) to link broken edges (edge with edge pixels at a small distance from each other). Then, we extract the edge skeleton using the skeleton algorithm [74] and remove small branches on these edges using a spur operation with pruning [74]. Next, we discard the following noisy edges.

1) Edges with the number of edge pixels outside a pre-defined range *E* of between 100 and 500 pixels since these edges are either too short or too long for a typical appendiceal orifice edge for the image resolution used in our experiments.
2) Approximated linear edges (determined based on least squares linear fitting errors according to parameters in Table 4.1 no. 3) and edges with *Curvature* out of the range of commonly seen appendiceal orifice edges (between 1/20 and 1/320). The *Curvature* of a given edge is computed as the median of curvature of all pixels on this edge. Let W_i be a window of size 10x10 pixels centered at pixel *i* of the edge. The curvature of the pixel *i* is computed as the ratio of D_i to the length of the edge segment in W_i, where D_i is the absolute difference of the

tangent angles of the two endpoint pixels of the edge segment. The tangent angle of a pixel is the slope of the line fitting the edge segment in a window centered at that pixel. Approximated linear edges are usually those of colon folds, instruments, etc.

3) Edges caused by strong light reflection. A light-reflected pixel is normally very bright compared to the brightness of the entire image and has less pure color compared to non-light-reflected pixels. Figure 4.2 shows an example of the result of our light reflection detection algorithm that works as follows.

   Step 1: We build a saturation distribution of bright pixels (i.e., all pixels in the image with value (V) in HSV color space larger than a brightness threshold (V>0.4) and with saturation (S) larger than a saturation threshold (S>0.2)) using a Gaussian kernel function with the mean and the variance set to the mean and the variance of saturation of these pixels.

   Step 2: A pixel is considered a light reflected pixel if the following three conditions are satisfied: 1) The value (V) is larger than a brightness threshold (V>0.4), 2) the saturation (S) is smaller than the mean saturation obtained in Step 1, and 3) the probability of the saturation of this pixel on the saturation distribution modeled in Step 1 is smaller than a saturation threshold P_s (0.6).

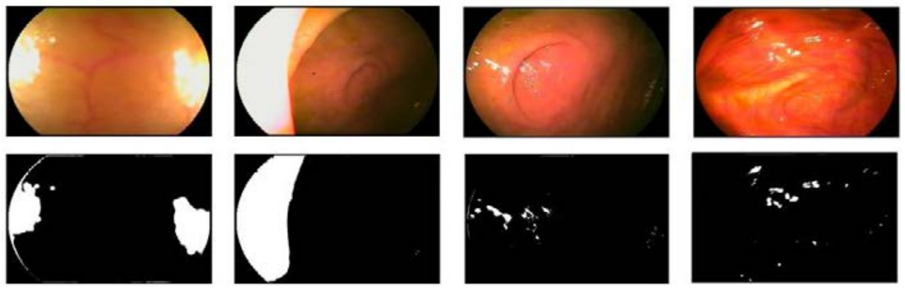
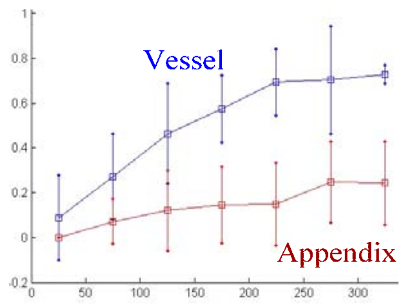


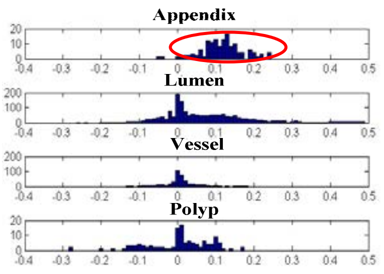
Figure 4.2: Results of light reflection detection. Top row: original images; Bottom row: detected light-reflected regions shown in white

4) Edges that do not fit well on ideal ellipses using least squares ellipse fitting. For each edge, we perform least squares ellipse fitting and a dilation operation on the detected ideal ellipse using a circle structuring element with a 3 pixel radius. The dilation is to account for some nearby edge pixels that do not lie perfectly on the ideal ellipse. Let N be the total number of pixels on the edge. We define $F_{ellipse}(N)$ as the percentage of edge pixels not covered by the dilated ideal ellipse. We reject edges in which $F_{ellipse}(N)>0.2$ for $N\leq 50$, $F_{ellipse}(N)>0.4$ for $50\leq N<100$, and

$F_{ellipse}(N)>0.7$ for $N\geq 100$. The criteria is derived from our finding that as N increases, the value of $F_{ellipse}(N)$ of vessel edges increases much faster than that of appendiceal orifice edges. Figure 4.3 (a) shows a plot of $F_{ellipse}(N)$ based on our training data set. Therefore, this method can discard most vessel edges and noise edges with connected branches or strange shapes (e.g. snake-like shapes or shapes with sharp corners).



(a)



(b)

Figure 4.3: Plot of edge features of selected representative images (137 appendiceal orifice edges, 1634 lumen edges, 424 vessel edges, 122 polyp edges) from 7 training colonoscopy videos. Left: Plot of the mean and variance of $F_{ellipse}(N)$ (ellipse fitting errors) and edge size (in pixels) N for vessel and appendiceal orifice edges; more errors are observed for vessel edges compared to appendiceal orifice edges. Right: Histogram of the strength of intensity drop feature obtained from the intensity edge cross-section profile.

4.2.2 *Key Point Based Features on Edge Profile*

We extract the features on the key positions of ECSP for classification as follows.

1) *Intensity of a concave region and intensity difference between concave and convex regions:* With the slit appearance of appendiceal orifice edges, we observe an intensity difference between the two sides of the slit. Furthermore, the concave side of an appendiceal orifice edge is typically brighter than that of lumen and diverticula, but less bright than that of polyps and instruments. We calculate the average intensity difference between the two regions as $\Delta\bar{I} = \bar{I}_{concave} - \bar{I}_{convex}$ where $\bar{I}_{convex}$ is the average intensity value on ROI_{convex}, and $\bar{I}_{concave}$ is the average intensity value on $ROI_{concave}$, respectively. For appendiceal orifice edges, $\Delta\bar{I}$ is often close to zero. For some other kinds of edges, for example, for edges of diverticulum and lumen, $\Delta\bar{I}$ is typically negative since they are

typically further away from the camera. For edges of polyps, $\Delta\bar{I}$ value is often high positive. This is because the polyp surface tends to reflect more light.

2) *Strength of intensity drop before crossing the edge and strength of intensity return after crossing the edge:* A plot of the intensity cross-section profile of an appendiceal orifice edge shows a very interesting pattern, different from most other edges in most cases. The intensity values decrease from the convex side of the slit to the lowest value at the middle position of the cross section and then increase to a similar level after that. Figure 4.4 (c) shows this pattern of a sudden drop by h_1 and then a sudden increase by h_2. The pixel positions to measure h_1 and h_2 should be the pixels at which the intensity change is very small just before and after the drop. These positions correspond to the positions when the first-order derivative of the intensity function is within a threshold $d_{first\text{-}order}$ (*0.001*) which is very close to zero. See Figure 4.4 (c) and (d). For different kinds of edges, the values of these features are different. From our investigation, we found appendiceal orifice edges usually have a large drop intensity value h_1, while for other kinds of edges the distributions of h_1 are different. See a distribution plot of this feature in Figure 4.3 (b). We found that for a diverticulum edge, the return intensity after the drop is usually noticeably lower than the intensity prior to the drop. Hence, the value of this feature is lower than that of the appendiceal orifice edge. For many colon folds near the colon lumen, the intensity after the drop does not rise back up. Polyp edges are usually brighter at the concave side. Hence, the return intensity surpasses the intensity level before the drop. We use a feature of normalized return-intensity-ratio $h_2/(h_1+h_2)$ to measure the strength of the intensity return after the drop.

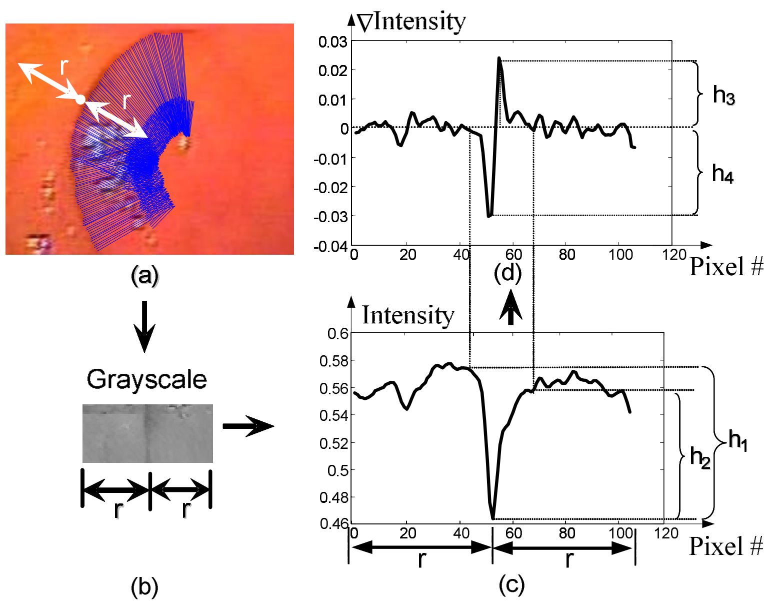


Figure 4.4**:** Cross-section profile of an edge of appendiceal orifice: (a) Appendiceal orifice edge with norm vectors extracted; (b) grayscale image of ROI around the edge (r=55) in image (a); (c) Plot of the intensity values of the intensity cross-section profile computed on image (b); (d) Plot of the first-order derivative values of the intensity cross-section profile in (c)

3) *Edge sharpness before and after crossing the edge:* To derive edge sharpness (how quick the intensity drops and returns) at the two sides of the edge, we first calculate the first-order derivative of the intensity cross-section profile. Let h_4 be the absolute difference between the minimum first-order derivative value and the first-order derivative value measured at the zero-crossing position where h_1 is measured; h_3 is the first maximum value on the first-order derivative curve after the middle position of the cross-section profile. See Figure 4.4 (d). We use the values h_3 and h_4 to measure the edge sharpness before and after the intensity drop and the ratio $h_3/(h_4+h_3)$ to measure the similarity of sharpness between the two sides of an edge. Figure 4.4 (a-d) illustrate these features for different types of edges. For appendiceal orifice edges, the sharpness between two edge sides are similar because the return speed of the intensity after crossing the edge is quite similar to the speed of the intensity drop. From our observation, the appendiceal orifice edges usually have a larger value of sharpness before crossing the edge (h_4) than other kinds of edges.

4) *Number of edge intersections along the norm direction:* We measure $N_{cross\text{-}section}$ as the number of local minimum pixels on the first-order derivative of its intensity cross-section profile in the range $r = \lambda/Curvature$ ($\lambda=1.0$ in this case) (Figure 4.4 (a)) where *Curvature* is the median curvature of all pixels on this edge. For each intensity cross-section profile, we want to count the number of edges nearby as a feature. Therefore, λ is set to *1*, the maximum value within its valid range (*0.5~1.0*) for the calculation of $N_{cross\text{-}section}$. If there are several local minimum pixels which are close to each other (within a short distance $d_{cross\text{-}section}$ of *20* pixels) without zero-crossing pixels among them, these local minimum pixels are counted as one edge intersection. For a single crescent slit of the appendiceal orifice, this feature has the value of one. The value of this feature is greater than one for edges of nested colon folds around the colon lumen. See Figure 3.3 (c).
5) *Edge Cross Section Width:* The width of the edge ($W_{cross\text{-}section}$) is the number of pixels between positions where h_1 and h_2 are measured (Figure 3.3 (b)). Empirically, we found that most of vessel edges have a smaller width than that of appendiceal orifice edges.

4.2.3 Appendiceal Orifice/Non-appendiceal Orifice Image Classification

We utilize the domain knowledge that the appendiceal orifice appears near the end of the insertion and only in *wall image*s---images without distant lumen. Images with the distant lumen in the field of view are called *lumen images*. We use our lumen/wall image classification method [13] to label clear images as either lumen images or wall images. Then, we detect the frame indicating the end of insertion using our previous method based on camera motion [12]. Finally, we only consider images around the detected end of insertion frame. In our study, it is sufficient to analyze one frame per second instead of analyzing the frames at the full video frame rate. We perform appendiceal orifice image detection only on the frames within a selected range EOI_{image} (4 minutes before and 8 minutes after the end of insertion frame) since appendiceal orifice images should appear around the end of insertion. The value of EOI_{image} is conservative based on the error rate of the method for the detection of the end of insertion frame. The selected range is to accommodate some inaccuracy of the end of insertion detection and the time for appendiceal orifice inspection. The inspection of appendiceal orifice is generally performed right after the end of the insertion time.

We use a two-layer hierarchical classification based on our extracted features as follows. In the first layer, we reject likely non-appendiceal orifice edges using a set of rules obtained from the J48 decision tree classifier of Weka [75]. This classifier is a variant of the well-known C4.5 decision tree algorithm [76]. The feature vector is $F_{tree}=\{h_1, h_2/(h_1+ h_2), \Delta\bar{I}, \bar{I}_{concave}, h_4, h_3/(h_3+ h_4), N_{cross\text{-}section,},$ $Var_{Curvature}\}$, where all features except the last one are discussed previously. $Var_{Curvature}$ is the variance of curvatures of all the pixels on the edge. The features, $h_1, h_2/(h_1+ h_2), \bar{I}_{concave}, h_3/(h_3+ h_4),$ play an important role in separating most of non-appendiceal orifice edges from appendiceal orifice edges with a very small mis-classification error for appendiceal orifice edges. We use a logic rule set $R =$ $\{h_1< 0.0557, \bar{I}_{concave}<0.4333, h_2/(h_1+ h_2)> 0.6248, h_3/(h_3+ h_4)> 0.769\}$ obtained as follows to identify non-appendiceal orifice edges and discard them.

To derive the above logic rule set, we experimented with several individual classifiers provided in Weka [75] including Naïve Bayes, Decision Tree J48, Support Vector Machines (SVM) with various parameters, and a supervised linear regression classifier using Radial Basis Function (RBF) as the model function. Among these classifiers, we chose the decision tree classifier because it gave the best performance in specificity and a reasonable performance in sensitivity using *10-fold cross validation* on the training data set. However, we found that a better sensitivity is obtained if the rules related to features with low information gain in the decision tree classifier are removed. We used this method to obtain the aforementioned logic rule set *R*. This is because the low-information gain features have a significant overlap in Gaussian-like distributions of the feature values. For such features, the decision tree classifier does not perform well. A classifier that utilizes probability of multivariable Gaussian distributions such as a Gaussian RBF is better.

Therefore, in the second layer, we extract three features from each remaining edge: $F_{RBF} = \{h_4,$ $W_{cross\text{-}section}, N_{cross\text{-}section}\}$. We compute a weighted Gaussian based RBF as

$$P = \sum_{i=1}^{n} w_i \cdot exp\,\{-\frac{(x_i-\mu_i)^2}{2\sigma_i^2}\},$$

where w_i is a pre-determined weight of feature *i*; μ_i and σ_i are the mean and standard deviation of feature *i* obtained from training; x_i is the value of feature *i* for the edge being considered and *n* is the number of features. In our experiments, equal weights are assigned to the three features. The remaining edge that has the value of *P* larger than a threshold P_{thld} (*0.3*) is detected as an appendiceal orifice edge. If an image has at least one detected appendiceal orifice edge, it is declared as an appendiceal orifice image.

4.3 Appendix Video Detection

Recall that an appendix video is defined as a colonoscopy video that shows at least 3 seconds of appendiceal orifice images. During this period, the endoscopist may not necessarily pause, but moves the camera around the orifice or closer or farther. Strong light reflection and partially blurred images can occur as a result. We propose an appendiceal orifice shot detection algorithm that detects near pause video segments of the appendiceal orifice visualization. Our near pause detection is based on similarity of global intensity histograms of sequential images, but allows for some dissimilarity to handle the aforementioned issues. The near pause detection helps to recall appendiceal orifice images with weak edges, poor illumination as well as reduce false positives caused by the edge-profile based appendiceal orifice image detection.

4.3.1 Near Pause Detection

Our near pause detection algorithm consists of three steps as follows.

Step 1: We divide each frame into *n* non-overlapping blocks of size B_i (15x15 pixels). Let $dist_k(i,j)$ denote the Euclidean distance between intensity histograms of block *k* between any two frames *i* and *j*. Given a threshold T_{blk}, a similarity score for block *k* between the two frames is defined as follows.

$$s_k(\mathrm{i},\mathrm{j}) = \begin{cases} 1, if\ dist_k(i,j) \le T_{blk} \\ 0, \qquad otherwise \end{cases}$$

$$S(i,j) = \sum_{k=1}^{n} s_k(i,j)$$

The value for T_{blk} depends on B_i and is determined based on experiments with the training data. The similarity score *S(i,j)* between frames *i* and *j* is the number of similar blocks between the two frames. Given a threshold T_{frm}, we define a frame similarity function between any two frames *i* and *j* as follows.

$$Sim(\mathrm{i,j}) = \begin{cases} 1, if\ S(i,j) \leq T_{frm} \\ 0, \qquad otherwise \end{cases}$$

We will discuss how to choose an appropriate value for T_{frm} shortly after relevant concepts related to this threshold are presented.

Figure 4.5 shows a graph that depicts frame similarity between frames in a video. Nodes represent frames and an edge between any two nodes *i* and *j* denotes frame similarity between frames *i* and *j*. Their similarity is strong (strong edge) when *Sim(i, j)* is one and it is weak (weak edge) when *Sim(i,j)* is zero. A weak similarity may be caused by light reflection, large camera movements, presence of instruments, etc.

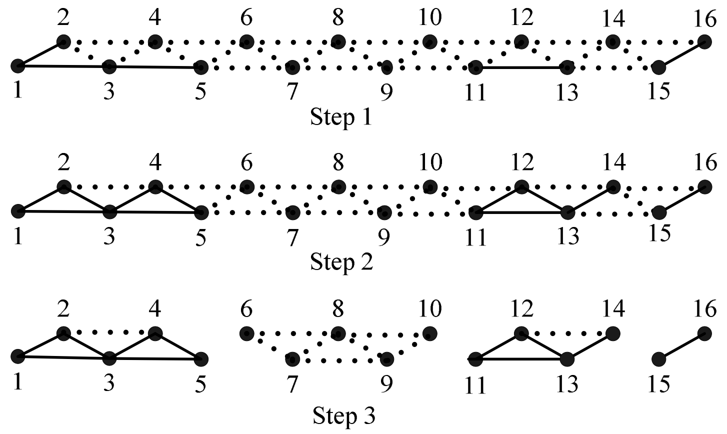


Figure 4.5: Example results of near pause detection for 16 sequential frames: Each node represents one frame; solid lines between each pair represent strong similarity between the frames and dot lines represent weak similarity between the frames.

*Step 2: T*o allow few frames to be very different from neighboring frames (e.g., frames with strong light reflection, low illumination, and blurriness) during a near pause, we refine weak edges as strong edges according to 2.1 and 2.2. Let frame *i* be the frame under consideration.

2.1: If *Sim(i-1, i)* is 0, but *Sim(i-1, i+1)* is 1 (i.e., frame *i* is different from its previous frame, but the two nearest neighbors of frame *i* are strongly similar), we assign 1 to *Sim(i-1, i)* and *Sim(i, i+1)*. That is, we include frame *i* into the same group as frames *i-1* and *i+1* even frame *i* is not so similar. We repeat this process until no further refinement can be made. For instance, in Step 2 of Figure 4.5, edges between nodes representing frames 3 and 4 and frames 4 and 5 are refined as strong edges (solid lines). Similarly, edges between nodes representing frames 11 and 12 and frames 12 and 13 are refined as strong edges.

2.2: If *Sim(i-1, i), Sim(i, i+1)* and *Sim(i-1, i+1)* are all zeros (i.e., all three consecutive frames are not similar, we check the values of *Sim(i-2, i-1)* and *Sim(i+1, i+2)*. If both of them are 1, assign 1 to *Sim(i-1, i)* and *Sim(i-2, i)* if *S(i-1, i) >= S(i, i+1)*. Otherwise, we assign 1 to *Sim(i, i+1)* and *Sim(i, i+2)*. In other words, we assign frame *i* to the group to which it is most similar. In Step 2 of Figure 4.5, the edge between nodes of frames 13 and 14 is refined as a strong edge since frame 14 is more similar to frame 13 than frame 15 does.

Step 3: We group sequential frames with strong edges into a near pause group. For instance, in Step 3 of Figure 4.5, frames 1 to 5, frames 11 to 14, and frames 15 to 16 each belong to a near pause group. We group sequential frames in between two near pause groups into a motion group. Frames 6 to 10 in Figure 4.5 are considered in the same motion group.

4.3.2 Refinement of Appendiceal Orifice Image Detection Results

We utilize the domain knowledge that the appendiceal orifice appears only in wall images and the results of the aforementioned appendiceal orifice image detection and near-pause detection. We determine for each image whether it is a wall image or a lumen image using our view mode detection technique [13], whether it is an appendiceal orifice image or non-appendiceal orifice image using the appendiceal orifice image detection technique, and whether it belongs to a near pause or a motion group. Lumen images are more sensitive to camera movement due to the large variance in pixel intensity distribution. Hence, they likely belong to motion groups. Wall images are more likely in near pause groups. The value of the aforementioned T_{frm} for near pause detection was chosen based on experiments with the training data to minimize the number of frames in a near pause group of wall images (where the appendiceal orifice appearance is more likely) from being divided incorrectly into subgroups.

Regardless of the group type, if the percentage of lumen images in the group is larger than $Thld_{Lumen}$ (50%), all images in the group are marked as non-appendiceal orifice images. This is done to eliminate errors since the appendiceal orifice appears only in wall images. If the percentage of detected appendiceal orifice images in a near pause group is larger than $Thld_{Wall}$ (66%), we consider all wall images in the group as appendiceal orifice images. This helps to recall true appendiceal orifice images that were not detected by the appendiceal orifice image detection. We discuss how to select the values of $Thld_{Wall}$ and $Thld_{Lumen}$ in Section 4.4.

4.3.3 Video Classification

This step determines whether the video is an appendix video. Even if the appendiceal orifice image detection technique has a 90% specificity, the results will include many false positives (non-appendiceal orifice images detected as appendiceal orifice images) since there are many more non-appendiceal orifice images than appendiceal orifice images in a colonoscopy video. This prevented us from using the number of detected appendiceal images as marker that determines whether a video is an appendix video. We solve this problem by significantly increasing the specificity and performing appendiceal orifice image detection in a narrower image range around the end of insertion phase. We empirically determined a range between 2 minutes before and 6 minutes after the end of insertion frame determined by our analysis method (EOI_{video}) [12]. We label all images in near-pause groups as non-appendiceal orifice images if the group duration is less than 2 seconds. A 2 second duration is used here (instead of 3 seconds) since we want to be conservative and allow detection of acceptable quality of appendiceal orifice visualization. We also label all images in motion groups as non-appendiceal orifice images. If the number of detected appendiceal orifice images is larger than a threshold *K* (7 images chosen based on the training data), we determine that the video is an appendix video. Otherwise, the video is considered a non-appendix video. We discuss how to select the value of *K* in more detail in Section 4.4.

Out of the many parameters listed in Table 4.1, only 9 of these parameters are more sensitive than the others. Five of them as indicated in the table depend on the input image size; they are to be reconfigured for adult or pediatric scopes where there is a significant difference in image sizes. Normalization of values based on the input image size will be investigated as future work. The logic rule set *R*, $Thld_{Lumen}$, $Thld_{wall}$, *and K* can be obtained via training as discussed in Sections 4.2 and 4.4.

We show the flow chart of our appendix video detection algorithm in Figure 4.6 and the pseudo code of the algorithm in Figure 4.7.

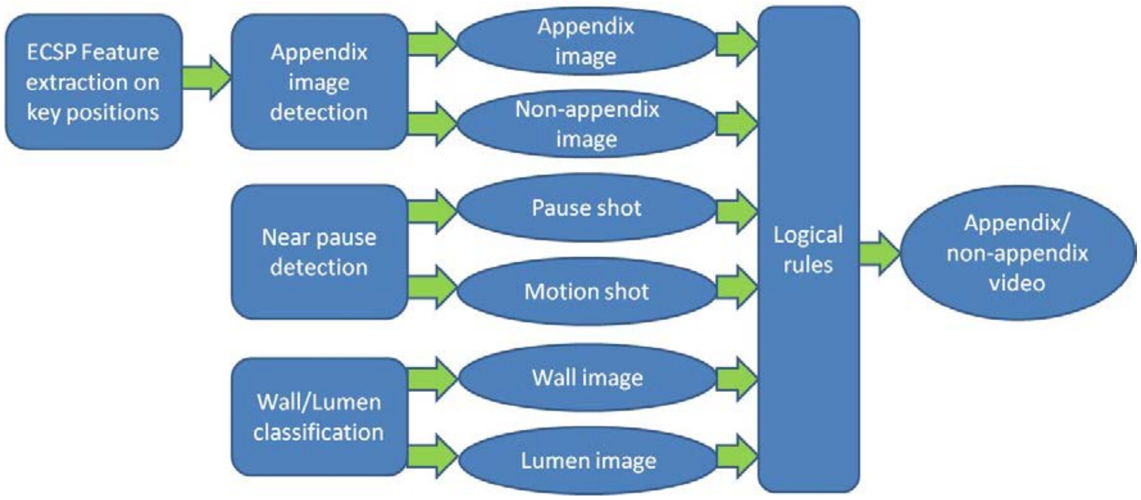


Figure 4.6: The flow chart of the appendix video detection algorithm

Algorithm: Appendix Video Detection

```
// Functionality: Classify a video as an appendix or a non-appendix video
// Input:
//      video: Input video
//      $EOI_{video}$: End of insertion frame number
//      FrameRate: Frame rate of the input video
//      $Thld_{Lumen}$: Threshold of the percentage of lumen images in a shot for detecting a motion
//                  shot
//      $Thld_{Wall}$: Threshold of the percentage of detected appendix images in a pause shot for
//                 refining all images in the shot as appendix images
//      K: Minimum number of detected appendix images for indicating the video as an appendix
//         video or a non-appendix video
// Output: True if the input video is detected as an appendix video
//          False, otherwise
ISAPPENDIXVIDEO (video, $EOI_{video}$ , FrameRate, $Thld_{Lumen}$, $Thld_{Wall}$, K)
1:  // Process frames 2 minutes before and 6 minutes after the end of insertion frame
2:  startFrame.id ← $EOI_{video}$ – 180*FrameRate
3:  endFrame.id ← $EOI_{video}$ + 360*FrameRate
4:  videoClip ← CREATEVIDEOCLIP( startFrame, endFrame)
5:  // Near pause detection for motion/pause shot segmentation
6:  // shot[]: Array of structures storing all shot information of videoClip
7:  // nShot: Number of shots in videoClip
8:  shot[], nShot ← PAUSEDETECTION(videoClip)
9:  // Call lumen/wall classification algorithm in [13] to classify frames as lumen/wall frames
10: // frameType: Array storing the type of images as either lumen or wall images
11: frameType ← LUMENWALLCLASSIFICATION(videoClip)
12: // Check the logical rule to classify images into appendix/non-appendix images
13: // videoClip.clearFrame[]: Array containing all information of clear frames
14: // fId: Frame ID in videoClip.clearFrame[]
15: fId ← 0
16: while ISEXIST(videoClip.clearFrame[], fId) do
17:     // sId: shot ID in the array shot
18:     sId ← GETSHOTINFORMATION(videoClip.clearFrame[], fID, shot[])
19:     // All frames in a motion shot or a short duration shot are non-appendix images
20:     if shot[sId ].Type == MOTION or shot[sId].Duration < 2 seconds then
21:         fId ← fId+1
22:         continue
23:     end if
```

Algorithm: Appendix Video Detection (continued)

```
        //Count the frame percentage of clear lumen images in the shot
        lumenPct ← LUMENPERCENTAGE (shot[sId], frameType[fId])
        //Non-appendix image if the lumen percentage in a pause shot > $Thld_{Lumen}$
        if lumenPct > $Thld_{Lumen}$ then
            fId ← fId+1
            continue
        end if
        // Classify an images into an appendix or a non-appendix image
        // videoClip.clearFrame[fId].Type: an appendix or a non-appendix type
        videoClip.clearFrame[fId].Type
        ← APPENDIXIMAGEDETECTION(videoClip.clearFrame[], fId)
        fId ← fId+1
    end while
    // Refine detected appendix images in the pause shot
    for sId ← 0 to nShot
        if shot[sId].Type == PAUSE then
            // Count the percentage of detected appendix images in the pause shot
            appendixPct ← APEENDIXPERCENTAGE(shot[], sId)
            if appendixPct > $Thld_{Wall}$ then
                // Set all images in the shot as appendix images
                SETALLASAPPENDIX(videoClip, shot[], sId)
            end if
        end if
    end for
    // Count total number of detected appendix images in the video clip
    if COUNTAPPENDIXIMAGES(videoClip) > K then
        return true
    end if
    return false
```

Figure 4.7: Pseudo code of the appendix video algorithm

Table 4.1: Parameters and Values for Appendix Image Detection

No.	Parameter Name	Valid Parameter Value/Range
1	Canny edge detector	Low threshold of 0.05; high threshold of 0.1; sigma of 3
2	Edge size range *E**	The number of edge pixels of valid candidate edges is between 100 and 500 pixels.
3	Linear edges*	The edge with the least squares fitting error < 8 pixels along either the vertical or horizontal image direction is determined as a linear edge.
4	*Curvature*	Curvature range of candidate appendiceal orifice edges is between 1/320 and 1/20.
5	W_i	10x10 pixels; the range from 10x10 to 25x25 pixels is the best to model the tangent vector direction
6	Light-reflected pixels	$V>0.4$ and $S>0.2$; $P_s=0.6$ for modeling and using the saturation distribution
7	$F_{ellipse}(N)$*	Reject edges with $F_{ellipse}(N)>0.2$ for $N<50$, $F_{ellipse}(N)>0.4$ for $50 \leq N<100$, and $F_{ellipse}(N)>0.7$ for $N\geq 100$
8	M_i	5x5 pixels
9	$Thld_{degree}$	15 degrees; the value of $Thld_{degree}$ depends on M_i and 15 degrees are sufficient for the window size of 5x5 pixels
10	λ	0.5~1.0 are shown to perform the best
11	$d_{first\text{-}order}$	0.001
12	$d_{cross\text{-}section}$\*	20 pixels
13	EOI_{image}	4 minutes before and 8 minutes after the end of insertion frame; this number is conservative based on the error rate of the method for the detection of the end of insertion frame
14	Logic rule set *R*	$R = \{h_1< 0.0557, \bar{I}_{concave} <0.4333, h2/(h1+ h2)> 0.6248, h3/(h3+ h4)> 0.769\}$ extracted from the decision tree classifier
15	P_{thld}	0.3
16	B_i	15x15 pixels
17	T_{blk}	70 (depending on B_i)
18	T_{frm}*	700 blocks
19	$Thld_{Lumen}$	50%
20	$Thld_{Wall}$	66%
21	EOI_{video}	2 minutes before and 6 minutes after the end of insertion frame
22	K	7 images

All parameters in this table were obtained based on experiments with the training data set. They were then used for running the experiments on the testing data set.

* Parameters sensitive to an input image size

4.4 Experimental Results

We implemented all image analysis code in MATLAB. The data sets used in this study are from a test bed generated using FUJINON endoscopes during routine screening colonoscopy. Table 4.2 shows our training data set for obtaining various parameters for image level analysis. This data set consists of manually selected clear images from *7* appendix videos. To obtain parameter values for video level analysis, we used a different training set that consists of all clear images from *5* other appendix videos and *5* non-appendix videos. Our testing data set consists of *23* video files of colonoscopy and flexible sigmoidoscopy. The training data sets and testing data set do not overlap. We first extracted 1 frame per second from the videos from the two data sets and saved them in JPEG format with a resolution of *720×480* pixels and 24 bit color. We discarded blurry (out-of-focus) images from these videos using the blurry frame detection method in [70]. Both training and testing images cover image types often seen in colonoscopy video including images showing appendiceal orifice, lumen, wall, polyps, vessels and instruments. Strong light reflection and weak edges appear in these image sets. The 23 test videos were divided into 3 categories. Category 1 includes 10 videos that show single crescent or circle slit appendiceal orifice shapes which are often seen. We focus on detecting good visualization of these appendiceal orifice shapes. Category 2 includes 5 videos that show multiple connected crescent shapes or single unclear/out-of-bound appendiceal orifice shapes which are sometimes seen. Category 3 has 8 non-appendix videos including 6 colonoscopy videos and 2 flexible sigmoidoscopy videos. The correctness of the classification of the ground truth was verified by the domain expert. The experiments were run on an Intel Xeon 1.86 GHz PC with 4GB of RAM and Windows 2003 Server as the operating system. The average running time per image was 8.7 seconds (6.4 seconds for image pre-processing and feature extraction and 2.3 seconds for classification and near pause detection). Only clear images within 2 minutes before and 6 minutes after the detected end of insertion frame were analyzed and the analysis rate of 1 frame per second was sufficient. Therefore, the average total running time for detecting the quality visualization of appendiceal orifice in a colonoscopic procedure was 38 minutes and 17 seconds. The detection time is fast enough to confirm the orifice visualization quality to the patient before he/she leaves the endoscopic screening facility. Manual review for quality control is subjective, tedious, and time-consuming since it requires the reviewer to first locate the appendiceal orifice images using fast-forward/reverse playback and to carefully look at these frames to determine whether the orifice inspection is of good quality.

Table 4.2: Training Data for Image Level Selected From 7 Videos for Appendix Detection

	Appendix	Lumen	Polyp	Vessel	Instrument
Number of images	121	292	221	71	71
Number of edges	137	1634	122	424	98

To choose good parameter values for $Thld_{Wall}$, $Thld_{Lumen}$, and K used in video level analysis, we first studied the effectiveness of different combinations of values of ($Thld_{Wall}$, $Thld_{Lumen}$). As not all possible combinations of $Thld_{Wall}$ and $Thld_{Lumen}$ in the range [0.0, 1.0] can be selected, we decided on 7 representative values {0, 1/4, 1/3, 1/2, 2/3, 3/4, 1} for each of the thresholds. As a result, there are a total of 49 possible combinations. Recall that our training data set has two classes of videos: 5 appendix videos and 5 non-appendix videos. Let the number of detected appendiceal orifice images in an appendix video i be $\#APX_i$, the number of detected appendiceal orifice images in a non-appendix video j be $\#APXN_j$, and the number of real appendiceal orifice images in an appendix video i be $\#APXR_i$. The following two steps are used to select the best combination.

Step 1) Select candidate combinations of ($Thld_{Wall}$, $Thld_{Lumen}$) with large

$$Dist_{between} = \frac{Min\{\#APX_i\} - Max\{\#APXN_i\}}{Min\{\#APX_i\} + Max\{\#APXN_i\}} \ ,$$

where i, j are from 1 to M (the maximum number of videos used for training in each class; M = 5 in our study). This is to find the threshold values that result in a large difference between the numbers of detected appendix images between the two classes.

Step 2) Select the combination with the largest value of

$$Recall_{witnin} = \frac{1}{M}\sum_{i=1}^{M}\frac{\#APXR_i}{\#APX_i}$$

among candidate pairs of ($Thld_{Wall}$, $Thld_{Lumen}$) obtained from Step 1 for the appendix video class. The higher the $Recall_{within}$, the more true positives are detected. The combination of (2/3, 1/2) for ($Thld_{Wall}$, $Thld_{Lumen}$) was selected and the threshold K was calculated as (Min{#APX$_i$} + Max{#APXN$_j$})/2 using the selected combination, which results in K=6.5. We used K of 7 for testing.

After all parameter values were determined using the training data sets, we ran our code on the testing set using the obtained parameter values. In the image level classification, we measured performance using sensitivity (recall) and specificity as the percentage of correctly detected appendiceal orifice and non-appendiceal orifice images of ground truth images. In the video level, we use accuracy which is the percentage of videos that are correctly classified. Based on the experiment results of the image level classification, we have the average performance on the 23 test videos as shown in Tables 4.3, Table 4.4 and Table 4.5. For Category 1, the average sensitivity and specificity are 91.82% and 91.02% without near pause detection. With near pause detection, the average sensitivity increases noticeably to 96.86% and the average specificity slightly decreases to 90.04%. For Category 2, the average sensitivity and specificity are 48.01% and 93.17% without near pause detection, and are 50.93% and 93.96% with near pause detection. The reason for low sensitivity for Category 2 is that some appendiceal orifice images are not detected because the orifice is 1) near the image boundary or 2) in an area with weak illumination or covered by water or 3) distorted such that it no longer has a nice crescent like shape. The distortion is caused by air suction, resulting in multiple connected curvilinear structures with a sharp corner as a connection point of these shapes. Such multiple connected orifice shapes are rejected by ellipse fitting in the pre-processing step unless the sharp corner was not detected or the orifice edges are broken apart in the edge extraction step. The misses in the first two cases are not crucial in practice because optimal inspection of the appendiceal orifice should have adequate illumination with the clear seen orifice in the center of the image. For Category 3 (non-appendix videos only), near pause detection improves the average specificity from

89.33% to 90.93%.

At the video level, recall that the goal of the video level is to increase specificity while maintaining a reasonable sensitivity to determine whether the video is an appendix video. Near pause detection significantly increases the average specificity to 97.68% for Category 1, 98.91% for Category 2, and 98.30% for Category 3 (results not shown in any tables). We label the video as an appendix video if we find at least K images (K=7) detected as appendiceal orifice images. Otherwise, the video is labeled as a non-appendix video. The algorithm correctly classifies 91.30% (21/23) of the videos including around 93% (14/15) of appendix videos and around 87.5% (7/8) of non-appendix videos. For the 2 mis-classified videos, one video is in Category 2 (*Video ID 15* in Table 4.4) and the other video is in Category 3 (*Video ID 18* in Table 4.5), but there is no misclassified video in Category 1. See Table 4.6.

Table 4.3: Performance of Image Level Classification Results on Appendix Videos (Category 1)

Video ID	Appendiceal Orifice Shape	Before Near Pause Detection					After Near Pause Detection				
		Confusion Matrices			Performance		Confusion Matrices			Performance	
			# Detected		Sen.	Spe.		# Detected		Sen.	Spe.
		RL	AP	NP	%	%	RL	AP	NP	%	%
1	Single crescent slit	AP	18	1	94.74	86.43	AP	18	1	94.74	88.39
		NP	76	484			NP	65	495		
2	Single crescent slit	AP	7	1	87.50	88.88	AP	8	0	100	88.75
		NP	90	719			NP	91	718		
3	Single crescent slit	AP	3	0	100	90.65	AP	3	0	100	93.53
		NP	39	378			NP	27	390		
4	Single crescent slit	AP	48	1	97.96	90.06	AP	48	1	97.96	90.21
		NP	67	607			NP	66	608		
5	Single crescent slit	AP	12	1	92.31	87.66	AP	12	1	92.31	89.37
		NP	108	767			NP	93	782		
6	Circle	AP	12	3	80.00	85.04	AP	15	0	100	89.23
		NP	82	466			NP	59	489		
7	Single crescent slit	AP	17	1	94.44	95.18	AP	16	2	88.89	96.07
		NP	60	1186			NP	49	1197		
8	Single crescent slit	AP	34	4	89.47	92.88	AP	36	2	94.74	94.63
		NP	77	1004			NP	58	1023		
9	Single crescent slit	AP	9	2	81.82	82.59	AP	11	0	100	85.44
		NP	98	465			NP	82	481		
10	Single crescent slit	AP	15	0	100	88.62	AP	15	0	100	89.05
		NP	80	623			NP	77	626		
Cat. 1	Average				91.82	91.02				96.86	90.47

Cat. - Category; Sen. - Sensitivity; Spe. - Specificity; RL - Real Number of Images; AP - Appendiceal Orifice Image; NP - Non-Appendiceal Orifice Image

Table 4.4: Performance of Image Level Classification Results on Appendix Videos (Category 2)

Video ID	Appendiceal Orifice Shape	Before Near Pause Detection					After Near Pause Detection				
		Confusion Matrices			Performance		Confusion Matrices			Performance	
			# Detected		Sen. %	Spe. %		# Detected		Sen. %	Spe. %
		RL	AP	NP			RL	AP	NP		
11	Multiple curvilinear structures	AP	20	22	47.62	91.93	AP	22	20	52.38	92.26
		NP	74	843			NP	71	846		
12	Multiple curvilinear structures	AP	12	6	66.67	94.84	AP	13	5	72.22	95.51
		NP	54	993			NP	47	1000		
13	Single crescent slit orifice out-of-bound	AP	13	9	59.09	90.33	AP	14	8	63.64	90.82
		NP	59	551			NP	56	554		
14	Single crescent slit but with water spots	AP	12	6	66.67	91.40	AP	12	6	66.67	93.35
		NP	66	701			NP	51	716		
15	Single crescent slit but poor illumination	AP	0	9	0.00	97.37	AP	0	9	0.00	97.85
		NP	22	814			NP	18	818		
Cat. 2	Average				48.01	93.17				50.98	93.96

Cat. - Category; Sen. - Sensitivity; Spe. - Specificity; RL - Real Number of Images; AP - Appendiceal Orifice Image; NP - Non-Appendiceal Orifice Image

Table 4.5: Performance of Image Level Classification Results on Non-appendix Videos (Category 3)

Video ID	Video Type	Before Near Pause Detection				After Near Pause Detection			
		RL	# Detected		Spe.	RL	# Detected		Spe.
		NP	AP	NP	%	NP	AP	NP	%
16	Colonoscopy video without appendix	1248	37	1211	97.04	1248	33	1215	97.36
17	Colonoscopy video without appendix	223	23	200	89.69	223	22	201	90.13
18	Colonoscopy video without appendix	213	42	171	80.28	213	31	182	85.45
19	Colonoscopy video without appendix	335	32	303	90.45	335	29	306	91.34
20	Colonoscopy video without appendix	313	34	279	89.14	313	32	281	89.78
21	Flexible sigmoidoscopy video	425	29	396	93.18	425	22	403	94.82
22	Flexible sigmoidoscopy video	124	21	103	83.06	124	15	109	87.90
23	Colonoscopy video without appendix	612	60	552	90.20	612	57	555	90.69
Cat. 3	Average				89.13				90.93

Cat. - Category; Sen. - Sensitivity; Spe. - Specificity; RL - Real Number of Images; AP - Appendiceal Orifice Image; NP - Non-Appendiceal Orifice Image

Table 4.6: Performance of Appendix Video Classification Results

Video ID	Appendiceal Orifice Shape	Detected Type	Correctly Detected
1	Single crescent	Appendix Video	Yes
2	Single crescent	Appendix Video	Yes
3	Single crescent	Appendix Video	Yes
4	Single crescent	Appendix Video	Yes
5	Single crescent	Appendix Video	Yes
6	Circle	Appendix Video	Yes
7	Single crescent	Appendix Video	Yes
8	Single crescent	Appendix Video	Yes
9	Single crescent	Appendix Video	Yes
10	Single crescent	Appendix Video	Yes
11	Multiple curvilinear structures	Appendix Video	Yes
12	Multiple curvilinear structures	Appendix Video	Yes
13	Single crescent but orifice out-of-bound	Appendix Video	Yes
14	Single crescent but with water spots	Appendix Video	Yes
15	Single crescent but poor illumination	*Non-Appendix Video*	*No*
Video ID	Non-Appendix Video Type	Detected Type	Correctly Detected
16	Colonoscopy video without appendix	Non-Appendix Video	Yes
17	Colonoscopy video without appendix	Non-Appendix Video	Yes
18	Colonoscopy video without appendix	*Appendix Video*	*No*
19	Colonoscopy video without appendix	Non-Appendix Video	Yes
20	Colonoscopy video without appendix	Non-Appendix Video	Yes
21	Flexible sigmoidoscopy video	Non-Appendix Video	Yes
22	Flexible sigmoidoscopy video	Non-Appendix Video	Yes
23	Colonoscopy video without appendix	Non-Appendix Video	Yes

4.5 Chapter Summary

This chapter introduces two algorithms for detecting the image showing the clearly seen appendiceal orifice and the video showing at least three seconds of the appendiceal orifice inspection based on ECSP features. The proposed appendix video detection algorithm first uses new local features derived from ECSP and its 1st – order derivative profile to detect the appearance of the appendiceal orifice and then uses near pause detection to recall miss detected appendiceal orifice images with weak edges and reject some false classifications. The experimental results show a reasonable performance at both image level and video level. Our proposed approach is able to handle object rotation, translation and scaling in image, and noisy images with strong light reflection.

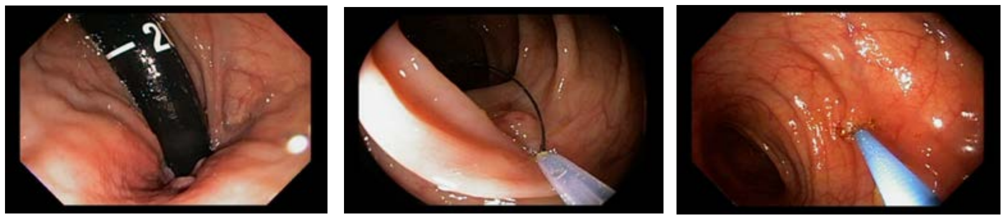
The future work includes reducing the analysis time of the appendiceal orifice image detection by converting the analysis code from MATLAB to C/C++ and further optimizing through multi-threading and Graphical Processing Units (GPU) for parallel execution.

CHAPTER 5 RETROFLEXION IMAGE AND VIDEO DETECTION

This chapter introduces two algorithms for detecting the appearance of an endoscope during a retroflexion operation in colonoscopy videos. The first algorithm detects the region of the endoscope using region location and shape based (RSL) features. The second algorithm uses an edgeless ECSP features extraction method that extracts ECSP features without using edges.

5.1 Introduction

Retroflexion is an endoscope maneuver where the tip of a flexible endoscope equipped with a wide angle lenses is deflected more than *90* degrees from the axial direction of the shaft of the endoscope. This results in visualization of intestinal mucosa along the shaft of the endoscope (Figure 5.1 (a)). Recall that the colon consists of six parts: cecum with appendix, ascending colon, transverse colon, descending colon, sigmoid and rectum. When retroflexion is performed in the rectum, physicians can inspect the peri-anal mucosa. Studies reported that retroflexion improved the yield of polyps [14] [15]. Rectal retroflexion was also suggested as an essential part of the colonoscopy examination of the large bowel [14]. In a small pilot study rectal retroflexion rates were studied and found to be low (data not shown). Even after reminders that retroflexion should be performed low retroflexion rates were still detected. We envision that if the absence of retroflexion in either right colon or rectum can be shown to the endoscopist during the procedure - given near real-time detection - it should improve quality and the chances for polyp or tumor detection. In this chapter, we propose a novel algorithm able to detect the operation of retroflexion effectively in *near real-time* (at least *1* frame per second) on a modern PC. For clinical use, if retroflexion is not detected by our system during colonoscopy, the system could alert the endoscopist before the end of the procedure. Instead of using our proposed automatic detection of retroflexion, the endoscopist can push a button when performing retroflexion. However, this adds to the work of the endoscopist and is subject to human error.



(a) Endoscope seen during retroflexion **(b) Polyp removal using a snare** **(c) Biopsy using forceps**

Figure 5.1: Examples of instruments used during colonoscopy

Colonoscopy allows careful colon examination and operations like polyp removal and retroflexion are typically performed during the withdrawal phase of the procedure. Rectal retroflexion is usually short (around *1-3* seconds) compared to the entire withdrawal phase (at least *6* minutes duration recommended [1]). Because examination of the rectum is close to the end of the colonoscopic procedure, rectal retroflexion generally starts within the last minute of a procedure. During retroflexion, the endoscope is typically seen as shown in Figure 5.1 (a). The endoscope is generally black. It has a longitudinal shape like other instruments used for colonoscopy such as snares and biopsy forceps (Figure 5.1 (b-c)). We call a region (*8*-direction connected pixel component) showing an endoscope in an image a *retroflexion region*, an image showing an endoscope a *retroflexion image* and a video showing retroflexion images a *retroflexion video*. Otherwise, the region is called a *non-retroflexion region*, the image is called a *non-retroflexion image*, and the video is called a *non-retroflexion video*.

Challenges of retroflexion detection: The challenges of the retroflexion detection problem include the following aspects. Firstly, retroflexion typically occurs in a very short duration. Therefore, a very high specificity of retroflexion image detection is required to identify that whether there exists a retroflexion operation or not during the withdrawal phase. Secondly, the appearance of the endoscope has large variation. During retroflexion, the endoscope may be bent (Figure 5.2 (a)), appear in gray color (Figure 5.2 (b)) or blurry due to rotation of the scope, or partially occluded (Figure 5.2 (c)). The scope may appear in a small portion of the screen (Figure 5.2 (d-e)) blending with the black background at the edge of its field of view. Hence, methods based on template matching [3] may not able to model the shape of an endoscope in image and it is time consuming to obtain an endoscope region using existing image segmentation algorithms. Lastly, the most difficult challenge causing false alarms is that the endoscope color and illumination resemble those of the dark, distant lumen (Figure 5.2 (f-h)) that appears frequently. Hence, using color features alone is not sufficient.

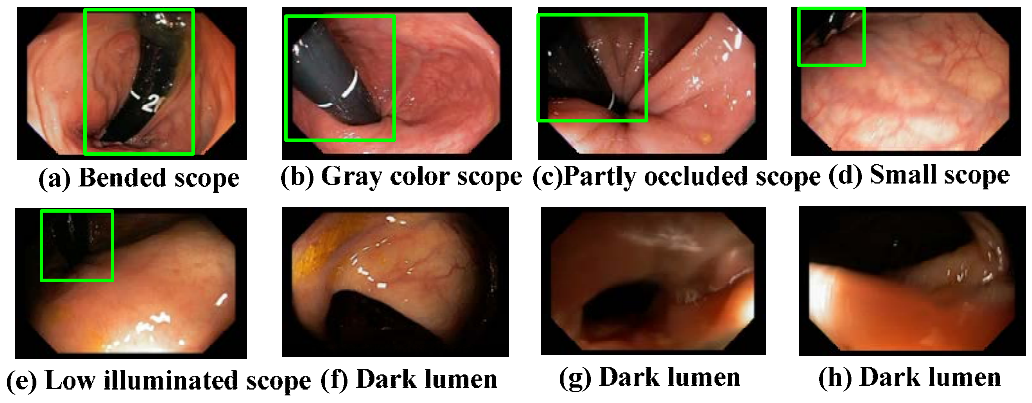


Figure 5.2: Examples of retroflexion images and dark lumen images. Retroflexion regions are marked in green rectangles.

5.2 Image Preprocessing

We obtain the candidate retroflexion regions in the images by applying the following two steps. The threshold and parameter values mentioned in this section were obtained from our experiments described in Section 5.5.

Step 1: **Candidate retroflexion region extraction:** We simply threshold pixels values to keep as many real retroflexion regions as possible by experimenting with the training data. We extract pixels that satisfy Red $< \text{Thld}_R$ (100) and Hue $> \text{Thld}_H$ (0.1), where Red is the value of the red channel in RGB color space and Hue is the hue value in HSV color space. A binary image *I* is obtained by assigning 1 to the extracted pixels, and 0 to the other pixels. We connect isolated pixels in I by applying a dilation operation [74] with a structuring element of a circle shape with a radius of r_1 (5 pixels). Finally, we obtain each 8-neighbor connected pixel component as a candidate retroflexion region.

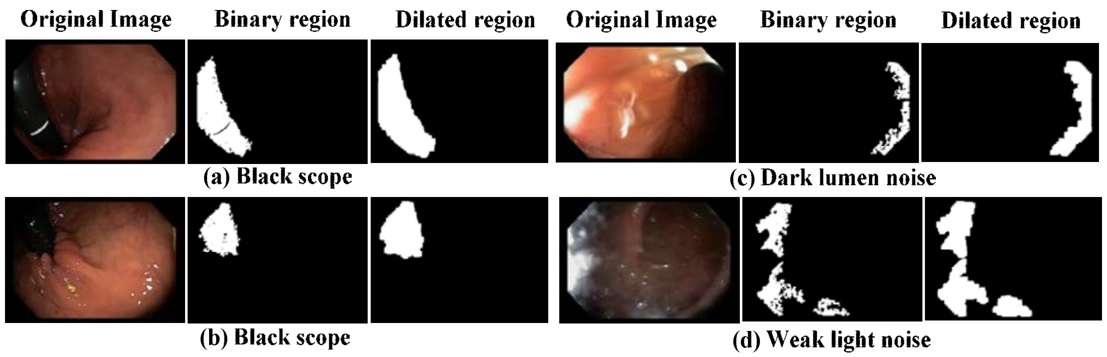


Figure 5.3: Dark regions of black scopes (a-b) and some noises (c-d): the binary regions in the last column are after the dilation using the structuring element with a circle shape of radius r_2.

Step 2: **Region filtering:** 1). We first discard small candidate regions having less than $\mathrm{Thld}_{\mathrm{size}}$ (*3000*) pixels. 2). Based on our observation, the darkness of most black scopes is due to their inherent color of the material while the darkness of some noisy images is caused by insufficient light. This results in an interesting pattern in that the retroflexion regions usually look more "bushy" (i.e., the pixels are already more strongly connected) than the regions caused by insufficient light. See Figure 5.3. We discard some noisy regions using a dilation operation with a large structuring element of a circle shape with a radius of r_2 (*15* pixels). Specifically, we discard a region R if $\mathrm{N_R}/\mathrm{N_{R'}} < Thld_{\mathrm{Dilate}}$, where $\mathrm{N_R}$ and $\mathrm{N_{R'}}$ are the number of pixels in the region before and after the dilation, respectively. We set $\mathrm{Thld}_{\mathrm{Dilate}}$ to *0.6*.

The images with at least one remaining region after the region filtering step are used for the feature extraction. Based on our experiments (described in Section 5.5), after the image preprocessing step, about *10%* of the images showing false alarms and *88%* of the images showing true retroflexion remain. Next, we further reduce the number of false alarm images using classification of our region features.

5.3 Region and Location Features

In this section, we describe our feature extraction for RSL features for each remaining candidate region using the following steps.

For each candidate retroflexion region *R*, we obtain its sub-region R_s using the downsampling method by the sampling factor of L (*5* pixels), i.e., selecting one pixel out of every L pixels. Downsampling is used to achieve a fast calculation of Principle Component Analysis (PCA) on the coordinates of the pixels in R_s as follows.

Let the vector $\vec{u}_i = [x_i, y_i]^T$ be coordinates of pixel *i* on the 2D image $I(\vec{u}_i)$. Define a 2D data matrix $X = [\vec{u}_1, \dots, \vec{u}_{N_{R_s}}]$, where N_{R_s} is the number of pixels in R_s. The covariance matrix of *X* is given by $\Sigma = \big(X - E(X)\big)(X - E(X))^T$, where *E(X)* is the expectation vector of the matrix *X*. Let the eigenvalues of Σ be $|\lambda_1(\Sigma)|$ and $|\lambda_2(\Sigma)|$, where $|\lambda_1(\Sigma)| > |\lambda_2(\Sigma)|$, and their corresponding eigenvectors be $e_1(\Sigma)$ and $e_2(\Sigma)$, respectively. The eigenvector $e_1(\Sigma)$, corresponding to the largest absolute eigenvalue $|\lambda_1(\Sigma)|$, represents the direction along which the variance of variables in *X* is maximum. Since Σ is a *2-by-2* symmetric matrix, calculation of the corresponding eigenvalues and eigenvectors is simple and fast without the time-consuming Singular Value Decomposition method. We skip processing a region whose Σ is singular. However, this is a very rear case in our problem. For example, the shape of an endoscope region generally cannot be a perfect circle or a line with the width of one pixel.

Calculation of two end positions: We project all data points in the region R_s onto the $e_1(\Sigma)$ axis and obtain p_j *and* p_k as the two end points of the endoscope portion seen in the field of view (FoV) of the endoscope. These two end points have the largest Euclidean distance from each other among the projected points. See Figure 5.4 and 5.5.

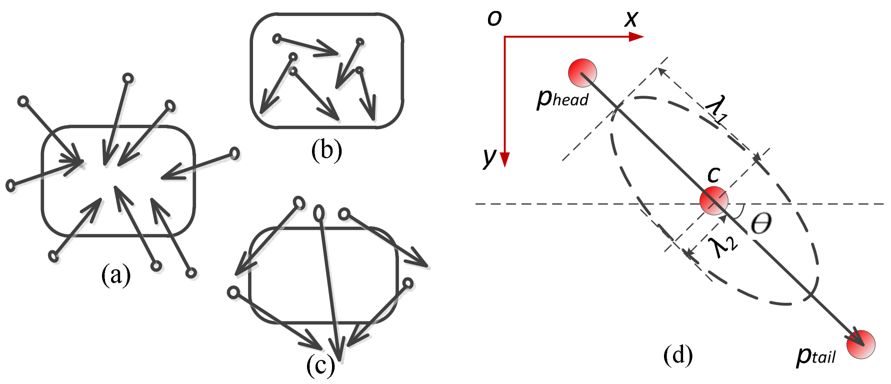


Figure 5.4: (a-c) possible insertion directions during retroflexion: Each rounded rectangle represents $\boldsymbol{A_{inside}}$; the top red circle on an arrow (vector) represents p_{head}; (d) diagram showing some region features

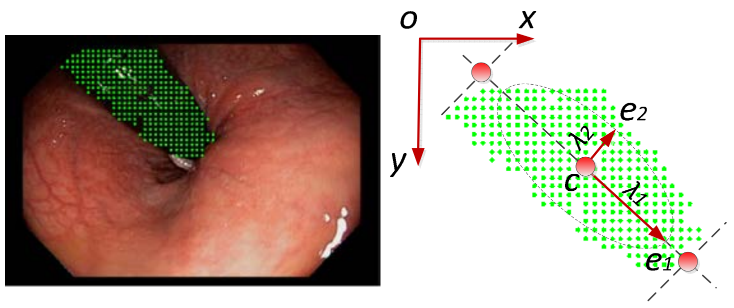


Figure 5.5: Left: Downsampled candidate region shown in green. Right: Eigenvectors and eigenvalues of the downsampled region on the left; the two end points and the center c are marked as red points.

Estimation of the insertion direction: Let A_{inside} be the area in the FoV of the endoscope. Let $A_{outside}$ be the area whose pixels do not belong to A_{inside} such as the black area outside the FoV of the endoscope. We can determine the black area by analyzing several frames in the beginning of the procedure. We shrink $\mathrm{A_{inside}}$ by dilating $A_{outside}$ using a structuring element of a circle shape with a radius of r_3 (*45* pixels).

The region direction (insertion direction) of an endoscope is represented by $\overrightarrow{p_{head}p_{tail}}$ – a 2D vector pointing from p_{head} to p_{tail}. From the domain knowledge, the endoscope is always inserted from $A_{outside}$to A_{inside}, and generally inserted from the top of the image to the image bottom. Therefore, we match p_{head} and p_{tail} to the end points p_j and p_k subject to the following conditions.

1) If both end points belong to $\mathrm{A}_{\mathrm{inside}}$ or $\mathrm{A}_{\mathrm{outside}}$, let $\overrightarrow{p_{head}p_{tail}}$ point from the top to the bottom as shown in Figure 5.4 (b-c).
2) If one of the end points belongs to $\mathrm{A}_{\mathrm{inside}}$whereas the other one belongs to $\mathrm{A}_{\mathrm{outside}}$, let $\overrightarrow{p_{head}p_{tail}}$ point from $\mathrm{A}_{\mathrm{outside}}$to $\mathrm{A}_{\mathrm{inside}}$ as shown in Figure 5.4 (a).

RSL descriptor: Given a candidate region, we extract the following properties as RSL features:

1) Center location of the region: $c = [x_c, y_c]^T$

2) Tail position: $p_{tail} = [x_t, y_t]^T$

3) Head position: $p_{head} = [x_h, y_h]^T$

4) Approximated half length: $|\lambda_1(\Sigma)|$

5) Approximated half width: $|\lambda_2(\Sigma)|$

6) Ratio of width to length: $|\lambda_2(\Sigma)|/|\lambda_1(\Sigma)|$

7) Insertion direction: the angle θ between $e_1(\Sigma)$ and the x-axis where θ is in the range of [*0-360*) degrees.

Figure 5.4 (d) shows some corresponding RSL features.

5.4 Edgeless Edge Cross-Section Profile

In Chapter 3, we proposed a method to calculate edge cross-section profile (ECSP). The technique extracted features along edge cross-section directions. We obtain candidate edges using the Canny edge detection algorithm and their edge cross-section directions (edge normal directions). We call the ECSP feature extraction algorithm that relies on edge detector the **edge-based ECSP**. In Chapter 4, we extracted features on ECSP functions for the detection of images showing appendiceal orifice in colonoscopy. Our previous edge-based ECSP for appendix image and video detection has the following limitations. *1).* Some endoscope edges were not extracted due to the image blur or complex motion such as the fast rotation of endoscope during retroflexion. *2)* Our edge-based ECSP calculation method requires the curve like shape of an edge. However, some edges of endoscopes obtained from edge detection algorithms were broken apart into small segments in our experiment. Many broken endoscope edge segments had a close to linear shape. Another solution is to extract endoscope contours using an image segmentation algorithm. See Figure 5.6 (a). However, processing a *720-by-480* resolution image using an image segmentation algorithm – like JSEG generally takes around *22* seconds per image on the same machine. In this section, we first briefly review the edge-based ECSP for object detection. Next, we propose an edgeless ECSP feature extraction algorithm to overcome these limitations.

5.4.1 Brief Review of Edge-Based ECSP

Let $\{I(\vec{u}_i)\}_{i=1}^{M}$ be a set of intensity values of the edge pixel $\vec{u}_i$, where *M* is the total number of pixels on this edge. The vector $\vec{u}_i$ denotes the pixel at the coordinate (x_i, y_i) on a 2D image. A unit vector $\vec{n}_i$ is an edge normal passing $\vec{u}_i$ pointing to the *concave side* of the edge. See Figure 5.6 (c). The ECSP function $f_i(r)$ for the edge pixel $\vec{u}_i$ is defined in Equation 3.1. The ECSP function $F(r)$ for the entire edge is defined in Equation 3.2.

We extract ECSP features based on edge position on $F(r)$ and zero-crossing value positions on $\nabla F(r)$ as described in Chapter 4. Some example features include edge strength before crossing the edge (h_1), edge strength after crossing the edge (h_2), edge sharpness before crossing the edge (h_3), edge sharpness after crossing the edge (h_4). See Figure 5.6 (d-e) for the labeling of these features. More features and their detailed explanations can be found in Chapter 4.

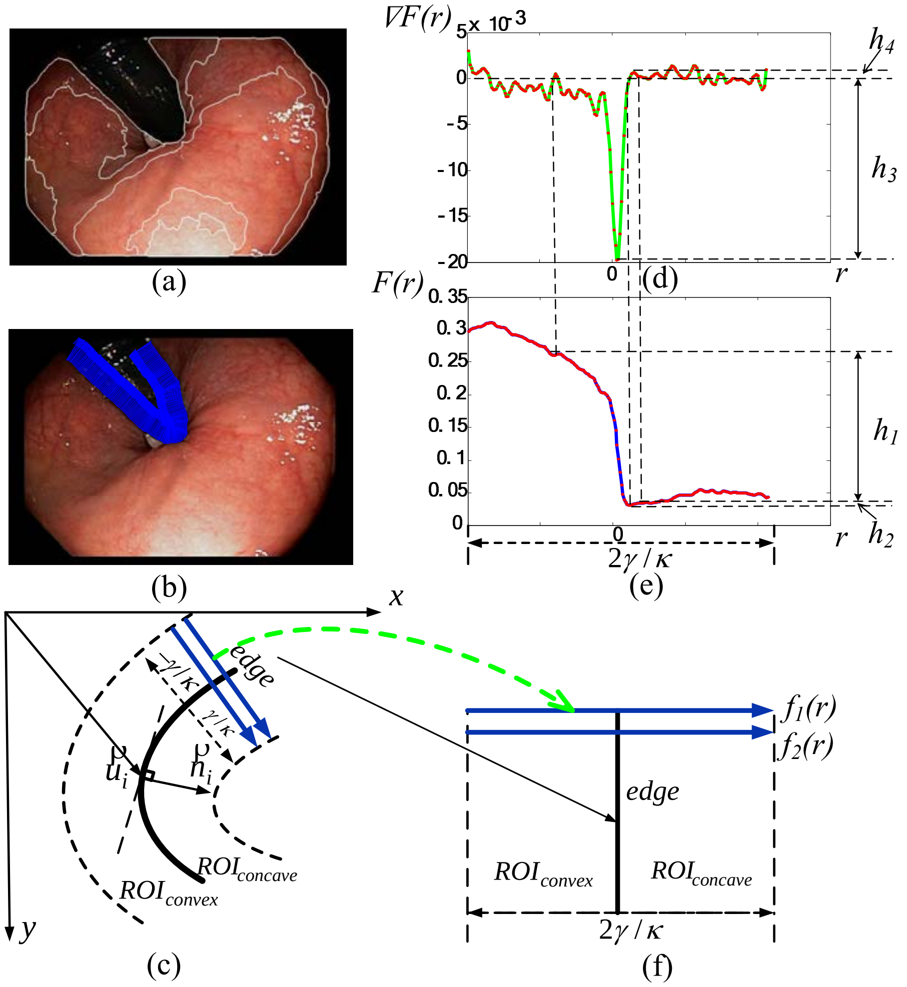


Figure 5.6: Edge-based ECSP calculation and feature extraction. (a) A retroflexion image with edge contours obtained using an image segmentation algorithm – JSEG; (b) Edge normal vectors (showing in blue arrows) of endoscope edge pixels; (c) The demonstration of edge-based ECSP calculation; (d) The corresponding gradient profile of *F(r)*; (e) The ECSP function *F(r)*; (f) The region of interest pixels along edge normal directions.

5.4.2 Edgeless ECSP

In this section, we propose an algorithm to extract ECSP features without using an edge detection algorithm – **edgeless ECSP**. Our algorithm overcomes the aforementioned limitations and the calculation speed is fast (within half a second to process an image using MATLAB). The algorithm has the following four steps: 1) edge normal estimation, 2) ECSP calculation, 3) key positions selection and 4) Feature extraction.

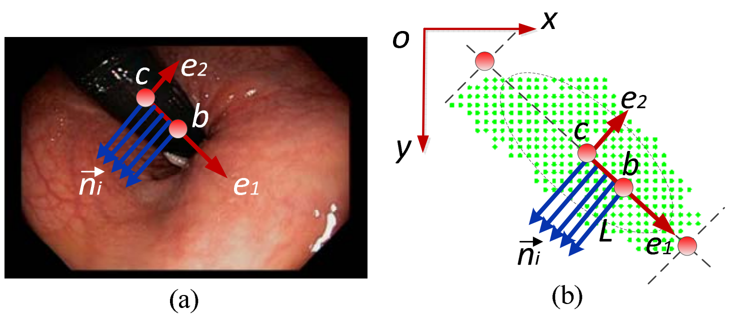


Figure 5.7: Edge normal directions labeled by blue arrows

Edge normal estimation: Without the detection of edge positions, the edge normal directions of endoscope edges are unknown. However, as an endoscope has a longitudinal shape, the edge normal directions of the edge pixels along the endoscope are approximately perpendicular to its insertion direction. We approximate its edge normal directions by a single direction – the eigenvector $e_2(\Sigma)$, corresponding to the second largest eigenvalue $\lambda_2(\Sigma)$. See Figure 5.7.

ECSP calculation: Let $\{I(\vec{u}_i)\}_{i=1}^{M}$ be a set of intensity values of pixels along the eigenvector $e_1(\Sigma)$, starting from the region center *c* and ending at position *b*, where *M* is the total number of pixels from *b* to *c*. We limit the *b* position along $e_1(\Sigma)$ by defining the Euclidian distance between *b* and *c* as $|\overrightarrow{bc}| = \beta|\lambda_1(\Sigma)|$, where β is a constant and $\beta \in [0,1]$. We set $\beta = 0.5$ to exclude the pixels close to p_{tail} since they are typically affected by the colon mucosa. As a result, the shape of the endoscope close to p_{head} is no longer approximately linear. The eigenvalue $|\lambda_1(\Sigma)|$ is used because it is proportional to the length of the endoscope. To fasten the calculation, we use a subset $\{\vec{u}_i\}_{i=1}^{m}$ by downsampling $\{\vec{u}_i\}_{i=1}^{M}$ with the sampling factor of Q (5 pixels), where $m = M/Q$.

Let the intensity value along the edge normal direction be the function of the pixel distance from $\vec{u}_i$. We define the two corresponding ECSP functions $f_i^+(r)$ and $f_i^-(r)$ along edge normal directions of the two sides of $\vec{u}_i$ as

$$f_i^+(r) = I(\vec{u}_i + r \cdot \vec{n}_i), r \in [0, \xi|\lambda_2(\Sigma)|] \tag{6.1}$$

and

$$f_i^-(r) = I(\vec{u}_i - r \cdot \vec{n}_i), r \in [0, \xi|\lambda_2(\Sigma)|], \tag{6.2}$$

where r = *0* starts from the pixel $\vec{u}_i$ which is the pixel *i* among the downsampled pixels between *b* and *c* in Figure 5.7 (b). The maximum value of *r* is proportional to the approximated width of endoscope $|\lambda_2(\Sigma)|$ and scaled by a positive constant ξ. Therefore, endoscope scales are reflected by *r*. To increase the chance that ECSPs pass real edge positions, we selected $\xi = 1.5$ based on experimental results. We do not use the ECSP if it extends into the $A_{outside}$ area; for example, we do not use $f_i^-(r)$ functions in Figure 5.8 (b) and $f_i^+(r)$ functions in Figure 5.8 (c). Figure 5.9 shows an example of a function $f_i^+(r)$. The two functions $f_i^+(r)$ and $f_i^-(r)$ generally appear similar for the same $\vec{u}_i$ if both $f_i^+(r)$ and $f_i^-(r)$ are used, e.g. Figure 5.8 (a).

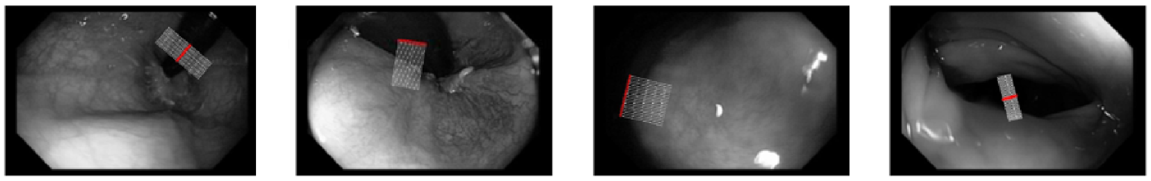


Figure 5.8: Grayscale images of endoscopes during retroflexion. (a-b) and non-retroflexion images (c-d). The estimated insertion directions are labeled as red color bars (r = *0*). The calculated ECSPs along edge normal directions are labeled as white color bars. In (b) and (c), only ECSPs that do not go beyond $\boldsymbol{A_{inside}}$ are labeled.

Key positions selection: We select some key positions to calculate ECSP features including a) the edge position on $f_i(r)$ and the corresponding position on its derivative function $\nabla f_i(r)$, b) the closest zero-crossing positions away from the edge position on $\nabla f_i(r)$ and c) their corresponding positions on $f_i(r)$. We calculate these key positions using the following steps. First, we smooth $f_i^+(r)$ using a moving average filter (with the window size of *5* pixels) and obtain its derivative function $\nabla f_i^+(r)$. Next, we determine a P_i^+ position as the position with the maximum value on $\nabla f_i^+(r)$ for each $\vec{u}_i$. See Figure 5.9. We consider the corresponding P_i^+ position on $f_i^+(r)$ as the edge position. Last, we search $\nabla f_i^+(r)$ along *r* from the two sides of P_i^+ until we find the two closest zero-crossing positions from P_i^+. Therefore, we obtained totally *6* key positions on $f_i^+(r)$ and $\nabla f_i^+(r)$: a) P_i^+ as the edge position on $f_i^+(r)$ and its corresponding position on $\nabla f_i^+(r)$, b) the two zero-crossing positions on $\nabla f_i^+(r)$ and c) their two corresponding positions on $f_i^+(r)$. We use the same steps to select the *6* key positions on $f_i^-(r)$ and $\nabla f_i^-(r)$.

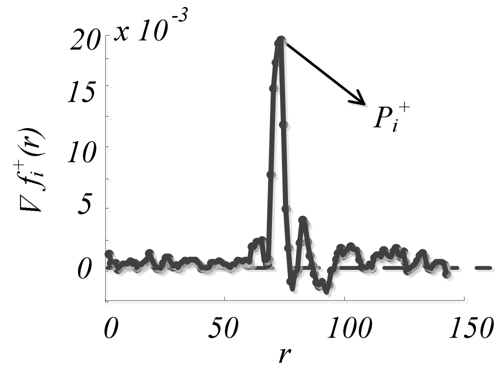


Figure 5.9: An example $\boldsymbol{\nabla f_i^+(r)}$ function and the selected $\boldsymbol{P_i^+}$ position. The maximum *r* value is $\boldsymbol{\xi|\lambda_2(\Sigma)|}$.

Feature extraction: Based on the selected key positions on $f_i^+(r)$ and $\nabla f_i^+(r)$, we apply the algorithm in Chapter 4 to calculate ECSP features for each $\vec{u}_i$. These features include

1) Edge strength before crossing the edge ($h_1^{+,i}$)
2) Edge strength after crossing the edge ($h_2^{+,i}$)
3) Edge strength ratio after and before crossing the edge ($h_2^{+,i}/(h_1^{+,i}+h_2^{+,i})$)
4) Edge sharpness before crossing the edge ($h_3^{+,i}$)
5) Edge sharpness after crossing the edge ($h_4^{+,i}$)
6) Edge sharpness ratio after and before crossing the edge ($h_4^{+,i}/(h_3^{+,i}+h_4^{+,i})$).

These ECSP features follow the same meaning of corresponding $h_1,\ h_2,\ h_2/(h_1+h_2),\ h_3,\ h_4$ and $h_4/(h_3+h_4)$ in Chapter 4. See Figure 5.6 (e-f). We use the median feature value of all *m* pixels as the corresponding feature for that edge. For example, $h_1^+ = median\{h_1^{+,i}\}_{i=1}^m$. Similarly, we obtain the same features from $f_i^-(r)$ and $\nabla f_i^-(r)$.

Finally, we obtain two feature vectors from $f_i^+(r),\ f_i^-(r),\ \nabla f_i^+(r)$ and $\nabla f_i^-(r)$ presented by

$$V^{(ECSP^{(+)})} = (h_1^+, h_2^+, h_2^+/(h_1^+ + h_2^+), h_3^+, h_4^+, h_4^+/(h_3^+ + h_4^+))$$

and

$$V^{(ECSP^{(-)})} = (h_1^-, h_2^-, h_2^-/(h_1^- + h_2^-), h_3^-, h_4^-, h_4^-/(h_3^- + h_4^-)).$$

Figure 5.10 shows the flow chart of our near real-time retroflexion image/video detection system using an ensemble classifier.

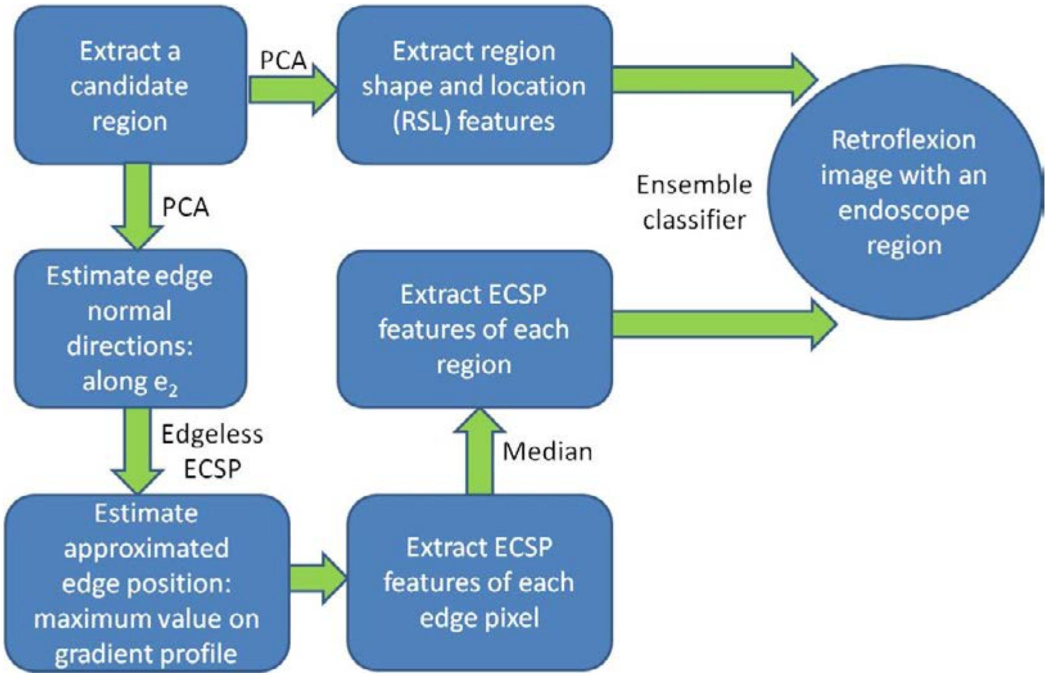

Extract a candidate region
PCA
Extract region shape and location (RSL) features
Retroflexion image with an endoscope region
Ensemble classifier
PCA
Estimate edge normal directions: along e2
Extract ECSP features of each region
Edgeless ECSP
Median
Estimate approximated edge position: maximum value on gradient profile
Extract ECSP features of each edge pixel

Figure 5.10: Flow chart of the near real-time retroflexion image/video detection system

We describe the pseudo code of our retroflexion region detection algorithm in Figure 5.11 and our retroflexion image/video detection algorithm in Figure 5.12. We consider a video as a retroflexion video if we detect at least one retroflexion image in the video.

Algorithm: Detection of A Retroflexion Region

```
// Functionality: Identify whether a region is a retroflexion region or a non-retroflexion region
// Input:
//      region: 2D coordinate values of pixels in a region
//      L: Down sampling rate of pixel coordinates in a region
// Output: True if a retroflexion region is detected
//          False, otherwise
IsRetroflexionRegion (region, L)
1:  region.downSampledPixels ← DownSampling(region, L)
2:  region.$\Sigma$ ← Covariance(region.downSampledPixels)
3:  // Get region shape and location features region.$V^{(RSL)}$ by Principle Component Analysis
4:  region.$V^{(RSL)}$ ← PrincipleComponentAnalysis(region.$\Sigma$)
5:  // Estimate the direction of the eigenvector $e_2$ of all pixels as the edge normal direction
6:  region.normal ← region.$V^{(RSL)}$.$e_2$
7:  // Extract pixel values along the edge normal direction and its reverse direction as ECSPs
8:  // M: Number of pixels along the principle axis of PCA used for edgeless ECSP calculation
9:  // The edgeless ECSP functions of a region are calculated based on Equations 6.1 and 6.2
10: $\{f_i^-(r)\}_{i=1}^M$, $\{f_i^+(r)\}_{i=1}^M$ ← ECSPExtraction(region)
11: // Calculate the 1st – order derivative functions
12: $\{\nabla f_i^-(r)\}_{i=1}^M$ ← Gradient($\{f_i^-(r)\}_{i=1}^M$)
13: $\{\nabla f_i^+(r)\}_{i=1}^M$ ← Gradient($\{f_i^+(r)\}_{i=1}^M$)
14: // Estimate the global maximum location on $\nabla f_i^-(r)$ and $\nabla f_i^+(r)$ as the edge positions
15: $\{P_i^-\}_{i=1}^M$ ← Max ($\{\nabla f_i^-(r)\}_{i=1}^M$)
16: $\{P_i^+\}_{i=1}^M$ ← Max ($\{\nabla f_i^+(r)\}_{i=1}^M$)
17: // Calculate edgeless ECSP features on $f_i^-(r)$, $f_i^+(r)$, $\nabla f_i^-(r)$ and $\nabla f_i^+(r)$
18: $\{h_1^{-,i}(r), h_2^{-,i}(r), h_3^{-,i}(r), h_4^{-,i}(r)\}_{i=1}^M$ ← ECSPFeatures ($\{f_i^-(r), \nabla f_i^-(r), P_i^-\}_{i=1}^M$)
19: $\{h_1^{+,i}(r), h_2^{+,i}(r), h_3^{+,i}(r), h_4^{+,i}(r)\}_{i=1}^M$ ← ECSPFeatures ($\{f_i^+(r), \nabla f_i^+(r), P_i^+\}_{i=1}^M$)
20: // Calculate edgeless ECSP features as the median feature values obtained from all pixels
21: region.$V^{(ECSP^{(-)})}$ ← Median($\{h_1^{-,i}(r), h_2^{-,i}(r), h_3^{-,i}(r), h_4^{-,i}(r)\}_{i=1}^M$)
22: region.$V^{(ECSP^{(+)})}$ ← Median($\{h_1^{+,i}(r), h_2^{+,i}(r), h_3^{+,i}(r), h_4^{+,i}(r)\}_{i=1}^M$)
23: // classify a region as a retroflexion or a non-retroflexion region using a ensemble classifier
24: if Classification(region.$V^{(RSL)}$) == true then
25:     if Classification (region.$V^{(ECSP^{(-)})}$) == true
26:         or Classification (region.$V^{(ECSP^{(+)})}$) == true then
27:         return true
28:     end if
29: end if
30: return false
```

Figure 5.11: Pseudo code of the retroflexion region detection algorithm

Algorithm: Detection of A Retroflexion Video

```
// Functionality: Classify a video as a retroflexion or a non-retroflexion video
// Input:
//        video: Input video
//        $EOI_{video}$: End of insertion frame number
//        $Thld_R$: Threshold of red color in the RGB color plane to select candidate region pixels
//        $Thld_H$: Threshold of hue color in the HSV color plane to select candidate region pixels
//        $Thld_{Dilate}$: Threshold of the ratio of region sizes before and after the dilation
//        $Thld_{Size}$: Minimum number of pixels in the region
//        $r_1$: Circle radius of the structuring element of the dilation operation used in the
//             candidate region extraction step
//        $r_2$: Circle radius of the structuring element of the dilation operation used in the region
//             filtering step
//        L: Downsampling rate of a region
// Output: True if the input video is detected as a retroflexion video
//              False, otherwise
ISRETROFLEXIONVIDEO (video, $EOI_{video}$ , $Thld_R$ , $Thld_H$, $Thld_{Dilate}$ , $Thld_{Size}$, $r_1$, $r_2$, L)
1:    // Analysis images after the end of insertion frame
2:    fId ← $EOI_{video}$
3:    // endOfProcedureFrame: The last frame of the procedure
4:    while (fId < endOfProcedureFrame) do
7:        // Preprocess an image
8:        // allRegion[]: Array storing the information of detected regions in video.clearFrame
9:        // nRegion: Number of regions in video.clearFrame[]
10:       allRegion[], nRegion
11:       ← IMAGEPREPROCESSING( video.clearFrame[fId], $Thld_R$ , $Thld_H$, $Thld_{Dilate}$ , $Thld_{Size}$,
12:                                   $r_1$, $r_2$)
13:       // Identify each region in an image as a retroflexion region or a non-retroflexion region
14:       for rID ← 0 to nRegion do
15:           // Detect video as a retroflexion video if we detected at least one retroflexion region
16:           // in the video
17:           if ISRETROFLEXIONREGION(allRegion[rID], L) == true then
18:               return true
19:           end if
20:       end for
21:       fId ← fId+1
22:   end while
23:   return false
```

Figure 5.12: Pseudo code of the retroflexion video detection algorithm

5.5 Experimental Setup and Results

We implemented our algorithm in MATLAB. We built classifiers using MATLAB statistical toolbox. We performed a number of experiments on an Intel Xeon 3.86 GHz PC with 4GB of RAM and Windows 2003 Server.

5.5.1 Experimental Data

We randomly selected *75* colonoscopy videos from a test bed generated using OLYMPUS endoscopes during routine screening colonoscopy. This set includes *64* retroflexion videos and *11* non-retroflexion videos. We extracted *720-by-480* resolution JPEG images under the frame rate of *1 fps* (frame per second) in the withdrawal phase of each video. The beginning frame of the withdrawal phase was calculated using our algorithm in [77]. We then ran our blurry frame removal algorithm [70] to remove out-of-focus images. The rest of the images were used in our experiments. Our domain expert marked the ground truth of the retroflexion images and non-retroflexion images.

We split *75* videos into *25* training videos and *50* testing videos as described in Table 5.1. We included all *11* non-retroflexion videos in our test set. *1) Training set:* using all images from the entire withdrawal phase, we obtained a small portion of retroflexion regions compared to non-retroflexion regions. We calculated the average ratio of the number of samples between two classes in the training data. We obtained the ratio value as *1:43*. The classification on this imbalanced data set is a typical challenging problem in machine learning research [78]. As a result, we did not successfully train our classifiers using the images from the entire withdrawal phase. Based on the domain knowledge, in the vast majority of colonoscopies, the rectum is inspected during the last minute (or last *45* seconds). Some exceptions happen when other operations (e.g., biopsies) are performed after retroflexion in the rectum. For these cases, retroflexion occurs during the last *1-2* minutes in the video. We trained our classifiers using images obtained during the last *45* seconds of the rectum inspection. *2) Testing set:* We used all images in the withdrawal phase as testing data. *3) Parameter value settings:* We set all parameter values (Table 5.2) based on experiments on the training data. The performance is not sensitive to a small variation of these parameter values.

Table 5.1: Description of Images Used in Our Experiment for Retroflexion Detection

	Training Set		Testing Set		
Total	RT Videos		RT Videos		NRT Videos
# videos	25 (last 45 seconds)		39		11
Total	RT	NRT	RT	NRT	NRT
# images	129	877	306	18835	3788

RT- Retroflexion, NRT - Non-retroflexion, # - the number of images or videos

Table 5.2: Parameters and Values Used for Retroflexion Detection

Parameter	$Thld_R$	$Thld_H$	$Thld*_{size}$	$Thld_{Dilate}$	r_1	r_2	r_3	β	Q	ξ
Value	100	0.1	3000 pixels	0.6	5 pixels	15 pixels	45 pixels	0.5	5	1.5

* Parameters sensitive to an input image size

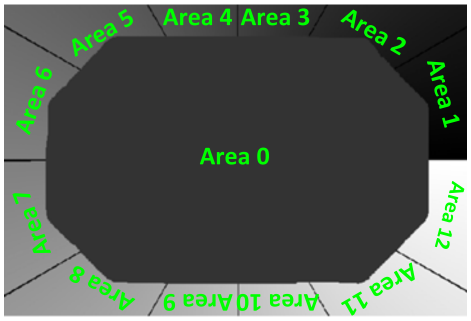
5.5.2 Experiments on Region Classification

We selected our features using a forward feature selection method [80]. We start with selecting an initial feature by testing each individual feature $v_1, v_2, v_3, \ldots, v_G$ from all G features and outputting the feature vector (v_i) which gives the best performance. We selected the next feature by concatenating v_i with each of the remaining G-1 features to be new feature vectors and outputting the feature vector (v_i, v_j) which performs the best among these new feature vectors. We repeated the above step to add a new feature at each step and stopped until the next best concatenated feature vector negatively contributes to the performance. We started with evaluating *15* RSL features and *11* ECSP features and selected *11* RSL features and *6* ECSP features as presented in this chapter. Based on extracted features, the retroflexion/non-retroflexion image classification problem is cast into the problem of training and testing these features to classify whether the regions are retroflexion or non-retroflexion regions. We evaluated with different features and classifiers for retroflexion/non-retroflexion region classification, including 1) RSL features, 2) ECSP features and 3) the concatenated RSL and ECSP features.

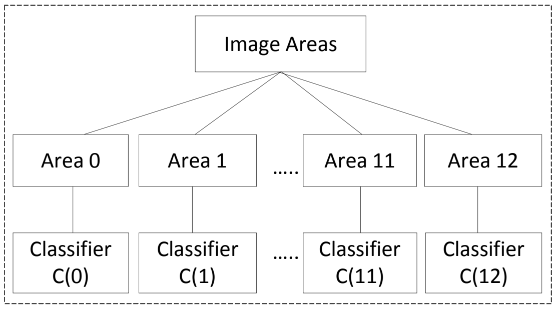
RSL features: The simplest way using RSL features is training and testing them with a machine learning classifier on the feature vector $V^{(RSL)}$ represented by

$$V^{(RSL)} = (x_c, y_c, x_t, y_t, x_h, y_h, |\lambda_1(\Sigma)|, |\lambda_2(\Sigma)|, \frac{|\lambda_2(\Sigma)|}{|\lambda_1(\Sigma)|}, \theta).$$

However, we found that splitting an image into non-overlapping areas, and training multiple classifiers, one for each area resulted in a better performance than training a single classifier. Therefore, we split an image into *N* overlapping areas (Area *0*, Area *1*,…, Area *N*-1) where *N* = *13* in our case (Figure 5.13 (a)). A_{inside} denotes the Area *0*. The areas *1* to *12* are numbered in a counter-clock-wise manner after splitting $A_{outside}$ by the angle, which results in *π/6* angle for each area. We chose the value of *N* based on our observation of the possible general scope insertion directions from the training videos.



(a)



(b)

Figure 5.13: (a) Pre-defined areas for $\boldsymbol{V}^{(\boldsymbol{RSL}^{(n_t)})}$ features classification; (b) The design of region classifiers using one classifier per area; *C(* $\boldsymbol{V}^{(\boldsymbol{RSL}^{(n_t)})}$*)* is simply denoted as *C(n$_t$)*.

We consider that a region belongs to the area n_t if its p_{head} is inside n_t. Let $V^{(RSL^{(n_t)})}$ represent the feature vector of the region belonging to the area n_t, where

$$V^{(RSL^{(n_t)})} = \left(id, x_c, y_c, x_t, y_t, x_h, y_h, |\lambda_1(\Sigma)|, |\lambda_2(\Sigma)|, \frac{|\lambda_2(\Sigma)|}{|\lambda_1(\Sigma)|}, \theta\right),$$

and *id* denotes the area *ID* where p_{tail} belongs to. We train one classifier *C(*$V^{(RSL^{(n_t)})}$*)* using $V^{(RSL^{(n_t)})}$ of retroflexion and non-retroflexion regions which belong to that area n_t in the training data. To test a feature vector of a region, we first determine its area n_t and classify $V^{(RSL^{(n_t)})}$ of this region using the classifier *C(*$V^{(RSL^{(n_t)})}$*)*. Figure 5.13 (b) illustrates our design for the classification on $V^{(RSL^{(n_t)})}$ features.

ECSP features: Not every retroflexion region has both $V^{(ECSP^{(+)})}$ and $V^{(ECSP^{(-)})}$ as we do not use the one of $f_i^+(r)$ or $f_i^-(r)$ that goes beyond A_{inside}. We consider a region as a retroflexion region if either $V^{(ECSP^{(+)})}$ or $V^{(ECSP^{(-)})}$ is classified as a retroflexion region. We simply use a notation $V^{(ECSP)}$ to represent either $V^{(ECSP^{(+)})}$ or $V^{(ECSP^{(-)})}$. Since ECSP features are not related to the insertion direction, we do not split an image into non-overlapping areas for training and testing $V^{(ECSP)}$.

Concatenated RSL and ECSP features: We concatenated $V^{(ECSP)}$ and $V^{(RSL)}$ as a single feature vector ($V^{(ECSP)} + V^{(RSL)}$). If a region has both $V^{(ECSP^{(+)})}$ and $V^{(ECSP^{(-)})}$, we concatenated each of them with the same $V^{(RSL)}$. As a result, we have at most two concatenated feature vectors ($V^{(ECSP)} + V^{(RSL)}$) per region, one is from $V^{(ECSP^{(+)})}$ and the other is from $V^{(ECSP^{(-)})}$. We consider a region as a retroflexion region if at least one feature vector ($V^{(ECSP)} + V^{(RSL)}$) from that region is classified as a retroflexion region. Similarly, we concatenated $V^{(RSL^{(n_t)})}$ and $V^{(ECSP)}$as ($V^{(ECSP)} + V^{(RSL^{(n_t)})}$) in the same way.

We investigated the effectiveness of the classification of retroflexion /non-retroflexion regions using Support Vector Machine (SVM) with linear kernel, SVM with Radial Basis Function kernel, J48 Decision Tree, and Generalized Linear Model with Poisson, Normal, and Binomial distributions on feature vectors $V^{(RSL)}$, $V^{(RSL^{(n_t)})}$, $V^{(ECSP)}$, ($V^{(ECSP)} + V^{(RSL)}$) and ($V^{(ECSP)} + V^{(RSL^{(n_t)})}$). The performance metrics were calculated using *10-fold cross-validation* on the training set. The parameter values of each type of classifiers were optimized to achieve the best performance. Table 5.3 showed the performance using *N=13* number of classifiers with each $C(V^{(RSL^{(n_t)})})$ trained on $V^{(RSL^{(n_t)})}$, and the performance single classifier $C(V^{(RSL)})$ trained on $V^{(RSL)}$. The results indicated that $C(V^{(RSL^{(n_t)})})$ outperformed $C(V^{(RSL)})$. For the same type of features, different classifiers produce similar performance. Among the compared classifiers, Decision Tree on classifying $V^{(RSL^{(n_t)})}$ showed the best *F-measure* [14] score– the harmonic mean of precision and recall. Table 5.4 shows the performance evaluated on $V^{(ECSP)}$, the concatenated features $(V^{(ECSP)} + V^{(RSL^{(n_t)})})$ and $(V^{(ECSP)} + V^{(RSL)})$. Table 5.4 and 5.3 indicate that either using concatenated features or splitting an image into non-overlapping areas can improve specificity for most types of classifiers. Therefore, in the following experiment on image/video classification, we used $V^{(RSL^{(n_t)})}$ instead of $V^{(RSL)}$. Note that we cannot directly compare results in Table 5.3 and Table 5.4. This is because we had more than one ECSP feature vector for a single retroflexion region in the experiments for Table 5.4 whereas the experiments for Table 5.3 did not. We also do not report the *F-measure* score for Table 5.4 for the same reason.

Finally, we chose Decision Tree as the classifier because it gave 1) the best *F-measure* score on RSL features and 2) the high specificity for most of classifiers on both RSL and ECSP features. Similar performance may be achieved using other types of classifiers.

Table 5.3: Evaluation of RSL Features Using Different Types of Classifiers

Classifier		GLM + Binomial			GLM + Poisson		
Performance Metrics		Sen.%	Spe.%	F-M	Sen.%	Spe.%	F-M
Feature	$V^{(RSL)}$	60.2	87.5	0.491	46.6	92.0	0.464
	$V^{(RSL(nt))}$	73.1	88.7	0.585	64.1	94.0	0.626
Classifier		GLM +Normal			SVM+Linear		
Performance Metrics		Sen.%	Spe.%	F-M	Sen.%	Spe.%	F-M
Feature	$V^{(RSL)}$	45.6	90.2	0.430	52.1	90.2	0.476
	$V^{(RSL(nt))}$	77.9	91.0	0.652	77.4	92.1	0.670
Classifier		SVM + RBF			Decision Tree		
Performance Metrics		Sen.%	Spe.%	F-M	Sen.%	Spe.%	F-M
Feature	$V^{(RSL)}$	70.0	94.0	0.664	67.8	88.0	0.544
	$V^{(RSL(nt))}$	74.5	92.1	0.653	72.7	93.6	0.672

Sen. - Sensitivity, Spe. - Specificity, F-M: F-Measure score, GLM - Generalized linear model, Binomial - Binomial distribution, Poisson - Poisson distribution, Normal - Normal distribution, SVM - Support vector machine, Linear - Linear kernel, RBF - Radial Basis Function Kernel

Table 5.4: Evaluation of ECSP Features and the Concatenated Features of RSL and ECSP Using Different Types of Classifiers

Classifier		GLM + Binomial		GLM + Poisson		GLM +Normal	
Performance Metrics		Sen.%	Spe.%	Sen.%	Spe.%	Sen.%	Spe.%
Feature	$V^{(ECSP)}$	75.3	73.7	70.8	75.3	74.1	77.3
	$V^{(ECSP+RSL)}$	74.7	81.4	73.0	82.8	81.0	75.9
	$V^{(ECSP+RSL(nt))}$	70.6	88.3	68.3	91.7	78.8	84.2
Classifier		SVM+Linear		SVM + RBF		Decision Tree	
Performance Metrics		Sen.%	Spe.%	Sen.%	Spe.%	Sen.%	Spe.%
Feature	$V^{(ECSP)}$	84.6	64.7	83.5	70.8	42.5	84.8
	$V^{(ECSP+RSL)}$	83.4	72.0	76.4	86.2	61.8	88.3
	$V^{(ECSP+RSL(nt))}$	74.4	89.8	72.9	91.7	66.1	92.1

Sen. - Sensitivity, Spe. - Specificity, GLM - Generalized linear model, Binomial - Binomial distribution, Poisson - Poisson distribution, Normal - Normal distribution, SVM - Support vector machine, Linear - Linear kernel, RBF - Radial Basis Function Kernel

5.5.3 Experiments on Image/Video Classification

We consider an image as a retroflexion image if it has at least one detected retroflexion region. We consider a video as a retroflexion video if we detect at least one retroflexion image in the video. We evaluated the performance for detecting retroflexion videos on *a)* the last *45 seconds*, *b)* the last *2 minutes*, and *c)* the entire withdrawal phase of each video. We conducted the first two tests because retroflexion occurs during the last minute (or last *45* seconds) for the vast majority of colonoscopies, or within the last *2* minutes for the exceptional cases when other operations like biopsies were performed as aforementioned. Compared with [6], our improved algorithm makes it possible to detect retroflexion in other parts of the colon during the withdrawal phase and to alert the endoscopist before the end of the procedure. In our data sets, the entire withdrawal phase contains on average *459* images per video. That means, our algorithm needs to achieve a high specificity for retroflexion video detection in the withdrawal phase. In the following paragraphs, we discuss the results of some main steps of our algorithm and features.

Candidate region extraction: Thresholding on *Red* and *Hue* values performed well for extracting candidate retroflexion regions. We only missed the gray color scope regions with very strong illumination. The region filtering step using $Thld_{size}$ and $Thld_{Dilate}$effectively rejected most noise regions. This step was robust to image blur.

Insertion direction calculation: Our calculation of the insertion direction is robust for most retroflexion images (Figure 5.14). It performs incorrectly when the scope is far from the camera and appears very small, in which the scope no longer has a longitudinal shape. Figure 5.15 shows the histograms of the estimated insertion direction angle θ in the training data. This angle appears as the first criterion (the root nodes) of the Decision Trees for Area 2 and Area 5.

To evaluate effectiveness of our features on classifying retroflexion/non-retroflexion images and videos, we setup the following tests using different classifiers and features. Table 5.5 shows the performance on *50* test videos.

1) $C(V^{(RSL^{(n_t)})})$
2) $C(V^{(ECSP)})$

3) $C(V^{(RSL^{(n_t)})} + V^{(ECSP)})$, and

4) a simple ensemble classifier $\mathrm{C}(V^{(RSL^{(n_t)})}) + \mathrm{C}(V^{(ECSP)})$ which classifies a region as a retroflexion region if both $\mathrm{C}(V^{(RSL^{(n_t)})})$ and $\mathrm{C}(V^{(ECSP)})$ classify that region as a retroflexion region.

Table 5.5: Performance of Retroflexion Image/Video Detection

Test Range	Classifiers and Features	Image Detection Performance		Video Detection Performance				
		Sen.%	Spe.%	Sen.%	Spe.%	Pre.%	Acc.%	# FL
45 sec.	C(RSL$^{(nt)}$)	58.0%	98.1%	84.6%	81.8%	94.3%	84.0%	0.48
	C(ECSP)	42.3%	98.5%	76.9%	81.8%	93.8%	78.0%	0.40
	C(RSL$^{(nt)}$+ECSP)	44.0%	99.1%	82.1%	90.9%	97.0%	84.0%	0.22
	C(RSL$^{(nt)}$)+C(ECSP)	36.8%	99.7%	71.8%	100.0%	100.0%	78.0%	0.10
2 min.	C(RSL$^{(nt)}$)	61.8%	98.3%	89.7%	63.6%	89.7%	84.0%	1.38
	C(ECSP)	44.5%	98.8%	84.6%	45.5%	84.6%	76.0%	1.06
	C(RSL$^{(nt)}$+ECSP)	47.2%	99.3%	87.2%	83.7%	91.9%	84.0%	0.64
	C(RSL$^{(nt)}$)+C(ECSP)	39.0%	99.8%	79.5%	90.9%	96.9%	82.0%	0.18
WTP	C(RSL$^{(nt)}$)	65.4%	98.4%	94.9%	18.2%	80.4%	78.0%	6.92
	C(ECSP)	48.1%	98.5%	89.7%	9.1%	97.1%	72.0%	6.08
	C(RSL$^{(nt)}$+ECSP)	49.6%	98.9%	92.3%	18.2%	80.0%	76.0%	4.50
	C(RSL$^{(nt)}$)+C(ECSP)	41.7%	99.8%	84.6%	72.7%	91.7%	82.0%	1.06

#FL – The average number of false alarm images per video (on average of 459 images per video); Sen. – Sensitivity, Spe. – Specificity, Pre. – Precision, Acc. – Accuracy; 45 sec. – test the last 45 seconds; 2 min. – test the last 2 minutes; WTP – test the withdrawal phase.

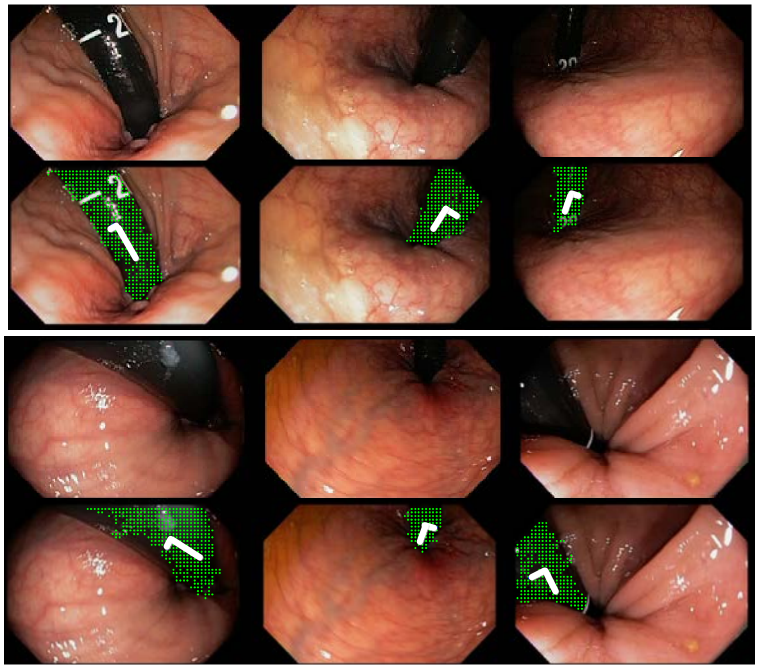


Figure 5.14: Retroflexion images and corresponding downsampled regions in the next row; the white arrows show the directions of eigenvectors: $e_1(\mathbf{\Sigma})$ and $e_2(\mathbf{\Sigma})$.

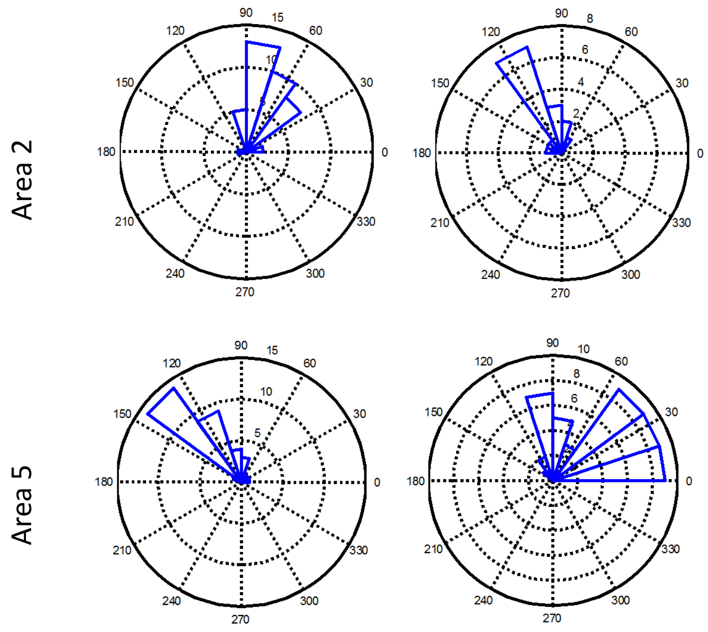


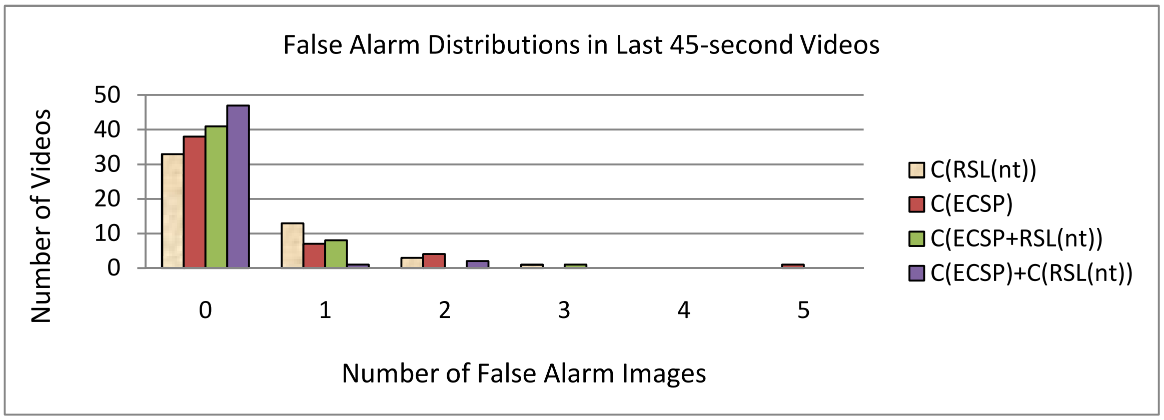
Figure 5.15: The histograms representing the distributions of the feature θ of retroflexion regions and non-retroflexion regions; each triangle shows the count of instances in the training data. Top row: Area 2; Bottom row: Area 5; Left column: retroflexion regions; Right column: non-retroflexion regions.

Retroflexion/non-retroflexion image classification: We classified an image as a retroflexion image if it had at least one detected retroflexion region. Otherwise, we classified it as a non-retroflexion image. The ECSP resulted in higher specificities but lower sensitivities than RSL. The concatenated ECSP and RSL features offered a trade off performance comparing with the performance of single classifiers using ECSP or RSL features alone. The sets of retroflexion images detected using a single classifier on ECSP features alone and on RSL features alone were significantly overlapping. However, most of false alarm images detected using ECSP features were stool or dye images, whereas most of false alarm images detected using RSL features were dark lumen images. This is because most of stool and dyes do not have good edges, while the shapes or/and locations of most of dark lumen regions are different from these of endoscope regions. As a result, the ensemble classifier using both classifiers significantly reduced the number of false alarm images with a slightly lower sensitivity than a single classifier on ECSP features alone. The ensemble classifier achieved the highest specificities compared with other single classifiers. In conclusion, the false alarm images were due to 1) dark lumen images and 2) images with stool or dyes. The miss rate of retroflexion images was not correlated to the duration of the operation. For example, most retroflexion images extracted from short retroflexion operations were correctly detected. The significant factor determining the miss rate is the quality of a retroflexion image. Our ensemble classifier failed to detect low quality retroflexion images. The endoscopes in these low quality retroflexion images 1) are covered by water, 2) or are with low illumination, 3) or appear in a too small area without a longitudinal shape.

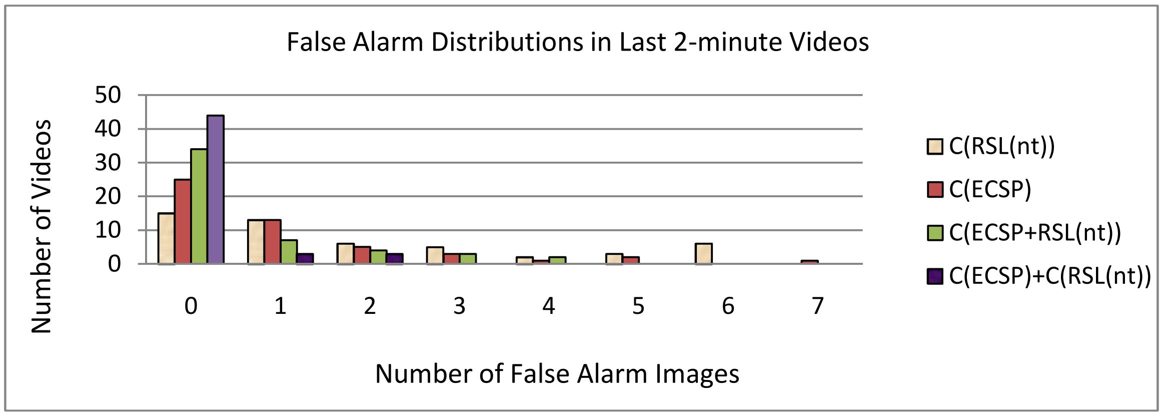
Retroflexion video detection: The ensemble classifier can effectively detect retroflexion videos. For the test on the last *45* seconds of each video, the specificity of the image classification increased to *99.7%* (*0.1* false alarm images per *45-second* video). As a result, we obtained *100%* (*11/11)* specificity, *100.0% (39/39)* precision and *78.0% (39/50)* accuracy on the retroflexion video detection. For the test on the last *2* minutes of each video, the specificity of the image classification also increased to *99.7%* (*0.18* false alarm images per *2-minute* video). As a result, we obtained *90.9%* (*10/11)* specificity, *96.9% (31/32)* precision and *82.0% (41/50)* accuracy on the retroflexion video detection. For the test on the withdrawal phases, the specificity of the image classification increased to *99.8%* (*1.06* false alarm images on the withdrawal phase per video). As a result, we obtained *72.7%* (*8/11)* specificity, *91.7% (33/36)* precision and *82.0% (41/50)* accuracy on the retroflexion video detection. Therefore, compared with the other single classifiers, the ensemble classifier significantly improved the specificity from *18.2%* to *72.7%* on the retroflexion video detection, and slightly improved the precision and accuracy. Figure 5.16 shows the distributions in terms of number of false alarm images per video. We can observe that for the tests on the last *45-second* and *2-minute* videos, the vast majority number of videos had no false alarm images. For the test on the withdrawal phases, the ensemble classifier significantly reduced the number of false alarm images per video to *0* or *1* false alarm images for the majority number of videos.

Analysis Time: The average execution time of our code is *0.46 seconds* per image. The main time cost is the dilation operations in the image preprocessing part. The calculation of PCA and RSL feature extraction only costs *30 milliseconds* per image. The downsampling of pixels significantly reduced the calculation time for the PCA. The calculation of edgeless ECSP and its feature extraction under the prior knowledge of the endoscope insertion direction only costs *26 milliseconds* per image. The downsampling of pixels reduced about half of the calculation time for the edgeless ECSP feature extraction.

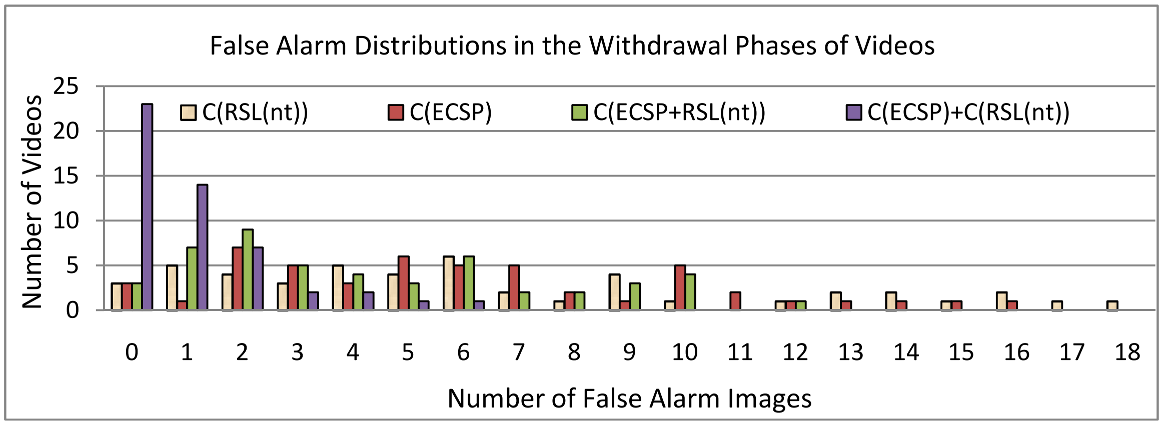
There is no existing technique with which we can compare our method. We experimented with texture features – Local Binary Pattern (LBP) and Opponent Color-LBP (OC-LBP) in [17]. LBP and OC-LBP features do not discriminate well between dark lumen and black scope regions. Most of dark lumen regions are detected as false regions. Therefore, these techniques did not give any comparable performance to that of our proposed technique. Our method is very promising for a near real-time retroflexion detection. An example of a clinical use case is as follows. A warning marker continually appears at the corner outside the endoscope FoV when retroflexion has not been detected in the withdrawal phase. If a retroflexion image is detected, the system should consecutively detect several retroflexion images during a period (e.g., *3* seconds) before it disables the warning marker. This is to recognize that good quality retroflexion is performed during the procedure.



(a)



(b)



(c)

Figure 5.16: The distributions in terms of the number of false alarm images per video detected on *50* test videos for different tests a) Test on the last 2 minutes, b) Test on the last 45 seconds, c) Test on the withdrawal phase.

5.6 Chapter Summary

In this chapter, we proposed a near real-time algorithm that can effectively detect a retroflexion operation during colonoscopy. This algorithm can be used as a new indicator of quality of colonoscopy. We investigated two types of features to detect appearance of endoscopes in colonoscopic images during retroflexion operations. These features are region shape and location (RSL) features that describe the location and shape of an endoscope region in an image based on Principle Component Analysis. We proposed an algorithm – edgeless edge cross-section profile (ECSP) calculation method to obtained features along the estimated endoscope edge normal directions without edge detection since edges of endoscope regions are not clear due to motion blur. We investigated different combination of classifiers, RSL features and edgeless ECSP features to classify retroflexion/non-retroflexion images and videos. Among these combination methods, the ensemble classifier using both ECSP and RSL features shows the best performance. Our near real-time system is promising for clinical use in terms of both the analysis time and detection rate. As future work, we will further optimize the analysis time by improving the computational complexity in the image processing step, and converting the analysis code from MATLAB to C/C++. We will also investigate whether the proposed RSL and ECSP features can be used to detect other instruments (e.g., cables) used for biopsies and therapeutic operations.

CHAPTER 6 POLYP IMAGE DETECTION AND SHOT EXTRACTION

This chapter introduces a new feature-based ECSP technique that segments multi-derivative ECSP functions into parts called ***part-based multi-derivative ECSP*** and obtains features on parts for polyp detection in colonoscopic video. We present two phases of our techniques as follows.

Phase 1: We introduce a part-based multi-derivative ECSP to detect the image with the appearance of polyp – ***polyp image***. We perform a study of the new features obtained from segmented parts for polyp image detection. We show the effectiveness of our technique by comparing it with the current state-of-the-art polyp detection techniques.

Phase 2: We further reduce the analysis time by simplifying features obtained from parts for polyp image detection. By combing the polyp image detection results with an existing edge tracking method, we further reduce the number of false positive edges at the image level and improve the overall detection rate. As a result, our system can correctly detect *97.7%* polyps in near real-time and does not generate false detected images for the *95.7%* duration in an entire colonoscopy procedure. The average number of false sub-shots per video is *36.2* and the average duration of each false sub-shot is less than *1.5* seconds. To our best knowledge, our polyp detection technique is the first technique that can effectively detect polyp image and extract polyp shot in near real-time (real-time under the frame rate of 10 fps) running on a modern PC. Our system is very promising for clinical use to assist the endoscopist by providing visual feedback of potential polyps during routine screening colonoscopy.

6.1 Introduction

Automated polyp detection in colonoscopy is challenging. 1) Endoscopic images can be blurry due to motion of the capturing camera or water injection, maybe under or over illuminated, or contain strong light reflection spots. 2) Polyps vary in their appearance, shape, size, amount of protrusion, and location in the colon. The same polyp may appear very differently in different images due to amount of colon insufflation, degree of colon muscular contraction, viewing angle, and distance from the capturing camera. For instance, a close inspection of a polyp results in a large polyp region whereas a distant inspection of the same polyp results in a small polyp region. Furthermore, polyps may be occluded by stools or therapeutic instruments. Existing computer-aided techniques utilize features such as texture, pixel color, geometric shape of edges, or combinations of these features together with machine learning classifiers for polyp detection. These techniques were summarized in Section 2.1.3.

We previously introduced the feature-based ECSP method for appendiceal orifice image detection in Chapter 4. We extracted features at some key positions (e.g. zero-crossing positions) of ECSP and its gradient profile of candidate edges. However, these features do not capture any global information of ECSP. Therefore, they are insufficient to represent complex characteristics of a protruding polyp such as the local shape of its ECSP, the protrusion, the texture or the smoothness of its region.

In this chapter, we propose two phases of methods for polyp detection. In the first phase, we introduce a new approach using ECSP and perform a study of new features for polyp image detection. Compared with the ECSP based method for appendiceal orifice image detection discussed in Chapter 4, we utilize the 2nd – order derivative of the multi-derivative ECSP and segment each of these profiles into parts. We model or extract features from different parts separately using methods suitable for individual parts. As a result, besides the local features obtained from key positions on ECSP (discussed in Chapter 4), we introduce features on parts which effectively describe some complex properties of polyp ECSP including the shape of the profile parts, texture of the polyp region, estimated polyp protrusion and the smoothness of a polyp surface. Finally, we combine the scores from modeling and feature scores as feature vectors for classification. Segmenting ECSP into parts gives flexibility to model or extract features from each part using the method component which is suitable for that part. Our technique is robust to some challenging issues of polyp detection and outperforms existing leading methods. The experimental results show that this method outperforms the current state-of-the-art techniques for polyp detection in terms of performance of free-response receiver operating characteristic curves and processing time on a test data set of over a thousand images (non-polyp and polyp images of 42 different polyps).

Our ultimate goal is to develop new technology that assists the endoscopist by providing visual feedback of a potential polyp during routine screening colonoscopy using a standard endoscope commonly used in practice. The endoscopist may use the feedback to improve the polyp detection rate during the procedure. Our second phase further improve the analysis time for polyp image detection by simplifying features obtained from part-based multi-derivative ECSP to the easily computation ones. We track polyp edges obtained from detected polyp image from Phase 1 using an existing edge tracking method. The combination of polyp edge detection and edge tracking reduce the number of false positive edges at the image level and improve the overall detection rate. Our system tested on *53* videos shows its promising for clinical use to assist the endoscopist by providing visual feedback of a potential polyp during routine screening colonoscopy in near real-time.

6.2 Part-based Multi-derivative Edge Profile Segmentation

We segment ECSP, its first-order profile, and its second-order profile into a total of 10 non-overlapping parts. Because the detected edge point r = 0 may not be the local minimum intensity point on F(r) due to the image smoothing, we search for the local minimum intensity point in a small 1-dimensional window centered at r = 0. We denote this local minimum intensity point as $\mathrm{r_0}$. The segmentation algorithm consists of two steps. See Figure 6.1 for an example and notations.

Backward segmentation from zero-crossing points: Let points $\mathrm{r = r_-^{(1)}}$ and $\mathrm{r = r_+^{(1)}}$ be two zero-crossing value points closest to $\mathrm{r = r_0}$ on $\nabla\mathrm{F(r)}$. Let points $\mathrm{r = r_-^{(2)}}$ and $\mathrm{r = r_+^{(2)}}$ be two zero-crossing value points closest to $\mathrm{r = r_0}$ on $\nabla^2\mathrm{F(r)}$. These points must satisfy both conditions below.

1) $r_-^{(1)} < r_0 < r_+^{(1)}$ and $r_-^{(2)} < r_0 < r_+^{(2)}$

2) $\nabla F(r_-^{(1)}) = \nabla F(r_+^{(1)}) = \nabla^2 F(r_-^{(2)}) = \nabla^2 F(r_+^{(2)}) = 0$

We segment $\mathrm{F(r)}$ at positions $\mathrm{c_1}$ and $\mathrm{c_2}$ which correspond to the points at $\mathrm{r_-^{(1)}}$ and $\mathrm{r_+^{(1)}}$ on $\nabla\mathrm{F(r)}$. Similarly, we segment $\nabla\mathrm{F(r)}$ at positions $\mathrm{c_3}$ and $\mathrm{c_4}$ which correspond to the points at

$\mathrm{r}_{-}^{(2)}$ and $\mathrm{r}_{+}^{(2)}$ on $\nabla^2\mathrm{F}(\mathrm{r})$. As a result, $\mathrm{F}(\mathrm{r})$ and $\nabla\mathrm{F}(\mathrm{r})$ are each divided into three parts in this step. Because the segmentation points on $\mathrm{F}(\mathrm{r})$ and $\nabla\mathrm{F}(\mathrm{r})$ are referred back from their corresponding derivative function, we call this backward segmentation. We do not segment the profile if its corresponding zero-crossing positions do not exist. Figure 6.1 (a-c) demonstrates the backward segmentation step.

Segmentation at key points: We further segment the parts obtained from the previous step. First, we segment $F(r)$ and $\nabla F(r)$ at $r = r_0$ as the illustrated points c_5 and c_6 in Figure 6.1 (d-e). Next, we segment $\nabla^2 F(r)$ at $r = r_{-}^{(2)}$ as shown at point c_7 in Figure 6.1 (f).

After the above steps, multi-derivative ECSP functions of an edge are segmented into *10* non-overlapping parts: *4* parts from $F(r)$, denoted as $\{S_t^{\nabla^0}\}_{t=1}^{4}$, *4* parts from $\nabla F(r)$, denoted as $\{S_t^{\nabla^1}\}_{t=1}^{4}$, and *2* parts from $\nabla^2 F(r)$, denoted as $\{S_t^{\nabla^2}\}_{t=1}^{2}$.

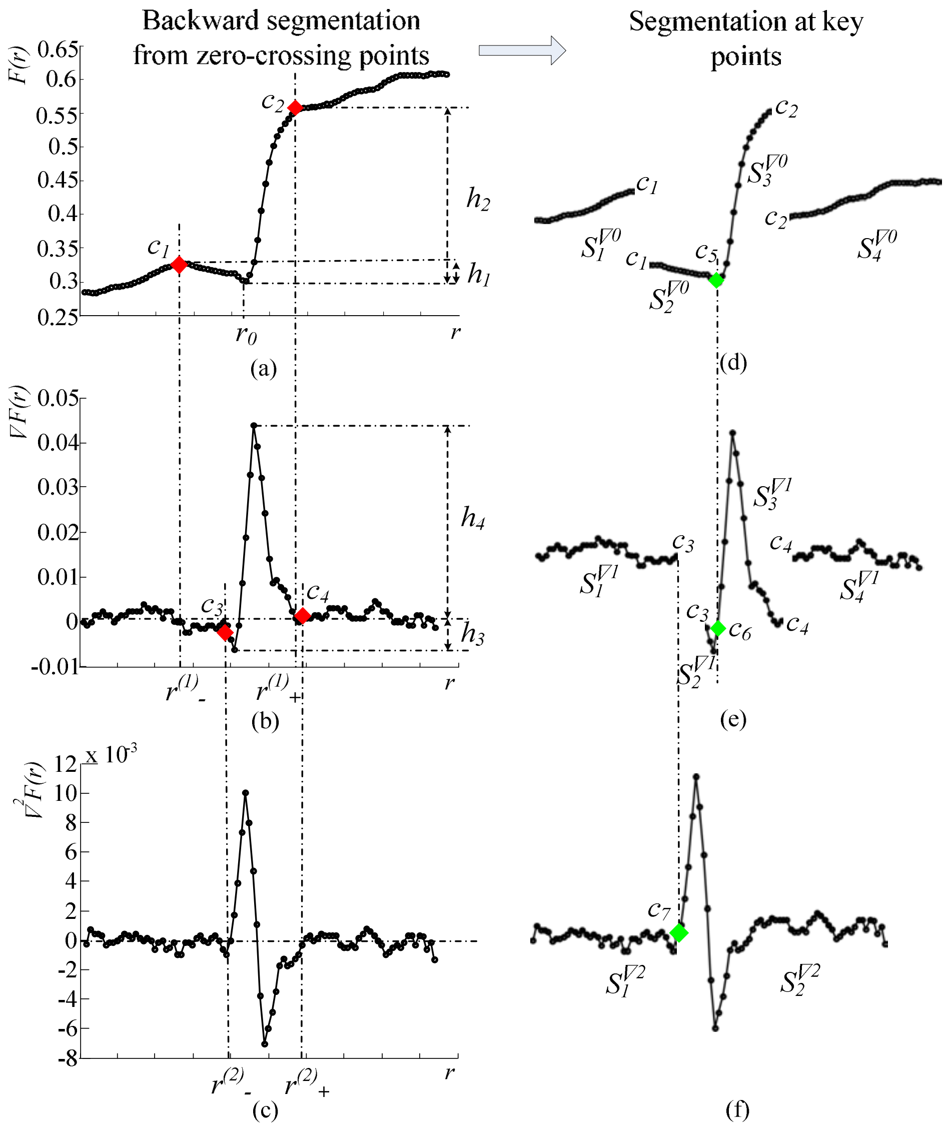


Figure 6.1: Example of application of segmentation steps to polyp ECSP functions. (a)-(c): Application of the backward segmentation step. Red rhombuses show the points between segmented parts. Dash-dotted lines link the red rhombuses on ECSP functions and their corresponding segmentation points at zero-crossing value positions on their derivative functions. (d)-(f): The segmentation at key points. Green rhombuses show the segmentation points in this step. Dash-dotted lines link the green rhombuses on ECSP functions and their corresponding segmentation points. The labels h_1, h_2, h_3 and h_4 correspond to the amount of intensity drop before crossing the edge, the amount of intensity return after crossing the edge, edge sharpness before crossing the edge and edge sharpness after crossing the edge. These features were introduced in Chapter 4.

6.3 Region of Interest Segmentation

Besides segmenting an edge, we also segment *ROI* in order to extract texture features. We segment a *ROI* corresponding to the edge into *4* non-overlapping sub-matrices, each corresponding to each segmented part of $F(r)$ ($\{S_t^{\nabla^0}\}_{t=1}^4$). See Figure 6.1 (d). We represent the segmented *ROIs* by $\{ROI_{convex}^1,\ ROI_{convex}^2, ROI_{concave}^3, ROI_{concave}^4\}$. We extract our texture features from some of these segmented *ROIs*.

6.4 Feature Extraction from Parts

Using part-based multi-derivative ECSP and *ROI*, we have flexibility to model or calculate features from a segmented part using the method which is suitable for that part. In this section, we introduce some features on parts which effectively describe some complex properties of polyp ECSP including texture of the region, shape of the parts, estimated polyp protrusion, the smoothness of a polyp surface, and intensity values on key positions on parts. We developed these features based on observations of characteristics of protruding polyps in colonoscopy and consultation with the domain experts.

6.4.1 Texture Features

Texture Features on $\boldsymbol{ROI_{concave}^3}$ and $\boldsymbol{ROI_{concave}^4}$: The protrusion of protruding polyps causes the intensity values on *F(r)* to abruptly increase from r_0 (the edge pixel position) to the right most point on $S_3^{\nabla^0}$ and then gradually increase afterward on $S_3^{\nabla^0}$ as shown in Figure 6.1 (d). We estimate the protrusion of polyps by capturing this tendency of intensity increase using texture features on $ROI_{concave}^3$ and $ROI_{concave}^4$ (Figure 6.2 (a-b)). First, we reduce the effect of small impulsive and speckle noises by smoothing the intensity values on each row and then on each column of $ROI_{concave}^3$and $ROI_{concave}^4$ using *Locally Weighted Scatterplot Smoothing* (LOWESS) with a tri-cube weight function [71]. LOWESS filter resists outliers by replacing them by linearly fitted points in a window. It is more suitable for preserving the tendency of intensity increase in the window caused by the polyp protrusion than most other filters. We also empirically found that LOWESS performs best among several smoothing methods such as average, median and moving average filters. Figure 6.2 (c) shows an example of $ROI_{concave}^{4}$ before and after the LOWESS smoothing.

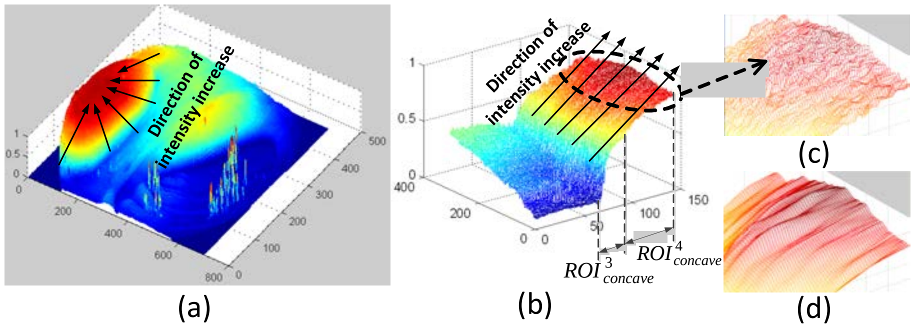


Figure 6.2: (a) 3D plot of intensity values of an image with a polyp (red color area). (b) 3D plot of intensity values for *ROI* obtained from the polyp region in (a). The intensity values increase on the polyp in the direction labeled by arrows in (a) and (b). (c) $\boldsymbol{ROI_{concave}^{4}}$ in (b) before LOWESS smoothing. (d) $\boldsymbol{ROI_{concave}^{4}}$ of (c) after LOWESS smoothing.

Next, we slide a *1x2* pixel window over $ROI_{concave}^{3}$ and $ROI_{concave}^{4}$ separately one pixel at a time to obtain texture features. We assign a binary score of *1* to that pixel if the pixel intensity value in the right window is greater than that in the left window, or a binary score of *0* otherwise. We experimented with different window sizes and chose a *1x2* pixel window due to its simplicity and effectiveness. Next, we obtain two binary matrices: one for $ROI_{concave}^{3}$ and the other for $ROI_{concave}^{4}$. Finally, we compute $V_{texture1}$ as the average of the binary values of the binary matrix of $ROI_{concave}^{3}$. Similarly, we compute $V_{texture2}$ from the binary matrix of $ROI_{concave}^{4}$. As a result, we have two texture features, each in the domain of [0, 1].

6.4.2 Shape Modeling

The global shapes of polyp ECSP functions do not match existing functions studied in [38] [39] [40] (e.g., a linear combination of several inversed Gaussian functions). However, the shape of the part $S_3^{\nabla^1}$ on $\nabla F(r)$ after profile segmentation skews to the left and looks similar to the shape of a Gamma probability distribution function (PDF). Figure 6.3 shows the shape of $S_3^{\nabla^1}$ of two example polyps. The segmentation of multi-derivative ECSP allows us to model each part separately without affecting the others. We model the shape of part $S_3^{\nabla 1}$ as follows.

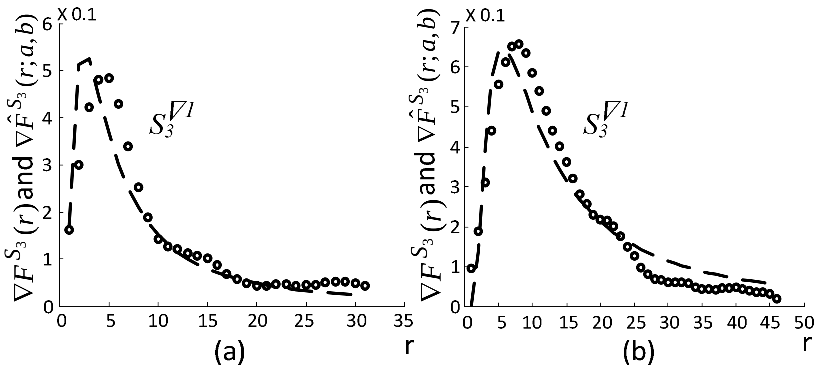


Figure 6.3: (a) and (b) show the part $\boldsymbol{S_3^{\nabla 1}}$ of two example polyps. Dots are points on function $\boldsymbol{\nabla F^{s_3}(r)}$ of $\boldsymbol{S_3^{\nabla 1}}$, the dashed lines are fitted Gamma PDF functions $\boldsymbol{\nabla \hat{F}^{s_3}(r; a, b)}$.

Let a function $\nabla F^{S_3}(r)$ denote the part $S_3^{\nabla 1}$ on $\nabla F(r)$. We define a Gamma PDF

$$\nabla \hat{F}^{S_3}(r; a, b) = r^{a-1} \frac{r^{a-1} e^{-\frac{r}{a}}}{b^a (a-1)!}, (a > 1, b > 0) \tag{6.1}$$

as the estimation of $F^{S_3}(r)$. The shape of the function $\nabla F^{S_3}(r)$ mainly depends on some particular points on the function. These points are 1) close to the function peak, or 2) with large function value differences from their neighbor points. See Figure 6.3. Therefore, we apply the *Weighted Levenberg-Marquardt* algorithm [79] to optimize the goodness of model fitting by setting higher weights for important pixels. The diagonal element W_{rr} in a diagonal weight matrix W presents the weight of pixel r defined as

$$W_{rr} = |\nabla F^{S_3}(r+1) + \nabla F^{S_3}(r)| \cdot |\nabla F^{S_3}(r+1) - \nabla F^{S_3}(r)|. \tag{6.2}$$

The first term of W_{rr} gives a large value for the pixels r and $r+1$ that are close to the function peak position. The second term measures the difference of function values between the pixel r and its right neighbor $r+1$. We compute a similarity score

$$V_{shape} = 1/(1 + \frac{Diff}{Comm}) \tag{6.3}$$

between the underlying model $\nabla F^{S_3}(r)$ and its estimated $\nabla \hat{F}^{S_3}(r; a, b)$ as the shape feature on part $S_3^{\nabla^1}$, where *Diff* is the difference between the areas under the curve of the two functions and *Comm* is the amount of the overlapping area between them.

6.4.3 Surface Smoothness Feature

Due to polyp protrusion, the intensity values on ECSP of polyps increase abruptly near their edge pixel positions. This property results in large amplitude values near the approximate edge pixel position $r = r_0$ on the derivative functions of the polyp ECSP (Figure 6.1 (b-c)). The surface of most polyps, in particular polyps less than *1-2* cm, is smooth. As a result, when comparing with the large amplitude values near $r = r_0$ for polyp edges, only very small oscillating amplitude values appear on non-edge positions of $\nabla^2 F(r)$ when $r > r_0$. However, if the surface of an object is noisy, as the calculation of a function's derivative amplifies the noise, the amplified noises are getting obvious on the profile functions when the order of the derivative is sufficiently large (e.g., the 2nd-order derivative). Therefore, some protruding non-polyp objects like stool show large amplitude values near both edge pixel position and non-edge pixel positions of $\nabla^2 F(r)$. $S_2^{\nabla^2}$ is the part which reflects the information of both edge pixel position and non-edge pixel positions on $\nabla^2 F(r)$ at the concave side of a polyp. We design a feature to capture the differences of amplitude values on $S_2^{\nabla^2}$ as follows.

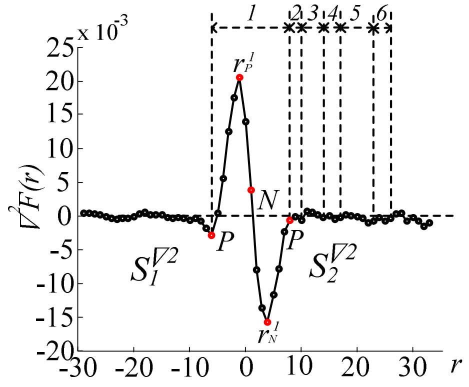


Figure 6.4: Six labeled *PNP* patterns on an example $\mathbf{S_2^{\nabla^2}}$ used for calculating surface smoothness feature

We define consecutive pixels as a *PNP* pattern on $S_2^{\nabla^2}$ if they contain *3* zero-crossing pixels in order: one *P-pixel*, one *N-pixel*, and another *P-pixel*. Specifically, *P-pixel* is a pixel at r whose function value satisfies $\nabla^2 F(r) < 0$ and $\nabla^2 F(r+1) > 0$. In other words, P-pixel is a zero crossing pixel from negative to positive. *N-pixel* is a pixel at *r* whose function value satisfies $\nabla^2 F(r) > 0$ and $\nabla^2 F(r+1) < 0$. In other words, N-pixel is a zero crossing pixel from positive to negative. First, we segment $S_2^{\nabla^2}$ into non-overlapping *PNP* patterns at all *P-pixel* positions. Figure 6.4 shows six such *PNP* patterns on $S_2^{\nabla^2}$, labeled from *1* to *6*. Next, we calculate the maximum and minimum function values of a *PNP* pattern as its amplitude, represented by a pair-value (e.g., function values at r_P^1 and r_N^1 labeled in Figure 6.4). Finally, we apply *k-means* clustering algorithm [75] on all pair-values of amplitude using Euclidean distance among the pair-values to cluster these *PNP* patterns into *2* groups: g_1 and g_2, where g_1 is the group with the number of patterns less than or equal to those in g_2.

If an object has smooth surface, then there is only one large amplitude pair-value near edge pixel position on $S_2^{\nabla^2}$. That is the group g_1 should contain only one *PNP* pattern – the pattern that is closest to $r = r_0$. Otherwise, g_1 should contain other *PNP* patterns caused by noise. We use a binary score to represent a smoothness feature denoted as $V_{smoothness}$. We assign *1* to $V_{smoothness}$ when the surface of the corresponding object is considered smooth. That is, if 1) the *PNP* pattern closest to $r = r_0$ belongs to g_1, and 2) g_1 contains at most q pair-values of amplitude. Otherwise, we assign *0* to $V_{smoothness}$. We set q as *2* to allow the maximum of *1* noisy *PNP* pattern in g_1.

6.4.4 Key Points Based Features

We extracted *6* features on key positions and *3* of them (h_2, h_3, and h_4) were introduced in Chapter 4. These features include the strength and the sharpness of edge. Table 6.1 shows the description of these features.

Table 6.1: Features on Key Positions of Parts Used For Polyp Detection

Type	Notation	Part(s)	Feature description
Edge strength	h_1 - h_2	$\{S_2^{\nabla^0}, S_3^{\nabla^0}\}$	Difference of the amount of intensity drop and increase before and after crossing the edge
	h_2	$\{S_3^{\nabla^0}\}$	Amount of intensity return after crossing the edge
	h_2/h_1	$\{S_2^{\nabla^0}, S_3^{\nabla^0}\}$	Ratio of the amount of intensity drop before crossing the edge and the amount of intensity return after crossing the edge
Edge sharpness	h_3	$\{S_2^{\nabla^1}\}$	Sharpness of edge before crossing the edge
	h_4	$\{S_3^{\nabla^1}\}$	Sharpness of edge after crossing the edge
	h_4/h_3	$\{S_2^{\nabla^1}, S_3^{\nabla^1}\}$	Ratio of edge sharpness before and after crossing the edge

Note: Notations of these features are indicated in Figure 6.1 (a-b).

6.4.5 Feature Vector

Our final feature vector V has ten features ($V_{texture1}$, $V_{texture2}$, V_{shape}, $V_{smoothness}$, h_1 - h_2, h_2, h_3, h_4, h_2/h_1, h_4/h_3). While additional features may be useful for detecting protruding polyps, we focus on these ten features in this study. These features are selected using a forward feature selection method [80]. To select features from m candidate features, we evaluate the performance of features based on *10*-fold cross validation on training data. We start with selecting an initial feature by testing each individual feature v_1, v_2, v_3… v_m from all m features and outputting the feature vector (v_i) which gives the best performance. We select next feature by concatenating v_i with each of the rest of m-*1* features to be new feature vectors and outputting the feature vector (v_i, v_j) which performs the best among these new feature vectors. We repeat the above step to add a new feature at each step and stop until the next best concatenated feature vector negatively contributes to the performance. Our selected features using this method show effectiveness for polyp detection based on our experiments (discussed in Section 6.5).

6.5 Experiments on Polyp Image Detection

We implemented the analysis software in MATLAB. We ran all experiments on an Intel Xeon 3.80 GHz CPU, 4 GB RAM workstation running Microsoft Windows Server 2003 operating system.

6.5.1 Experimental Setup

Data Set: We randomly selected *46* de-identified video files captured during routine screening colonoscopy at the image resolution of *720x480* pixels at *29.97* frames per second (fps). Each file contains only one colonoscopy procedure in its entirety. Twenty-eight of these videos were captured using FUJINON endoscopes, and the remaining eighteen video files were captured using OLYMPUS endoscopes. The colons seen in these videos had little stool inside them. We extracted from these video files *69* smaller clips: *50* clips with polyps and *19* clips without any polyps. The *50* clips represent *50* different polyps (one polyp per clip). Each polyp clip has consecutive frames showing the same polyp appearing at various *viewing angles, light conditions*, and *scales.* Pit patterns [56] of these polyps are not clearly seen since magnifying endoscopes were not used.

Our non-polyp clips contain images with most types of objects often seen in colonscopy, including *blood vessels, colon wall, colon folds, stool, retroflexion, appendiceal orifices* and *diverticula* [1]. For each clip, we extracted the images in JPEG format with a resolution of *720x480* pixels at the frame rate of *1* fps. We removed blurry images using blurry frame removal software [70], leaving the remaining clear (in-focus) images for the experiments. The blurry frame removal algorithm in [70] considers a frame as blurry if it does not have sufficient number of connected edge pixels detected using Canny edge detector. Totally, we obtained *1025* images each with a polyp in it and *488* images without any polyps on them. Two domain experts marked and agreed on the ground truth of all polyp regions in the images. Fig. 10 shows example ground truth polyp regions.

Table 6.2: Description of Data Sets Used for Polyp Image Detection

Category	# Polyp clips (# images, # edges)	# Non-polyp clips (# images, # edges)	Total
Training	8 (160, 844)	6 (87, 1305)	14 (247, 2149)
Testing	42 (865, 4983)	13(401, 1813)	55 (1266, 6796)
Total	50 (1025, 5827)	19 (488, 3118)	69 (1513, 8945)

Table 6.3: Number of Polyp Images of Each Video Clip in the Training Set

Video clips captured using FUFJINON endoscope					
Clip ID	1	2	3		
# Polyp images	19	10	36		
Video clips captured using OLYMPUS endoscope					
Clip ID	4	5	6	7	8
# Polyp images	20	43	8	18	6

Table 6.4: Number of Polyp Images of Each Video Clip in the Testing Set

Video clips captured using FUFJINON endoscope										
Clip ID	1	2	3	4	5	6	7	8	9	10
# Polyp images	9	21	4	17	8	17	7	26	11	8
Clip ID	11	12	13	14	15	16	17	18	19	20
# Polyp images	18	13	43	58	21	2	19	16	6	8
Video clips captured using OLYMPUS endoscope										
Clip ID	21	22	23	24	25	26	27	28	29	30
# Polyp images	19	23	30	45	8	7	53	17	8	10
Clip ID	31	32	33	34	35	36	37	38	39	40
# Polyp images	37	20	37	11	50	41	11	26	28	11
Clip ID	41	42								
# Polyp images	12	30								

Table 6.5: Polyp Categories in the Testing Set

Category	On colon wall	On colon folds	Near flat	With stool	All
# Images	437	428	130	196	865
# Clips	20	22	6	8	42

We split the polyp clips into non-overlapping training and testing sets. Table 6.2 describes the data sets including the number of clips, images and extracted edges after the image preprocessing step in each category. Table 6.3 and Table 6.4 describe the numbers of polyp images in each video clip in the training data and the testing data, respectively. We included eight polyp clips in the training set, which covered all categories of polyp appearance in Table 6.5 (discussed in more details in a later paragraph).

Parameter value settings: All parameter values were determined based on experiments with the training set. The parameter values may vary for different image resolutions in other data sets.

Image pre-processing and ECSP calculation: We extracted edges using Canny edge detector with the low threshold, high threshold, and standard deviation of Gaussian filter of *0.05*, *0.1*, and *3.0*, respectively. Next, we removed small branches on these edges. We discarded an edge if 1) it had less than *20* edge pixels (smaller than the number of pixels in the window D_i) or 2) it was an approximately linear edge. An edge is considered linear if the least squares fitting error of its pixel locations fitted to a linear line is less than ε ($\varepsilon = 8$ pixels in our case) using a linear regression as in Chapter 4. We cut an S-like shape edge that typically occurred when a polyp was connected with a colon fold into several C-like shape edges by tracking edge pixels and examining the change of the edge tangent directions. Because our data set has clear images obtained based on the quality of their Canny edges, we successfully extracted real polyp edges for the vast majority of polyp images (*1018* of *1025*) with these parameters. For ECSP calculation, the sliding window D_i of sizes between *10x10* and *25x25* pixels gave good performance. The valid range of λ was from *0.5* to *1.0*. In subsequent experiments, we set the size of D_i to *15x15 pixels* and λ to *0.5*, and limit λ/κ to a maximum value of *80* pixels. The one-dimensional window size centering at $r=0$ for finding the minimum intensity point r_0 in the window was *1x21*.

Feature normalization: The values of most of our features, $V_{texture1}$, $V_{texture2}$, V_{shape}, $V_{smoothness}$, h_2, h_3, h_4 are in the range of [0, 1]. The feature h_1 - h_2 is in the range of [-1, 1]. As the values of features h_2/h_1 and h_4/h_3 are in large scale ranges, we transformed the data of these two features using log functions log (h_2/h_1) and log (h_4/h_3). We then normalized each individual feature h_3, h_4, log (h_2/h_1) and log (h_4/h_3) by dividing it by its corresponding $C^{(d)}$, where $C^{(d)}$ is the absolute difference of maximum and minimum values for the feature *d* in the training data.

Performance metrics: We consider the minimum bounding rectangle of each edge obtained from Canny edge detector as one region. Let *DET_RLPLP_IMG*, *RLPLP_IMG*, *FL_REGION*, and *IMG* be the number of correctly detected polyp images, the number of real polyp images, the number of falsely detected regions (regions not overlapping with any real polyps), and the number of tested images, respectively. A polyp image is considered correctly detected if it has at least one correctly detected polyp region. A detected region is counted as a true positive if it overlaps with a ground truth polyp region. The number of wrongly detected polyp regions was counted as falsely detected regions (*FLR*). Regions that are falsely detected in true polyp images are also counted in FLR. We used the following performance metrics that were also used in [30].

$$True\ position\ rate\ (TPR) = \frac{DET\_RLPLP\_IMG}{RLPLP\_IMG}$$

$$Number\ of\ false\ regions\ per\ image\ (FLR) = \frac{FL\_REGION}{IMG}$$

To provide visual feedback of a detected polyp region, a high TPR is desirable. On the other hand, a low FLR is desirable since many false positives can distract the endoscopist. We used these metrics in the free-response receiver operating characteristic curves (FROC). Our TPR is not based on regions since our focus is not on detecting an accurate polyp contour.

Classifier selection: We investigated the performance of several classifiers: two-class Support Vector Machine (SVM) with radial basis function (RBF) kernel using Torch SVM library [81] and Generalized Linear Models (GLM) [82] with Binomial, Gaussian and Poisson distributions as kernels. We used feature vectors obtained from all images in the training and testing sets for a *10-fold cross validation* to obtain performance of these classifiers. We varied the standard deviation of Gaussian kernel for the SVM classifier, and the threshold values of probability for GLM classifiers to obtain FROCs. There are two more parameters for the SVM classifier [82], but they had little effect on FROCs. These different classifiers showed similar FROCs (plots omitted). SVM with RBF kernel and GLM with Gaussian kernel showed the best performance with the largest area under the FROC.

We used a two-class SVM classifier with RBF kernel as the classifier for our technique for the rest of the experiments. The two classes were polyp images and non-polyp images. In these experiments, we varied the standard deviation of the Gaussian kernel for the SVM classifier to train the classifier using all images in the training set and applied the trained classifier on the entire testing set to get TPR and FLR for each point on FROC.

6.5.2 Experiments with Different Features

Figure 6.5 shows FROC of individual features and the combination of all the features. Individual shape features, texture features, or *3* key features used in Chapter 4 showed similar performance. That is about *2/3* TPR comparing with the performance using all proposed features. The *6* key features presented in Table 6.1 or combination of smoothness, shape and texture features improved about *10%* TPR on average. The combination of all features (*10* features per edge) further improved about *10%* TPR on average and showed the best performance.

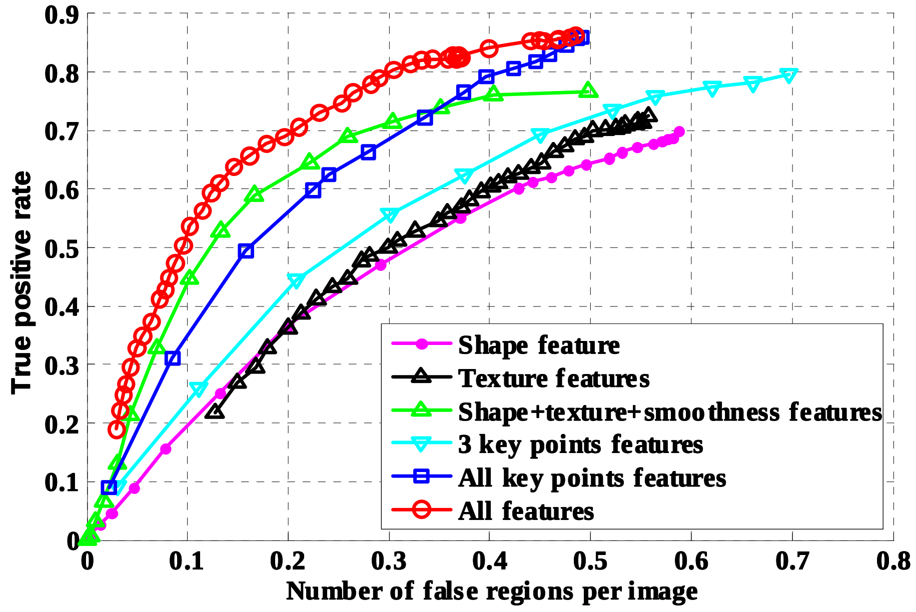


Figure 6.5: FROC curves of different features tested on all testing data. Shape feature is V_{shape}; texture features are $V_{texture1}$, $V_{texture2}$; smoothness feature is $V_{smoothness}$, *3* key points features are h_2, h_3, h_4 used in Chapter 4, and *6* key points features are the features presented in Table 6.1.

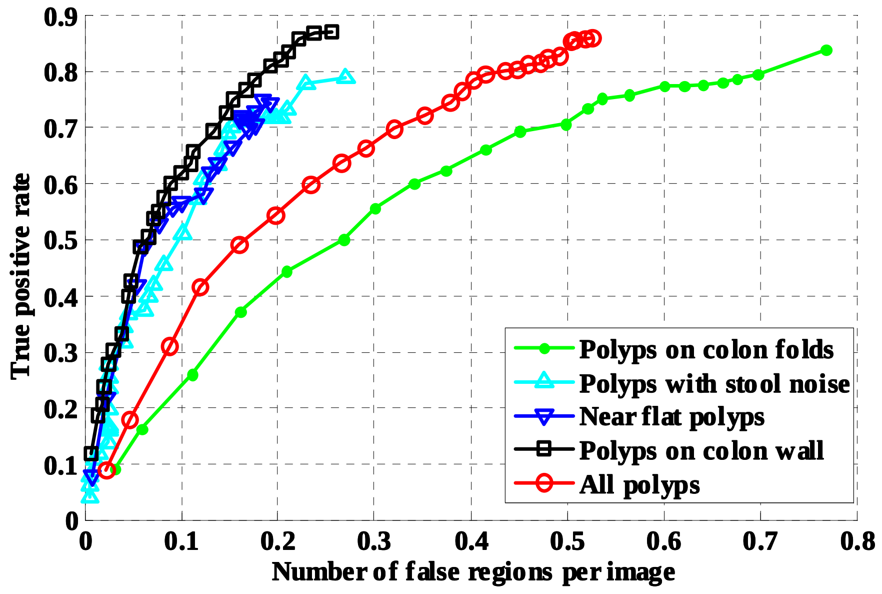


Figure 6.6: FROC curves for different types of polyps using the proposed features

6.5.3 Experiments with Different Types of Polyps

Table 6.5 shows polyp types based on their appearance: polyps on colon wall, polyps on colon folds, polyps with stool, and near flat polyps (polyps with little protrusion). A polyp image can belong to more than one category. For example, a near flat polyp image may also have stool.

Training and testing procedures were the same as in the previous experiment to evaluate effectiveness of different features. However, the TPR and FLR for each polyp category was calculated using detection results on polyp images in that category only. These polyp images have other edges besides polyp edges. Results of non-polyp images were excluded from Figure 6.6 since we already see the performance of our technique on the entire test set in Figure 6.5. Figure 6.6 shows the FROC of these types of polyps. The number of false regions per image is highest for the polyps on colon folds because some protruding colon folds look very similar to real protruding polyps. We obtained the highest TPR of *76%* for detecting near flat polyps. We found that these near flat polyps show similar patterns on parts as protruding polyps. Most stool was correctly rejected. Some polyps contain reflected smooth stool (e.g., water injected via the instrument) as falsely detected regions. Other false regions are caused by noises like light reflected spots and water. Our technique is robust to different viewing angles and polyp region sizes. Figure 6.8 shows some detection results under the TPR of *86.3%*. Detected edges are labeled by the minimum bounded rectangles of the edges. The examples show the complexities of the polyps in our data set. Under the TPR of *86.3%*, most mis-detected polyps are poorly illuminated or far away from the camera.

6.5.4 Performance Comparison with Existing Techniques

We compared our technique with two existing leading techniques using LBP and OCLBP features in [17] [50]. We chose these two techniques for comparison since OCLBP was shown to perform best in previous studies [17] [31]. Other newer techniques were only evaluated on small data sets and their performance was questioned in [11]. It is difficult to compare across different techniques using the same performance metrics. The LBP and OCLBP uses region texture in a sliding window while our technique uses features from edges. For comparison, in our technique, we consider the minimum bounding rectangle of a detected edge as a detected region representing that edge. The LBP and OCLBP algorithms slide a square window over an entire image. At each sliding step, the square region covered by the window is classified as a polyp candidate region or a non-polyp candidate region by a SVM classifier using the features obtained from that square region. Detected polyp candidate regions are sometimes connected in *8* directions. The union of these connected regions is considered as a single detected polyp region if it consists of least $Thld_{region}$ number of connected candidate polyp regions. Besides the parameters of SVM, we also varied $Thld_{region}$ from *1* to *7* for LBP and OCLBP. Since our technique does not use sliding windows, $Thld_{region}$ was not relevant. We used RBF kernel for SVM and *64* histogram bins for both LBP and intra- and inter-channel OCLBP patterns as in [17] [50]. The size of sliding window influences the performance of LBP and OCLBP features. Setting a window size too large will include many non-polyp pixels, resulting in ineffective classification. We investigated performance under three window sizes: *32x32*, *48x48,* and *64x64* pixels. The sliding window step was set to half the window size.

Figure 6.7 shows the comparison of these techniques. OCLBP outperforms LBP with the same window size. OCLBP with *48x48* pixels window size (denoted as OCLBP-*48x48*) shows the best performance comparing with LBP and OCLBP using the other window sizes for this data set. Our proposed technique outperforms these two methods in the following aspects. 1) Under the same true positive rate (*TPR*), the average number of false regions per image (*FLR*) of our technique is under *0.5* (Figure 6.5 and Figure 6.7), which is significantly lower than those of LBP and OCLBP. Furthermore, false regions generated by our technique are not equally distributed within non-polyp images. Most of false regions occur within images with *a)* strong light reflected spots, *b)* smooth stool, *c)* water, or *d)* protruding colon folds. For example, for an *81.4% TPR*, our technique has *0.32 FLR*. These false regions occur within only *20.5%* of non-polyp images. As a result, *79.5%* of non-polyp images do not have any false region under an *81.4% TPR*. However, OCLBP-*48x48* generates around *1.8 FLR* under the same *81.4% TPR*. These false regions are close to equally distributed on different types of non-polyp images. As a result, nearly every non-polyp image (*99.8%*) has at least one false region. 2) We observed that detected regions using OCLBP and LBP generally contained a large area of non-polyp pixels surrounding real polyp pixels: falsely detected regions were connected with real polyp regions and counted as a single correctly detected region. Compared with these sliding-window based methods, our method detects polyp edges, which reflects more accurate locations of the polyps. Thus, our results should be more meaningful in clinical practice. 3) Analysis time: Table 6.6 shows that the average time for our technique to analyze an image is *7.1* seconds. The bottlenecks are ECSP and *ROI* extraction. For LBP and OCLPB, the analysis time decreases as the window size increases. In contrast, our technique took around *1/3* of the time taken by the best performance using OCLBP-*48x48* and a few seconds less than that of the best performance using LBP.

Additional polyp detection algorithms [11] [30] were proposed after [17]. However, these recent techniques except our technique [16] were evaluated on data sets that were too small to convincingly validate the techniques. For instance, the algorithm in [30] provided an *86.2%* TPR and *1.26* FLR tested on only *37* images and took about *13* seconds on a similar workstation to process one image with around *1/4* size of our image size. The number of distinct polyps studied was not given. We did not directly compare with this technique as it used sliding-window based texture features which have similar drawbacks as using LBP and OCLBP. Our previous algorithm in [16] runs significantly slower than our proposed method and can miss a polyp whose entire shape does not fit an ellipse shape template well. Therefore, we did not compare with it.

Table 6.6: Average Analysis Time in Seconds per Image

Method	Module and its analysis time		Total time
	Module	Time	
ECSP	Image preprocessing	1.8	7.1
	ECSP, ROI extraction	4.0	
	Feature extraction, classification	1.3	
Method	Sliding window size		Total time
LBP	32x32		21.3
	48x48		9.7
	64x64		6.4
OCLBP	32x32		63.0
	48x48		27.9
	64x64		19.6

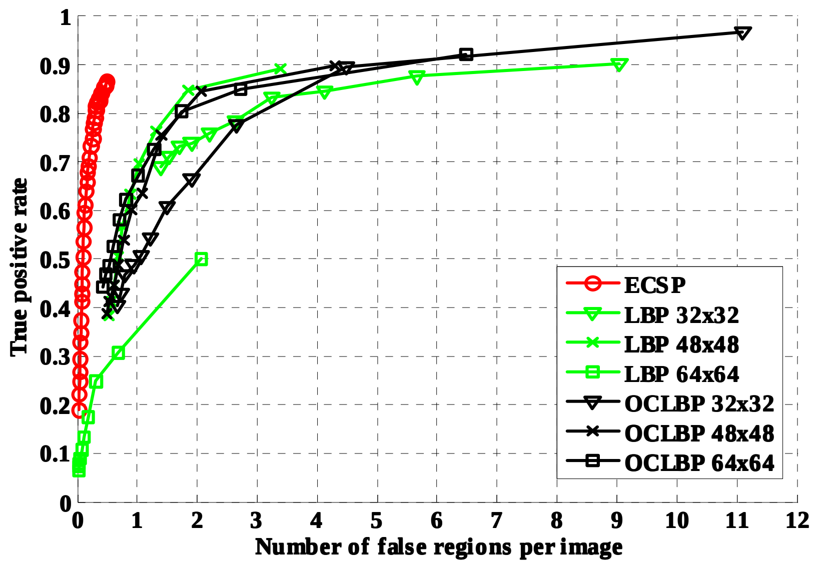


Figure 6.7: FROC curves of features obtained from part-based multi-derivative ECSP, LBP and OCLBP with different sizes of sliding windows; features were tested on a SVM classifier with RBF kernel.

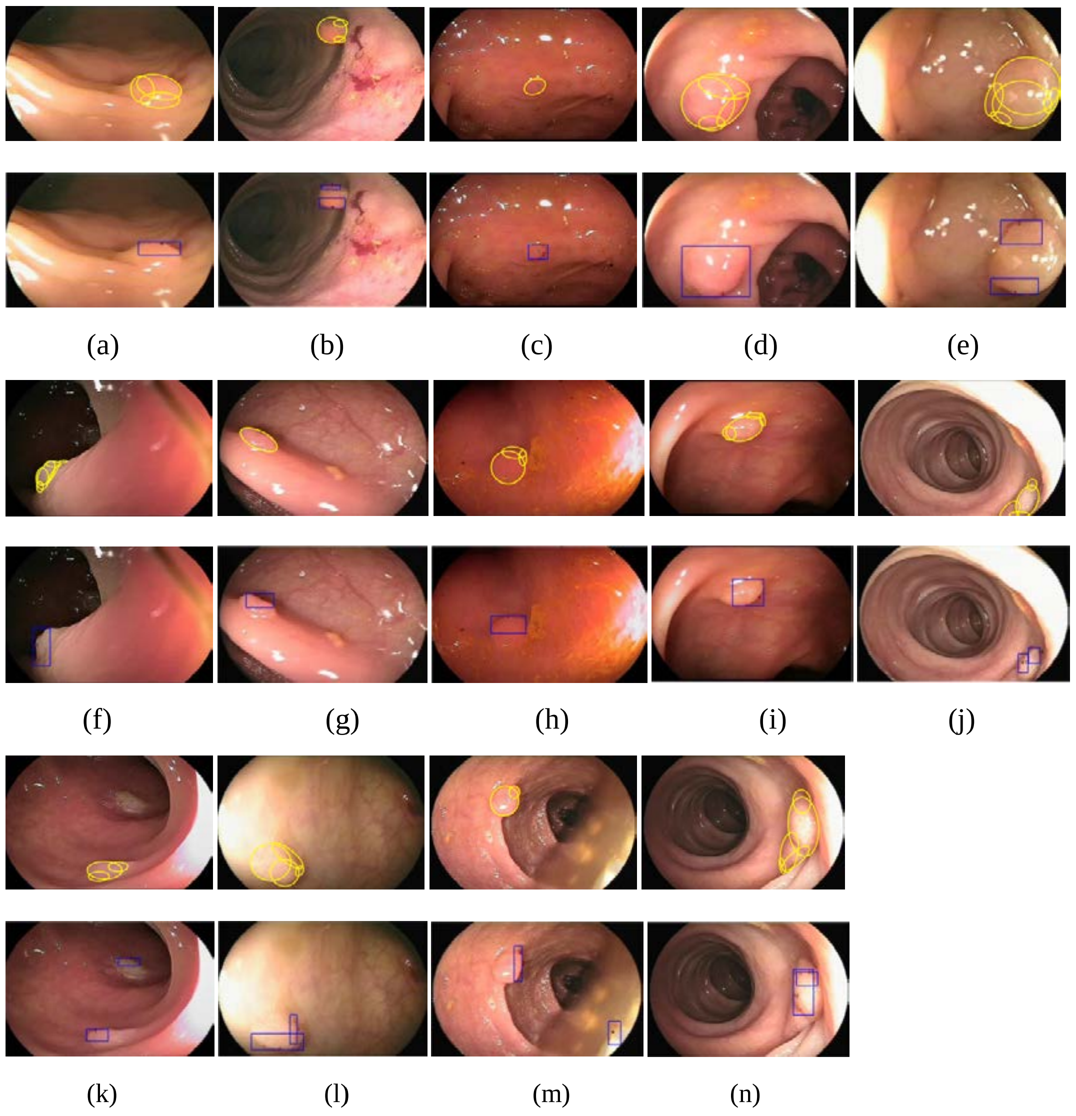


Figure 6.8: Examples of detected polyps: 1st and 3rd rows: ground truth is the area of the union of all marked ellipses; 2nd and 4th rows: detected results marked with the minimum bounded rectangle of the detected edges. (a) Polyp on colon wall. (b) Polyp on side of colon fold, with blood. (c) Small polyp on colon wall, near flat, facing the camera. (d) Polyp on colon fold, with stool, facing the camera. (e) Polyp on colon wall, facing the camera. (f) Polyp on colon fold. (g) Polyp on colon fold, with stool. (h) Small polyp on colon wall, near flat, with stool, facing camera. (i) Polyp on colon wall. (j) Small polyp on side of colon fold. (k) Polyp on colon fold, with stool. (l) Polyp on colon wall. (m) Polyp on colon fold, with water and stool. (n) Polyp on side of colon wall.

6.5.5 Discussion

Segmenting ECSP into parts gives flexibility to model or extract features from each part using the method component which is suitable for that part. Experimental results show that the proposed technique is robust to some challenging issues of polyp detection and outperforms existing leading methods. Our technique can effectively detect different types of protruding polyp appearance including polyps with little protrusion. It can mark detected polyp edges as visual feedback which is more precise and less intrusive than marking a large region as in a sliding-window based method. Furthermore, the false positives generated by our technique only appear in a small percentage of the number of images analyzed. We can further reduce the number of false positive edges to avoid overwhelming the endoscopist with unnecessary warnings by reducing the detection rate at the image level and using temporal information from consecutive frames (e.g., tracking of detected edges across frames) to improve the overall detection rate. Toward providing real-time feedback during live colonoscopy, we can already reduce the analysis time by code conversion to C/C++ and optimization on ECSP and feature extraction for the analysis rate of at least *1* frame per second.

6.6 Near Real-time Polyp Warning and Polyp Shot Detection

In this section, we further improve the analysis time for polyp image detection by simplifying features obtained from part-based multi-derivative ECSP. We combine polyp edge detection results with an edge tracking algorithm to reduce the number of false positive edges at the image level and improve the overall detection rate. Finally, we extract consecutive images showing appearance of detected polyp as semantic units in the video.

We call a sequence of consecutive images where each non-blurry image is a polyp image as a *polyp sub-shot*, and a single polyp sub-shot or a sequence of consecutive nearby polyp sub-shots including the non-polyp images between each two sub-shots together as a *polyp shot*. We consider that two consecutive polyp sub-shots are nearby if the duration between the ending frame of a sub-shot and the starting frame of the other sub-shot is smaller than a predefined threshold ($\boldsymbol{T_s}$). A single polyp sub-shot is considered as a polyp shot if it does not have any nearby sub-shot. Our software developed using C/C++ and Open Source Computer Vision (OpenCV) library can provide warning feedback on detected polyp sub-shots and extract polyp shots for endoscopists in near real-time under the frame rate of 10 fps. Figure 6.9 shows the flow chart components of our near real-time warning and polyp shot extraction system.

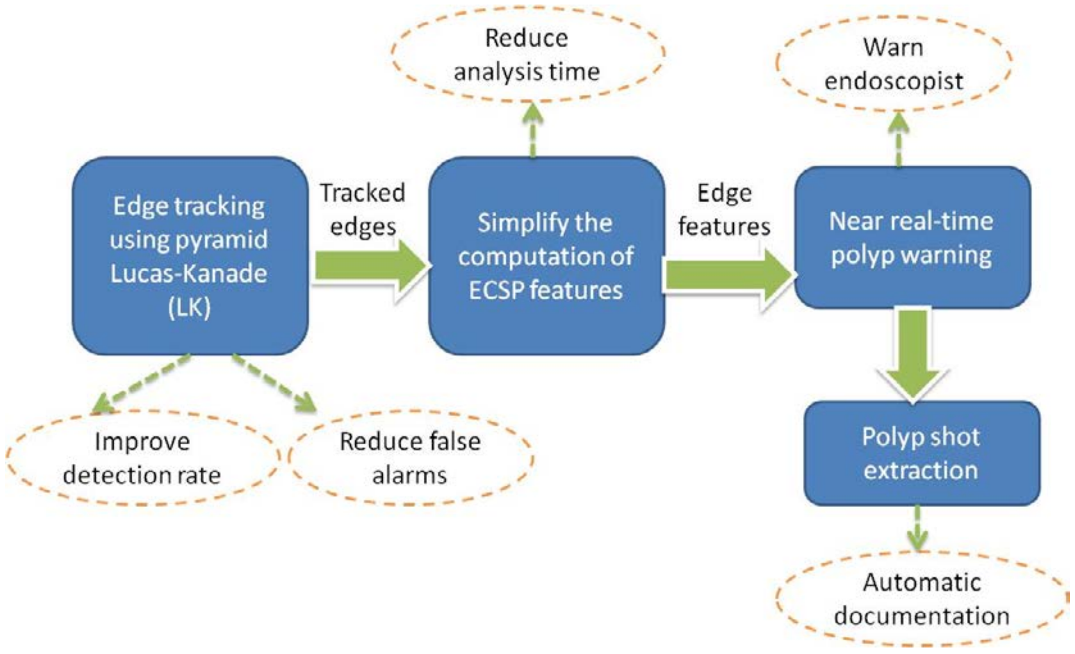


Figure 6.9: The flow chart of components of the near real-time polyp warning and polyp shot extraction system.

6.6.1 Simplified Features for Near Real-time Polyp Detection

Our features for polyp detection are mainly obtained from Part-based Multi-derivative ECSP. To speed up the computation time, especially the bottlenecks of the feature extraction part, we modified our feature extraction method as follows.

Texture feature on $ROI_{concave}$: We experimentally found that the filtering step using LOWESS filter dramatically slowed down the computation speed. Therefore, we removed the LOWESS filter. Instead, we compute the following texture feature without using any smoothing filter in our near real-time version. We compute one binary matrix for each $ROI_{concave}$ as detailed below. The width and height of the binary matrix are the same as those of its $ROI_{concave}$. Finally, we compute the texture feature $V_{texture}$ as the average of the binary values of the binary matrix of $ROI_{concave}$ in the domain of [0, 1].

1) If the width of $ROI_{concave}$ is larger than *17*, we slide a *1x17* pixel window over $ROI_{concave}$ one pixel at a time. We assign a binary score of *1* for the corresponding pixel in the binary matrix if the pixel intensity value in $ROI_{concave}$ in the right most position of the window is greater than that in the left most position of the window, or a binary score of *0* otherwise. We experimented with different window sizes and finally chose a *1x17* pixel window since it can better resist noise.
2) If the width of $ROI_{concave}$ is smaller than *17,* we slide a *1x2* pixel window over $ROI_{concave}$ one pixel at a time. We assign a binary score of *1* for the corresponding pixel in the binary matrix if the pixel intensity value in the right side of the window is greater than that in the left side of the window, or a binary score of *0* otherwise. A *1x2* pixel window is chosen due to the computation simplicity.

In most case, the tendency of intensity increase along the protruding direction of polyps on $ROI_{concave}$ is not obvious within a small size window without any smoothing due to the noise. Therefore, we need a sufficiently large window to compute the tendency of intensity increase. However, if $ROI_{concave}$ has a small width, it is most likely that the size of its corresponding edge is small (i.e., short edge). For a short polyp edge, it is most likely that its corresponding $ROI_{concave}$ is not very noisy. A short non-polyp edge is most likely caused by light reflection, which generally has a smooth surface on $ROI_{concave}$. Therefore, a *1x2* pixel window is sufficient for the computation without smoothing. Note that we calculate the texture feature on the entire $ROI_{concave}$ instead of calculating separate features on $ROI^3_{concave}$ and $ROI^4_{concave}$. This is because we use a larger window for the texture feature calculation in the near real-time version.

Shape of edge: We added one shape feature to remove edges that do not fit well with an ideal ellipse using the least squares ellipse fitting method [82]. We perform least squares ellipse fitting on all extracted edges in the binary edge image and dilate the fitted ideal ellipse for each edge using a circle structuring element with a radius of *4* pixels. The dilation is to account for some nearby edge pixels that do not lie perfectly on the ideal ellipse. Let *N* be the total number of pixels on the edge. Let $E_{ellipse}$ be the percentage of edge pixels not covered by the dilated ideal ellipse. We reject edges in which $E_{ellipse}$ is large than a threshold (Table 6.7).

We selected all useful features using the forward feature selection method [80] based on the training data. We set our parameter values based on *leave-one-out-cross-validation* in the training data. That is, for *N* polyp clips in the training set, we set parameter values based on training *N-1* polyp video clips and validate the performance on the remaining one polyp video clip. We repeated this step to select one polyp video clip from training set and record the performance at each validation step. We used the parameter values that gave the maximum average performance. To save the computation time, we no longer use some useful features presented in Section 6.4 like V_{shape} and $V_{smoothness}$. Instead, we replace them with some features calculated using the appendiceal orifice detection algorithm (Chapter 4). We summarized all used features and their value ranges for edges qualified as polyp edges in Table 6.7. The notations and descriptions of these features can be found in Section 6.4 and Chapter 4.

Table 6.7: Features, Their Order of Computation and Value Ranges for Near Real-Time Polyp Detection

Computation Order	Feature Name	Feature Description	Value Range
1	*EdgeLinearity**	MSE (mean squared error) of the edge fitted using least squares linear fitting	≥ 2 pixels
2	$E^*_{ellipse}$	Percentage of edge pixels not covered by the fitted dilated ideal ellipse	[0, 0.8]
3	*Curvature**	Curvature range of the edge	[1/400, 1/20]
4	h_2	Amount of intensity return after crossing the edge	≥ 0.02
5	h_2 / h_1	Ratio of amount of intensity return after crossing the edge to the amount of intensity drop before crossing the edge	≥ 0.5
6	h_3	Sharpness of edge before crossing the edge	≤ 1.67
7	h_4	Sharpness of edge after crossing the edge	≥ 0.652
8	$h_4 - h_3$	Difference between the sharpness of edge after crossing the edge and before crossing the edge	≥ 0
9	h_4 / h_3	Ratio of edge sharpness after crossing the edge to the edge sharpness before crossing the edge	≥ 1.0
10	$V_{texture}$	Texture feature on $ROI_{concave}$	(0.7, 1]

All parameters in this table were obtained based on experiments with the training data set. They were then used for running the experiments on the testing data set.
* Parameters sensitive to an input image size

6.6.2 A Coarse-to-Fine Near Real-Time Polyp Image Detection System

An edge is detected as a polyp edge if all features in Table 6.7 are within their value ranges. Otherwise, the edge is detected as a non-polyp edge. If any edge is detected as a polyp edge in an image, we consider that image as a polyp image.

Because the extraction of features is the most time consuming part in the system, to save the computation time, we design our polyp edge detection system as a coarse-to-fine detection system. That is, the system calculates one feature after another, starting from the calculation of the feature with the fastest computation time. We list the order of computations of these features in Table 6.7. If the value of any feature is not in its value range as specified in Table 6.7, the edge is detected as a non-polyp edge without the calculation of subsequent features. Comparing with the methods using machine learning models like SVM presented in Section 6.5, a coarse-to-fine detection system saves the overall computation time and benefits the analysis time of frames with many detected edges. We consider that a polyp image is correctly detected if the system detects at least one polyp edge in ground truth of that image.

6.6.3 Edge Tracking

Toward reducing the number of falsely detected edges and improving the overall detection rate using temporal information, we apply Lucas-Kanade tracking [83] method to track detected polyp edges across multiple frames. The intuition is that if a detected edge is indeed a real polyp edge, there is a high probability that the edge will be detected as a polyp edge in subsequent frames in a small time window. Otherwise, the edge is considered as a falsely detected edge.

There are many kinds of local features that can be used for tracking. Most popular trackable local features are the pixels with large derivative values in two orthogonal directions in the image [84]. These pixels are considered as interest points, such as *corners* [85]. Generally, we do not consider pixels on edges to be good for tracking. This is because many edges of objects have a local linear shape. The pixels on edges with local linear shape have only large derivative values in one direction, but small derivative values in another orthogonal direction. However, if pixels on edges have large derivative values in both of the two orthogonal directions, these edge pixels can be considered as good points to track, such as pixels on curve edges. Polyp edges are curve edges in most cases. We consider a subset of pixels on polyp edges as good feature points to track.

We use a popular tracking method – Lucas-Kanade (LK) to track edges. LK tracking is a *sparse optical flow* tracking method which has a reliable performance on tracking when given points with good features to track [84] [86], such as corners as discussed earlier. LK tracking is fast. In the following paragraphs, we give a brief review of the LK tracking technique and discuss how it is used for tracking polyp edges.

Brief review of LK tracking: Let $I(x, y, t)$ be the intensity value of a pixel at the coordinate $\vec{u} = (x, y)$ at frame *t*. Assume that the intensity value of pixel *I* does not change from frame *t* to *t+1*. We have

$$I(x + u, y + v, t + 1) = I(x, y, t), \tag{6.4}$$

where, $\vec{q} = (u, v)$ is the displacement of the pixel between frames *t* and *t+1*. Based on Taylor series expansion, for a small displacement (u, v), the following equation is satisfied:

$$I_x u + I_y v + I_t = 0, \tag{6.5}$$

where I_x and I_y are the partial derivatives of intensity values along the *x* and *y* directions, and I_t is the partial derivative of the intensity value along the temporal domain *t*. To solve for the displacement vector (u, v), we fit a small 2 dimensional window W_t centered at $\vec{u} = (x, y)$. Let us define $\vec{u}_i$ as the pixel in window W_t for *i = 1, …, N*. Equation (6.5) for all *N* pixels (i.e., $\vec{u}_1 \dots \vec{u}_N$) in the window W_t can be expressed in a matrix form as

$$\begin{bmatrix} I_x(\vec{u}_1) & I_y(\vec{u}_1) \\ \dots & \dots \\ I_x(\vec{u}_N) & I_x(\vec{u}_N) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(\vec{u}_1) \\ \dots \\ I_t(\vec{u}_N) \end{bmatrix}. \tag{6.6}$$

Let the matrix $A = \begin{bmatrix} I_x(\vec{u}_1) & I_y(\vec{u}_1) \\ \dots & \dots \\ I_x(\vec{u}_N) & I_x(\vec{u}_N) \end{bmatrix}$ and the matrix $B = \begin{bmatrix} I_t(\vec{u}_1) \\ \dots \\ I_t(\vec{u}_N) \end{bmatrix}$. The solution of this linear system is casted into solving the problem of minimizing the MSE error of least-squares fitting in the window. That is

$$\begin{bmatrix} u \\ v \end{bmatrix} = (A^T A)^{-1} A^T B. \tag{6.7}$$

It is obvious that a corner feature like pixel generally has two large eigenvectors of $A^T A$, because it has large derivative values in two orthogonal directions. Therefore, $A^T A$ is generally invertible for edge pixels on curve edges. To solve the problem of a large displacement of vector (u, v) that cannot fit in a small window, we use the Pyramid Lucas-Kanade [84]. The Pyramid LK calculates the tracked feature points from the top layer of the first Gaussian Pyramid. The calculated feature points are used at the next Gaussian Pyramid layer for motion estimation. Generally, Pyramid LK with a larger

Gaussian Pyramid PL has a better tracking accuracy but a higher computational time. A larger Gaussian Pyramid PL is better for tracking an edge with a large motion because it searches the corresponding tracked point in a finer Pyramid image. We experimentally selected $PL = 5$ for polyp edge tracking.

Tracking edge pixels using Pyramid LK: We track pixels on edges between two consecutive frames as follows. First, we downsample the pixels on an edge using the sampling factor of L_e (*14* pixels), i.e., selecting one pixel out of every L_e pixels on that edge. We consider the downsampled edge pixels as feature points for Pyramid LK tracking. Next, we use the OpenCV function cvCalcOpticalFlowPyrLK [84] to track the selected feature points. The cvCalcOpticalFlowPyrLK function calculates the error ε_t for each feature point as the difference of appearance between the feature point and its tracked point. The tracked point is searched within a small window W_{ε_t} surrounding the feature point. We consider that an edge pixel is correctly tracked if both of the following conditions are satisfied: 1) the tracking error $\varepsilon_t < Thld_{\varepsilon_t}$; 2) the distance $dist_t < Thld_{dist_t}$, where $dist_t$ is the Euclidean distance between the feature point in frame *t* and its tracked point in frame *t+1*.

Tracking/Matching edges between two frames: Given two consecutive frames *t* and *t+1*. We consider the edge *j* in frame *t+1* matches the edge *i* in the previous frame *t* if there is at least one pixel on edge *j* falls inside the 2D window of size R_w centered at any tracked feature point on edge *i*. If there is more than one such edge in previous frame *t*, we match edge *j* to the edge *i* which has the largest number of matched feature points.

6.6.4 Near Real-time Polyp Warning Feedback

We track each edge obtained from Canny edge detector. We start marking the tracked edge on screen as visual feedback to warn the endoscopist if that edge is detected as polyp edge for at least N_p consecutive times (i.e., in N_p consecutive clear frames). A warned polyp edge is marked until the tracking of that edge is lost. We consider a sequence of consecutively detected non-blurry frames as a detected polyp sub-shot if it contains the same tracked polyp edges in each frame. The start and end frames with consecutive tracked polyp edges are the start and end frames of a detected polyp sub-shot.

Unexpected loss of tracking: In an ideal case, the loss of a tracked edge is expected if that edge no long appears in subsequent frames in the rest of the video. However, the loss of a tracked edge may also happen if 1) the edge is not detected in consecutive images due to the noisy nature of colonoscopic images such as image blur, or 2) the edge disappears out of the FoV of the endoscope and then reappears in the FoV after a short time. To handle the unexpected loss of tracking, we set the value of N_p to either one of the constant values $Thld_{p1}$ or $Thld_{p2}$ where $Thld_{p2} > Thld_{p1}$ as follows.

1) If a tracked polyp edge i is lost at frame t, we set N_p to $Thld_{p1}$. During the next T_w seconds from frame t, if an edge is detected as a polyp edge for at least N_p consecutive times on different frames, that edge is considered as a warned polyp edge and the visual feedback on that edge is shown. The reason to use the lower of the two threshold values is to allow for a faster response time of the feedback on a polyp that reappears after it is briefly out of the FoV of the camera.
2) If a tracked polyp edge i is lost at frame t, and there exists any other edge which is still tracked at frame t, we also set N_p to $Thld_{p1}$ for all subsequent frames until all tracked edges from frame t are lost or an edge is detected as a polyp edge for Np consecutive times on different frames. That is, we use the presence of the common colon wall (sharing some common edges) to avoid ending the polyp sub-shot too early because the polyp is not too far to reappear in the FoV of the camera.
3) Otherwise, we set N_p to $Thld_{p2}$.

We describe the major functions in our algorithms in Figure 6.10 and 6.11 using pseudo code.

Algorithm: Polyp Edge Detection

```
// Functionality: Identify an edge as a polyp edge if all of its feature scores are within their value
//                        ranges
// Input:
//         edge[]: Array with each element storing the information of an edge
//         id: Edge ID
// Output: True if the edge corresponding to edge id is a detected polyp edge
IsPolypEdge (edge[], id)
1:   // MSE: Linear line fitting representing the edge linearity
2:   MSE ← MSEofEdgeLinearity(edge[id])
3:   if MSE < 2 pixels then
4:       return false
5:   end if
6:   // fitEllipse: Ellipse fitting score
7:   fitEllipse ← EllipseFittingScore(edge[id])
8:   if fitEllipse>0.8 then
9:       return false
10:  end if
11:  // Curvature: Curvature of an edge
12:  Curvature ← EdgeCurvature(edge[id])
13:  if Curvature < 1/400 or Curvature > 1/20 then
14:      return false
15:  end if
16:  // Calculate ECSP profiles and features for these profiles
17:  edge[id].profiles ← ECSPProfiles(edge[id])
18:  edge[id].$h_1$, edge[id].$h_2$, edge[id].$h_3$, edge[id].$h_4$ ← KeyPointFeatures(edge[id])
19:  if edge[id].$h_2$ / edge[id].$h_1$ < 0.02
20:      or edge[id].$h_3$ > 1.67
21:      or edge[id].$h_4$ < 0.652
22:      or edge[id].$h_4$ - edge[id].$h_3$ < 0
23:      or edge[id].$h_4$/edge[id].$h_3$ edge <1 then
24:          return false
25:  end if
26:  if TextureFeature(edge[id].profiles, id) ≤ 0.7 then
27:      return false
28:  end if
29:  return true
```

Figure 6.10: Pseudo code of the polyp edge detection algorithm

Algorithm: Near Real-Time Polyp Sub-shot Warning

```
// Functionality: Start to warn the endoscopist the detected polyp edge if we detect the edge as a
//                polyp edge in $N_p$ clear images.
// Input:
//      video: Input video
//      $Thld_{p1}$: The minimum times of detected edges to start to warn the endoscopist
//      $Thld_{p2}$: Threshold for handling the loss of tracking
//      $T_w$: Threshold for grouping two sub-shots into one sub-shot
//      frameRate: Frame rate of the input video
// Output: Warning edge(s) labeled on the screen
POLYPSUBSHOT (video, $Thld_{p1}$, $Thld_{p2}$, $T_w$, frameRate)
1:   // videoClip.clearFrame: Array storing information of all clear frames
2:   // fId: Frame ID
3:   fId ← 0
4:   //isPolypDetected: True if we detected at least a polyp frame in this video, false otherwise
5:   isPolypDetected ← false
6:   while ISEXIST(video.clearFrame[fId]) do
7:       // Calculate canny edges of current frame CurrFrame.edge
8:       CurrFrame.edge ← CANNYEDGES(video. clearFrame[fId])
9:       // If CurrFrame is the first frame with detected edges, set CurrFrame as the initial
10:      // frame for tracking
11:      if ISTHEFIRSTFRAME(video.clearFrame[fId]) == true then
12:          PrevFrame ← CurrFrame
13:          fId ← fId+1
14:          continue
15:      end if
16:      // trackedEdge[]: Array storing the tracked edges from previous frame and the
17:      //                detected new edges from current frame
18:      // nTrackedEdges: Number of combined edges
19:      trackedEdge[], nTrackedEdges ← TRACKEDGES (PrevFrame.edge, CurrFrame.edge )
20:      // Set $N_p$ value to handle the loss of tracking
21:      $N_p \leftarrow Thld_{p1}$
22:      // LatestPolypFrameID: Frame ID of the latest detected polyp image
23:      if isPolypDetected == true
24:          and CurrFrame.id – LatestPolypFrameID < frameRate * $T_w$
25:              $N_p \leftarrow Thld_{p2}$
27:      // Check whether there is at least one tracked edge between two frames
28:      else
29:          if nTrackedEdges > 0
30:              $N_p \leftarrow Thld_{p2}$
```

Algorithm: Near Real-Time Polyp Sub-shot Warning (continued)

```
31:             end if
32:         end if
33:         // Warn an edge to endoscopist if it is a detected polyp edge in $N_p$ clear images
34:         for eId ← 0 to nTrackedEdges
35:             if trackedEdge[eId].detectedPolypTimes < $N_p$
36:                 if IsPolypEdge(trackedEdge[eId]) == true
37:                     trackedEdge[eId].detectedPolypTimes
38:                     ← trackedEdge[eId].detectedPolypTimes +1
39:                     isPolypDetected ← true
40:                 end if
41:             else
42:                 // Display detected polyp edge(s) on a screen to warn the endoscopist
43:                 DisplayWarningOnScreen(trackedEdge[eId])
44:                 LatestPolypFrameID ← CurrFrame.id
45:             end if
46:         end for
47:         CurrFrame ← PrevFrame
48:         fId ← fId +1
49:     end while
```

Figure 6.11: Pseudo code of the near real-time polyp sub-shot warning algorithm

6.6.5 Near Real-time Polyp Shot Extraction

We group two consecutive detected polyp sub-shots and all frames which do not contain any warned polyp edge between the two detected polyp sub-shots into a single semantic unit – a detected polyp shot if any one of the following conditions is satisfied.

1) The duration between the ending frame of a detected sub-shot and the starting frame of its consecutively detected sub-shot are smaller than T_s seconds.
2) Two consecutively detected sub-shots have a commonly tracked edge.

We extract all these groups as detected polyp shots. We discard a detected polyp shot if it is shorter than T_p seconds.

6.7 Performance of Near Real-time Polyp Detection

We implemented all the analysis code of the near real-time polyp detection and edge tracking in C/C++ (Win32 in Microsoft Visual Studio 2008) with Open Source Computer Vision library (OpenCV Version 2.1).

6.7.1 Experimental Environment and Data

Data Set: We randomly selected *61* de-identified test video files captured during routine screening colonoscopy at the image resolution of *720x480* pixels at *29.97* frames per second (fps). Each file contains only one colonoscopy procedure in its entirety. Among these *61* video files, *8* video files were included in the training set, and the remaining *53* files were included in the testing set. Among the eight training videos, three of them were captured using an OLYMPUS endoscope and the remaining five videos were captured using a FUJINON endoscope. Each of these training videos contains at least one polyp. Next, we selected one polyp shot from each training video by manually labeling the starting and ending frames of each shot. We will discuss the way to label a polyp shot in a later paragraph. Finally, we obtained *8* polyp shots from *8* different videos as our training data. Among *53* test videos, forty-one of these videos were captured using OLYMPUS endoscopes, and the remaining twelve video files were captured using FUJINON endoscopes. Twenty two videos of these *53* test videos contain at least one polyp, and the remaining thirty one videos do not contain any polyp. The colons seen in these videos had little stool inside them. We ran our real-time version of blurry frame removal software [70], leaving the remaining clear (in-focus) images for polyp detection and edge tracking. The blurry frame removal algorithm in [70] considers a frame as blurry if it does not have sufficient number of connected edge pixels detected using Canny edge detector. The overall system consists of three consecutive components (modules). Table 6.9 shows the functional output, the frame analysis rate in fps of each module and the dependencies among the modules. We run these modules in parallel using our middleware software – SAPPHIRE [87]. Although these modules have the output dependence, the bottleneck is at the slowest module. The average analysis time is close to the analysis time of the slowest module in an ideal case [87].

Ground truth labeling: As aforementioned, the tracking of a polyp edge may be lost due to the noisy nature of colonoscopic images such as image blur. As a result, the consecutive frames showing a polyp could be easily broken into many polyp sub-shots in the detected results. We manually label the starting and ending frames of a polyp sub-shot and a polyp shot in the ground truth data by the following ways. We consider frames i and j as starting and ending frames of a polyp sub-shot respectively if that polyp continuously appears in images from the frame i to the frame j, and disappears at the frame $j+1$. In the ground truth, we consider consecutive polyp sub-shots and all non-polyp images between each two of them together as one polyp shot if these polyp sub-shots contain the same polyp. Therefore, a polyp shot in our ground truth can be either one sub-shot or consist of several nearby consecutive sub-shots. In the latter case, we label the starting frame of the first polyp sub-shot and the ending frame of the last polyp sub-shot in a polyp shot as the starting and ending frames of that shot. In our data set, a polyp removal operation typically appears after that polyp is found. During the removal of that polyp, snares or biopsy forceps (see Figure 5.1 (b-c) in Chapter 5) appear together with the polyp. These instruments sometimes occlude the polyp. In addition, after the polyp removal, only a partial polyp may remain and its shape may be changed. This is because the polyp is too big or is in a difficult location for the endoscopist to remove the entire polyp at once. Therefore, we do not consider images with polyps together with an instrument or frames after a polyp removal as part of our ground truth data.

Image pre-processing: We extract edges detected by Canny edge detector. Before we apply Canny edge detector, we smooth an image by two Gaussian filters separately with different standard deviations σ_1 and σ_2. We apply Gaussian filters twice with different standard deviations to avoid noisy edges caused by details in image at different scales. All Canny edges are used for polyp edge detection and edge tracking. To speed up the analysis time, we no longer cut an S-like shape edge into several C-like edges because Canny edges obtained using cvCanny function in OpenCV are sometimes broken into small parts.

Parameters: All parameter values were determined based on experiments with the training set of 8 polyp shots from the 8 training videos. The parameter values may vary for different image resolutions in other data sets. Table 6.10 lists the parameters values.

Table 6.8: Description of Videos in the Testing Set for Near Real-Time Polyp Detection

Category	OLYMPUS	FUJINON	Total
Number of videos	41	12	53
Number of polyps	20	23	43
Average video length (hh:mm:ss)	00:18:05	00:28:53	00:20:21
Total video length (hh:mm:ss)	12:11:58	05:46:41	17:58:39

Table 6.9: Three Modules of Near Real-Time Polyp Detection System

Module Name	Functional Output	Frame Analysis Rate	Dependency Module Name
Clear image detection	Clear images	30 fps	-
Edge tracking	Tracked edges	30 fps	Clear image detection
Polyp warning and shot extraction	Warned polyp sub-shots and extracted polyp shots	10 fps	Edge tracking

Table 6.10: Parameters and Values in OpenCV for the Proposed Near Real-Time Polyp Detection

No.	Parameter Name	Valid Parameter Value/Range
1	Gaussian filters	Standard deviations of Gaussian filters: $\sigma_1 = 9$ and $\sigma_2 = 3$
2	Canny edge detector	Low threshold of 10; high threshold of 15
3	D_i	Sliding widow size for the calculation of edge normal: 30x30 pixels
4	λ	The parameter in Equations 3.1 and 3.2: λ = 0.25
5	W_t*	Size of the window for LK tracking: 15x15 pixels
6	$Thld_{\varepsilon_t}$*	Threshold of error ε_t of tracked edge pixels; $Thld_{\varepsilon_t} = 500$
7	$Thld_{dist_t}$*	Threshold of the Euclidean distance between an edge pixel in frame *t* and its tracked pixel in frame *t+1*: $Thld_{dist_t} = 40$ pixels
8	R_w	Size of a window centered at the tracked edge pixel: 3x3 pixels
9	L_e	Downsampling factor of edge pixels on an edge: $L_e = 14$
10	$Thld_{p1}$	Minimum number of consecutive times on different frames an edge is detected as a polyp edge in order to consider this edge as a warned polyp edge after losing edge tracking: $Thld_{p1} = 2$
11	$Thld_{p2}$	Minimum number of consecutive times on different frames a tracked edge is detected as a polyp edge for this edge to be considered as a warned polyp edge: $Thld_{p2} = 3$
12	T_s	Minimum duration between two consecutive polyp sub-shots: $T_s = 15$ seconds
13	T_p	Minimum duration of a polyp shot: $T_p = 10$ seconds
14	PL	Gaussian Pyramid level of Pyramid LK tracking: PL = 5

All parameters in this table were obtained based on experiments with the training data set. They were then used for running the experiments on the testing data set.

* Parameters sensitive to an input image size

Performance metrics: We mark the tracked polyp edges as visual feedback – polyp warning feedback on screen to warn the endoscopist and extract polyp shots as defined in Section 6.6.5. We define the following criteria as the measurement of correctness for the detection.

1) A polyp is correctly warned if the warned polyp sub-shot contains at least one image showing that polyp.
2) A polyp is correctly detected in the polyp shot if at least one image showing that polyp is in the detected shot.
3) A detected polyp shot is a correctly detected polyp shot if it contains at least one real polyp image.
4) A detected polyp sub-shot is a correctly detected polyp sub-shot if it is a sub-shot in a polyp shot in the ground truth.

We define the following performance metrics for polyp warning and polyp shot detection using the notations in Table 6.11.

Performance metrics of polyp warning:

$$\mathrm{recall_{WRN}} = \frac{\mathrm{COR\_WRN\_PLP}}{\mathrm{NUM\_RL\_PLP}}$$

$$\mathrm{numFL_{WRN}} = \frac{\mathrm{numFL_{WRN}}}{\mathrm{NUM\_VIDEO}}$$

$$\mathrm{avgLenFL_{WRN}} = \frac{\mathrm{lenFL_{WRN}}}{\mathrm{numFL_{WRN}}}$$

$$\mathrm{ratioLenFL_{WRN}} = \frac{\mathrm{avgLenFL_{WRN}}}{\mathrm{avgLenVideo}}$$

Performance metrics of polyp shot extraction:

$$\mathrm{recall_{SOT}} = \frac{\mathrm{COR\_PLP\_SOT}}{\mathrm{NUM\_RL\_PLP}}$$

$$\mathrm{numFL_{SOT}} = \frac{\mathrm{numFL_{SOT}}}{\mathrm{NUM\_VIDEO}}$$

$$\mathrm{avgLenFL_{SOT}} = \frac{\mathrm{lenFL_{SOT}}}{\mathrm{numFL_{SOT}}}$$

$$\mathrm{ratioLenFL_{SOT}} = \frac{\mathrm{avgLenFL_{SOT}}}{\mathrm{avgLenVideo}}$$

Table 6.11: Notation and Explanation of Performance Metrics for Near Real-Time Polyp Detection

No.	Notation	Explanation of Performance Metrics
1	NUM_RL_PLP	Number of real polyps
2	COR_WRN_PLP	Number of correctly warned polyp
3	COR_PLP_SOT	Number of correctly detected polyp shots
4	NUM_VIDEO	Number of analyzed videos
5	$\mathrm{numFL}_{\mathrm{WRN}}$	Total number of falsely warned polyp sub-shots in all tested videos
6	$\mathrm{numFL}_{\mathrm{SOT}}$	Total number of falsely detected polyp shots in all tested videos
7	$\mathrm{avgNumFL}_{\mathrm{WRN}}$	Average number of falsely warned polyp sub-shots per video
8	$\mathrm{avgNumFL}_{\mathrm{SOT}}$	Average number of falsely detected polyp shots per video
9	$\mathrm{lenFL}_{\mathrm{WRN}}$	Total duration in seconds of all falsely warned polyp sub-shots per video
10	$\mathrm{lenFL}_{\mathrm{SOT}}$	Total duration in seconds of all falsely detected polyp shots per video
11	$\mathrm{avgLenFL}_{\mathrm{WRN}}$	Average duration in seconds of each falsely warned polyp sub-shot
12	$\mathrm{avgLenFL}_{\mathrm{SOT}}$	Average duration in seconds of each falsely detected polyp shot
13	$\mathrm{Recall}_{\mathrm{WRN}}$	Percentage of the number of correctly warned polyps out of the total number of polyps
14	$\mathrm{Recall}_{\mathrm{SOT}}$	Percentage of the number of correctly detected polyps in polyp shots out of the total number of polyps
15	$\mathrm{ratioLenFL}_{\mathrm{WRN}}$	Ratio of the average duration in seconds of each falsely warned polyp sub-shot and the average duration in seconds of each video
16	$\mathrm{ratioLenRecall}_{\mathrm{SOT}}$	Ratio of the average duration in seconds of correctly detected polyp shots and the average duration in seconds of polyp shots per video
17	avgLenVideo	Average video length in seconds
18	$\mathrm{avgLenPolyp}_{\mathrm{SOT}}$	Average duration in seconds of polyp shots per video

6.7.2 *Performance on Near Real-time Polyp Detection*

Table 6.12 shows the performance of our system for polyp warning and polyp shot extraction in terms of detection rate. We ran the system on the testing set of 53 videos.

Polyp Warning Performance: Our system correctly warned *97.7%* (*42* of *43*) of polyps. The total duration of false sub-shots on average is *4.3%* of a video duration. That is, our system can correctly warn *97.7%* polyps in near real-time and does not generate falsely detected images for the *95.7%* duration in an entire colonoscopy procedure. The average number of false sub-shots per video is *36.2* and the average duration of each false sub-shot is only *1.4* seconds. The only mis-detected polyp is due to the image blur. The clear image detection module detected all frames of this polyp as blur (out of focus) frames. Therefore, our polyp detection module did not analyze these frames.

Polyp Shot Extraction Performance: We extracted a polyp shot after T_p seconds from the detected end frame in the last sub-shot of that polyp shot. As aforementioned, we ignore a detected polyp shot if it is shorter than T_p seconds. As a result, we correctly extracted *90.7%* (*39* of *43*) of polyps. Each of these extracted polyp shots covers an average of *78.2%* of polyp shot frames in the ground truth. We extracted an average of *4.0* false shots per video with an average duration of *23.1* seconds. The duration of these false shots is *8.2%* of a video duration. We mis-detected *3* additional polyps. Two of the three mis-detected polyps have a tiny size on images and one of them does not have enough illumination. The algorithm does not detect longer than T_p seconds of the appearance of these polyps in their videos, although these polyps are correctly warned in near real-time.

The missed polyp images are of 1) a polyp that appears as a tiny region in an image and 2) a polyp with weak illumination. Polyps from these two categories were seen at a long distance from the capturing camera. It is hard to obtain good Canny edges from the images of these two types of polyps. The falsely detected images are from the following categories: 1) protruding colon folds with smooth surface, 2) protruding appendiceal orifice, 3) ileocecal valve and 4) smooth water. Most of falsely detection images are from ileocecal valve. Some of the falsely detection images are from protruding colon folds and appendiceal orifices. Only a few of them are from smooth water.

Table 6.12: Performance of Near Real-Time Polyp Warning and Polyp Shot Extraction

Endoscope Type	Polyp Warning		Polyp Shot Extraction	
	Performance Metrics	Value	Performance Metrics	Value
OLYMPUS	$\mathrm{recall}_{\mathrm{WRN}}$	95%	$\mathrm{recall}_{\mathrm{SOT}}$	85%
	$\mathrm{numFL}_{\mathrm{WRN}}$	32.8	$\mathrm{numFL}_{\mathrm{SOT}}$	3.7
	$\mathrm{avgLenFL}_{\mathrm{WRN}}$	1.3 s.	$\mathrm{avgLenFL}_{\mathrm{SOT}}$	23.9 s.
	$\mathrm{ratioLenFL}_{\mathrm{WRN}}$	4.3%	$\mathrm{ratioLenFL}_{\mathrm{SOT}}$	9.0%
			$\mathrm{ratioLenRecall}_{\mathrm{SOT}}$	81.1%
FUJINON	$\mathrm{recall}_{\mathrm{WRN}}$	100%	$\mathrm{recall}_{\mathrm{SOT}}$	100%
	$\mathrm{numFL}_{\mathrm{WRN}}$	48.0	$\mathrm{numFL}_{\mathrm{SOT}}$	4.8
	$\mathrm{avgLenFL}_{\mathrm{WRN}}$	1.5 s.	$\mathrm{avgLenFL}_{\mathrm{SOT}}$	24.7
	$\mathrm{ratioLenFL}_{\mathrm{WRN}}$	4.4%	$\mathrm{ratioLenFL}_{\mathrm{SOT}}$	7.3%
			$\mathrm{ratioLenRecall}_{\mathrm{SOT}}$	73.6%
Total	$\mathrm{recall}_{\mathrm{WRN}}$	97.7%	$\mathrm{recall}_{\mathrm{SOT}}$	90.7%
	$\mathrm{numFL}_{\mathrm{WRN}}$	36.2	$\mathrm{numFL}_{\mathrm{SOT}}$	4.0
	$\mathrm{avgLenFL}_{\mathrm{WRN}}$	1.4 s.	$\mathrm{avgLenFL}_{\mathrm{SOT}}$	23.1 s.
	$\mathrm{ratioLenFL}_{\mathrm{WRN}}$	4.3%	$\mathrm{ratioLenFL}_{\mathrm{SOT}}$	8.2 %
			$\mathrm{ratioLenRecall}_{\mathrm{SOT}}$	78.2%

Analysis time: We ran our analysis software on an Intel (R) Xeon 2.0 GHz duo-core CPU with 6.0 GB RAM on a 64-bit machine with Windows 7 Professional operating system. Table 6.13 shows the analysis time of our system to analyze an image for edge tracking and polyp detection modules. We used the clock function [88] in C++/C to record the starting time and ending time of each module running on each frame. We measured the analysis time of two modules running on 1) only clear images, and 2) both clear and blurry images. If we run edge tracking or polyp detection module along, each of them took an average of less than *30* ms to analyze a clear image, or an average of around *20* ms to analyze an image in an entire video. However, Table 6.13 does not truly reflect the analysis time of our system. This is because we ran our modules in parallel using the middleware software – SAPPHIRE [87]. We set different frame analysis rates for different modules – *30* fps for the clear image detection module, *30* fps for the edge tracking module, and *10* fps for the polyp detection module. As a result, we can run the system in near real-time to analyze a colonoscopy video.

Table 6.13: Average Analysis Time for Edge Tracking and Polyp Detection (Milliseconds (ms) per image)

Functionality	Clear images		All images			
	mean	median	mean	median	minimum	maximum
Edge tracking	27 ms	16 ms	19 ms	16 ms	0 ms	203 ms
Polyp detection	28 ms	16 ms	20 ms	16 ms	0 ms	188 ms

6.8 Chapter Summary

In this chapter, we introduced a part-based ECSP for polyp detection. We presented an algorithm to segment multi-derivative ECSP and ROI into non-overlapping parts. This part-based ECSP gives us flexibility to obtain different kinds of features from different parts or pick up a particular method to model a part that is suitable to that part. We investigated new features obtained from parts, and compared it with two existing texture and color based state-of-the-art techniques for polyp image detection. The simplified features obtained from part-based ECSP module further saved the computational time. By combining the detected polyp edges from our polyp image detection method and an existing edge tracking method, we correctly detected *97.7%* (*42* of *43*) of polyps obtained from *53* video files that were randomly selected from routine screening colonoscopy. Our polyp warning and shot extraction system can run in near real-time with an average of *36.2* falsely detected polyp sub-shots per video, and the average duration of each of these false sub-shots is less than *1.5* seconds. To our best knowledge, our system is the first technique that can effectively detect polyp images and polyp shots in near real-time. Our system is very promising for clinical use to assist the endoscopist by providing visual feedback of a potential polyp during routine screening colonoscopy. The future work of this research is to further reduce the computational time by optimizing both the algorithm and analysis code to achieve a *30* fps analysis rate for each module, and use our system in a clinical trial to determine additional needs in order to deploy the system in endoscopy practice.

CHAPTER 7 CONCLUSION AND FUTURE WORK

This dissertation describes several contributions in both ECSP based techniques and colonoscopic object detection as follows.

1) It demonstrates a new calculation method to extract features from multi-derivative ECSP (Chapter 3) and presents the first feature-based ECSP techniques for object detection.
2) It presents an edgeless ECSP method that calculates ECSP and obtains ECSP features without using edge detectors (Chapter 5).
3) It presents a part-based ECSP method that allows us to model and extract features from each segmented part using the method which is suitable for that part (Chapter 6).
4) It proposes several colonoscopic object detection algorithms including appendix image/video detection algorithms (Chapter 4), near real-time retroflexion image/video algorithms (Chapter 5), and near real-time polyp image detection and polyp shot extraction algorithms (Chapter 6).
5) It demonstrates a software package for evaluating our techniques in a clinical trial. The software is very promising for clinical use in terms of both detection rates and analysis time.

7.1 Detection of Appendix Image and Video

In Chapter 4, we introduced two algorithms for detecting the image showing the clearly seen appendiceal orifice and the video showing at least three seconds of the appendiceal orifice inspection based on ECSP features. The proposed appendix video detection algorithm first uses new local features derived from ECSP and its 1^{st} – order derivative profile to detect the appearance of the appendiceal orifice and then uses near pause detection to recall miss detected appendiceal orifice images with weak edges and reject some false classifications.

The experimental results show that our method is able to handle object rotation, translation and scaling in image, and noisy images with strong light reflection. The developed system can effectively detect appearance of appendiceal orifice in image and appendix video tested on *23* endoscopy procedures. The average sensitivity and specificity for the detection of appendiceal orifice images with the often seen crescent appendiceal orifice shape are *96.86%* and *90.47%*, respectively. The average accuracy for the detection of appendix videos is *91.30%*. The average analysis time for detecting the quality visualization of appendiceal orifice in a colonoscopic procedure was *38* minutes and *17* seconds. The detection time is fast enough to confirm the orifice visualization quality to the patient before he/she leaves the endoscopic screening facility.

7.2 Detection of Retroflexion Image and Video

In Chapter 5, we introduced region and shape (RSL) based features, and edgeless edge cross-section profile (edgeless ECSP) calculation method and its features for retroflexion detection. We investigate different combination methods of these features for classification of retroflexion/non-retroflexion images and videos. The ensemble classifier using both ECSP and RSL features show promising performance for retroflexion detection in terms of the analysis time and detection rate.

Our retroflexion detection algorithm tested on *50* colonoscopy videos gave a *100%* specificity, *100.0%* precision, and *78.0%* accuracy running on the last *45* seconds of each video, and a *72.7%* specificity, *91.7%* precision, and *82.0%* accuracy running on the entire withdrawal phases of each video for detecting retroflexion videos. Our retroflexion detection technique is the first technique that can detect the appearance of retroflexion. The average execution time of our MATLAB analysis code is *0.46* seconds per image on a modern PC. The system can run in near real-time to assist the endoscopist during live colonoscopy using the frame rate of *2* fps. Our retroflexion detection system is promising for clinical practice in terms of both of the detection rate and analysis time.

7.3 Near Real-time Polyp Detection and Shot Extraction

In chapter 6, we introduced two phases of part-based ECSP for polyp detection. In the first phase, we presented an algorithm to segment multi-derivative ECSP and ROI into non-overlapping parts. We investigated new features obtained from parts, and compared it with two existing texture and color based state-of-the-art techniques for polyp image detection. This part-based ECSP method gives us flexibility to model or extract features from each part using the method component which is suitable for that part. Experimental results show that the proposed technique is robust to some challenging issues of polyp detection and outperforms existing leading methods. Our technique can effectively detect different types of protruding polyp appearance including polyps with little protrusion. It can mark detected polyp edges as visual feedback which is more precise and less intrusive than marking a large region as in a sliding-window based method. Furthermore, the false positives generated by our technique only appear in a small percentage of the number of images analyzed.

In the second phase, we presented the simplified features obtained from part-based ECSP module to further save the computational time. By combining the detected polyp edges from our polyp image detection method and an existing edge tracking method, our polyp detection system can correctly detect *97.7%* (*42* of *43*) of polyp shots in near real time under the frame analysis rate of *10* fps tested on *53* randomly selected video files captured from two endoscopy brands – FUJINON and OLYMPUS during routine screening colonoscopy. The average number of false shots per video is *36.2* and the average duration of each of these false shots is less than *1.5* seconds. Our system can run in near real-time to assist the endoscopist by providing visual feedback of a detected polyp as a warning signal on monitor during routine screening colonoscopy. To our best knowledge, our system is the first system that can detect polyp image and polyp shot in near real-time running on a modern PC. Our polyp detection and shot extraction system is promising for clinical use in terms of both of the detection rate and analysis time.

7.4 Future Work

Although our proposed algorithms show their effectiveness for detecting these colonoscopic objects, some challenging issues are still remaining in the future work. The main challenge issues in terms of detection rate include 1) blurry images that easily break a shot and frequently appear in colonoscopic video, 2) detection of appendiceal orifices that are not clearly seen, 3) detection of appendiceal orifices with other multi-structure shapes other than crescent shape, 4) detection of retroflexion operation that performs very short duration, 5) detection of polyps with a tiny size in an image or with weak illumination from the light source, 6) removal of falsely detected polyp images including protruding colon folds with smooth surface, protruding appendiceal orifice, ileocecal valve and smooth water.

Towards improving the analysis time in the future, our final goal in the future is to achieve a real-time detection (less than *33* ms analysis time per image) for detecting all three types of proposed representative colonoscopic objects. Based on the performance of near real-time polyp detection in terms of both analysis time and detection rate, we believe that we can optimize our appendix detection and retroflexion detection algorithms by simplifying our feature extraction steps. We briefly propose some improvement for detecting the three types of colonoscopic objects as follows.

Appendix image/video detection: The future work of appendix detection includes reducing the analysis time of image detection by converting the analysis code from MATLAB to C/C++ and further optimizing through multi-threading and Graphical Processing Units (GPU) for parallel execution. By reducing the analysis time, we may apply the similar edge tracking method presented in Chapter 6 for tracking appendiceal orifice edges to further reduce falsely detected edges and improve the overall detection rate. The integrated appendix detection module and polyp detection module in our system may share the same output from edge tracking module to save the computational time.

Retroflexion image/video detection: The future work of retroflexion detection includes further optimizing the execution time and converting the analysis code from MATLAB to C/C++. The bottleneck of analysis time is the dilation operation in the image preprocessing step. The dilation operation generates a region profile of shape and location of the endoscope in image. In the future work, we may try to replace the dilation operation by constructing the approximated shape and location of an endoscope. For example, we may divide an image into non-overlapping blocks and present the shape of an endoscope using the centers of blocks which have significant overlap with endoscope pixels. By connecting the centers of these blocks, we may construct the approximated shape and location of the endoscope in image and obtain RSL and ECSP features for retroflexion detection. By reducing the analysis time, we can apply temporal information to further reduce the number of falsely detected retroflexion images and improve the overall detection rate. As the preprocessing step dilates each pixel independently, we may also use GPU for parallel execution of the preprocessing step to speed up our retroflexion detection algorithm.

Polyp image detection/shot extraction: The future work of our polyp detection is to further reduce the computational time by optimizing both the algorithm and analysis code to achieve a *30* fps analysis rate for each module, and use our system in clinical trial during live colonoscopy to optimize the functionalities based on the practice from endoscopist.

The future work of ECSP based techniques is to extend them to detect other types of colonoscopic objects such as diverticula, lumen and vessel, and analyze other endoscopy procedures, such as laparoscopy, upper gastrointestinal endoscopy, wireless capsule endoscopy and EGD. As a popular local appearance model, our research on ECSP based technique may bring a broad interest to investigate them in detecting other objects in both medical and non-medical fields.

We believe that our future work to incorporate retroflexion detection and polyp detection system into clinical practice for routine screening colonoscopy has potential to reduce the polyp miss rate and train new endoscopists. The future use of our appendix detection system for documenting the quality of colonoscopy may also contribute to the decline in the polyp miss rate. The automatic documentations of detected semantic units of these three colonoscopic object types can be helpful to discover unknown patterns of colorectal cancers or new diseases and used as educational resources for endoscopic research.

BIBLIOGRAPHY

[1] "Colorectal cancer facts & figures," American Cancer Society, [Online]. Available: http://www.cancer.org/downloads/STT/F861708_finalforweb.pdf. [Accessed 2008 - 2010].

[2] D. Lieberman, "Quality and colonoscopy: a new imperative," *Gastrointestinal Endoscopy,* vol. 61, no. 2, p. 392–394, 2005.

[3] 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, no. 7, p. 1268–1279, 2007.

[4] Y. Cao, W. Tavanapong, K. Kim, J. Oh, P. C. de Groen, "A Framework for Parsing Colonoscopy Videos for Semantic Units," in *Proc. of the IEEE International Conference on Multimedia and Expo*, Taipei, Taiwan, 2004.

[5] A. Pabby, R. E. Schoen, J. L. Weissfeld, 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," *Gastrointestinal Endoscopy,* vol. 61, no. 3, pp. 385-391, 2005.

[6] K. Rex, J. H. Bond, S. Winawer, T. R. Levin, R. W. Burt, and D. A. J. et al, "Quality in the technical performance of colonoscopy and the continuous quality improvement process for colonoscopy: recommendations of the U.S. multi-society task force on colorectal cancer," *American Journal Gastroenterol*, vol. 97, no. 6, pp. 1296-1308, 2002.

[7] C. D. John son, J. G. Fletcher, R. L. MacCarty, et al, "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, no. 3, pp. 672-80, 2007.

[8] N. N. Baxter, M. A. Goldwasser, L. F. Paszat, R. Saskin, D. R. Urbach, and L. Rabanek, "Association from Colonoscopy and Death From Colorectal Cancer: A Population-Based, Case-Control Study," *Annals of Internal Medicine,* vol. 150, no. 1, pp. 1-9, January 2009.

[9] D. G. Hewett, C. J. Kahi, and D. K. Rex, "Does Colonoscopy Work?," *Journal of the National Comprehensive Cancer Network*, vol. 8, pp. 67-77, 2010.

[10] Rohit Gupta, Michael Steinbach, Karla Ballman, Vipin Kumar, Petrus C. de Groen, "Colorectal Cancer Despite Colonoscopy: Critical Is the Endoscopist, Not the Withdrawal Time," *Gastroenterology Supplement,* vol. 135, no. 5, pp. A-55, 2009.

[11] S. Ameling, S. Wirth, and D. Paulus, "Methods for polyp detection in colonoscopy video: a review," 14/2008.

[12] S. Hwang , J. Oh , J. Lee , Y. Cao , W. Tavanapong , D. Liu , J. Wong, P. C. de Groen, "Automatic measurement of quality metrics for colonoscopy videos," in *Proceedings of the 13th annual ACM international conference on Multimedia*, Hilton, Singapore, November 06-11, 2005.

[13] 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 Biom. Engineering,*, Innsbruck, Austria, 2007.

[14] J. M. Hanson, W. S. Atkin, W. J. Cunliffe et al., "Rectal retroflexion: an essential part of lower gastrointestinal endoscopic examination," *Dis Col Rectum,* p. 44:1706.–8, 2001.

[15] N. A. Melville, "Retroflexion of right colon during colonoscopy improves polyp yield," *Medscape Medical News from the American College of Gastroenterology (ACG), Annual Scientific Meeting and Postgraduate Course,* 2011 .

[16] S. Hwang, J. Oh, W. Tavanapong, J. Wong, and P.C. de Groen, "Polyp detection in colonoscopy video using elliptical shape feature," in *Proc. IEEE Int. Conf. Image Processing*, San Antonio, Texas, 2007.

[17] D. K. Iakovidis, D. E. Maroulis, and S. A. Karkanis, "An intelligent system for automatic detection of gastrointestinal adenomas in video endoscopy," *Computers in Biology and Medicine,* vol. 30, no. 10, p. 1084–1103, October 2006.

[18] S. M. Krishnan, X. Yang, K. L. Chan, S. Kumar, and P. M. Y. Goh., "Intestinal Abnormality Detection from Endoscopic Images," in *Proceedings of the 20th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, 1998.

[19] B. V. Dhandra, Ravindra Hegadi, Mallikarjun Hangarge, and V. S. Malemath, "Analysis of Abnormality in Endoscopic Images using Combined HSI Color Space and Watershed Segmentation," in *In Proceedings of the 18th International Conference on Pattern Recognition*, 2006.

[20] J. Kang and R. Doraiswami, "Real-Time Image Processing System for Endoscopic Applications," in *IEEE Canadian Conference on Electrical and Computer Engineering,* 2003.

[21] P. Wang, S. M. Krishnan, C. Kugean, and M. P. Tjoa, "Classification of endoscopic images based on texture and neural network," in *Proceedings of the 23rd Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, 2001.

[22] S. A. Karkanis, Dimitrios K. Iakovidis, Dimitrios E. Maroulis, Dimitris A. Karras, and M. Tzivras, "Computer-Aided Tumor Detection in Endoscopic Video using Color Wavelet Features," *IEEE Transactions on Information Technology in Biomedicine,* vol. 7, no. 3, p. 141–152, 2003.

[23] L. Alexandre , J. Casteleiro and N. Nobre, "Polyp detection in endoscopic video using SVMs," in *Proceedings of the 11th European conference on Principles and Practice of Knowledge Discovery in Databases*, 2007.

[24] P. Li, K. L. Chan, and S. M. Krishnan, "Learning a multi-size patch-based hybrid kernel machine ensemble for abnormal region detection in colonoscopic images," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2005.

[25] M. Tjoa and S. Krishnan, "Feature Extraction for the Analysis of Colon Status from the Endoscopic Images," *BioMedical Engineering OnLine,* vol. 2, no. 1, p. 9, 2003.

[26] S. A. Karkanis, D. K. Iakovidis, D. A. Karras, and D. E. Maroulis, "Detection of lesions in endoscopic video using textural descriptors on wavelet domain supported by artificial neural network architectures," in *IEEE International Conference on Image Processing*, 2001.

[27] G. D. Magoulas, V. P. Plagianakos, D. K. Tasoulis, and M. N. Vrahatis, “Tumor detection in colonoscopy using the unsupervised k-windows clustering algorithm and neural networks,” in *5th European Symposium on Biomedical Engineering*, 2004.

[28] G. D. Magoulas, V. P. Plagianakos, and M. N. Vrahatis, “Improved neural networkbased interpretation of colonoscopy images through on-line learning and evolution,” *European Network of Excellence on Intelligent Technologies for Smart Adaptive Systems,* p. 38–43, 2001.

[29] G. D. Magoulas, V. P. Plagianakos, and M. N. Vrahatis, “Neural network-based colonoscopic diagnosis using on-line learning and differential evolution,” *Applied Soft Computing,* vol. 4, no. 4, p. 369–379, 2004.

[30] D. C. Cheng, W.C. Ting, Y.F. Chen, Q. Pu, X.Y. Jiang, “Colorectal Polyps Detection Using Texture Features and Support Vector Machine,” in *Proc. of International Conf. on Advances in Mass Data Analysis of Images and Signals in Medicine, Biotechnology, Chemistry and Food Industry*, 2008.

[31] S. Ameling, S. Wirth, D. Paulus, G. Lacey, and F. Vilariño, “Texture-based polyp detection in colonoscopy,” in *Bildverarbeitung fürdie Medizin*, 2009.

[32] Y. Cao, D. Liu, W. Tavanapong, J. Wong, J. Oh, and P. C. de Groen, “Automatic Classification of Image with Appendiceal Orifice in Colonoscopy Videos,” in *Proc. IEEE Int. Annu. Conf. Engineering in Medicine and Biology Society*, New York City, NY, USA, 2006.

[33] T. R. Levin, “Colonoscopy Capacity,” *Gastroenterology,* pp. 1841-1849, December 2004.

[34] T. F. Cootes, C. J. Taylor, D. H. Cooper, J. Graham, “Active shape models – their training and application,” *Computer Vision and Image Understanding,* vol. 61, no. 1, p. 38–59, 1995.

[35] F. Jiao, S. Li, H. Y. Shum, and D. Schuurmans, “Face alignment using statistical models and wavelet features,” in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, Wisconsin, 2003.

[36] Z. Wang, X. Xu, B. Li, “Bayesian tactile face,” in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, Alaska, 2008.

[37] Y. Tsin, Y. Genc, Y. Zhu and V. Ramesh, “Learn to track edges,” in *Proc. IEEE Int. Conf. Computer Vision*, Rio de Janeiro, Brazil, 2007.

[38] S. Chaudhuri, S. Chatterjee, N. Katz, M. Nelson, M. Goldbaum, “Detection of blood vessels in retinal images using two-dimensional matched filters,” *IEEE Trans. Medical Imaging,* vol. 8, no. 3, p. 263–269, September 1989.

[39] H. Azegrouz, E. Trucco, “Max-min central vein detection in retinal fundus images,” in *Proc. IEEE Int. Conf. Image Processing*, Atlanta, Georgia, 2006.

[40] V. Mahadevan, H. Narasimha-Iyer, B. Roysam, H. L. Tanenbaum, “Robust model-based vasculature detection in noisy biomedical images,” *IEEE Trans. Information Technology in Biomedicine,* vol. 8, no. 3, p. 360–376, September 2004.

[41] E. Claridge, A. Orun, “Modeling of edge profiles in pigmented skin lesions,” in *Proc. Medical Image Understanding and Analysis*, 2002.

[42] S. Belongie, J. Malik and J. Puzicha, “Shape matching and object recognition using shape contexts,” *IEEE Trans. On Pattern Analysis and Machine Intelligence,* vol. 24, no. 24, 2002.

[43] L. Fei-Fei, R. Fergus, and P. Perona, "Learning generative visual models from few training examples: an incremental Bayesian approach tested on 101 object categories," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition, Workshop on Generative-Model Based Vision*, 2004.

[44] R. Fergus, P. Perona and A. Zisserman, "Object class recognition by unsupervised scale-invariant learning," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, 2003.

[45] A. Torralba, K. P. Murphy, and W. T. Freeman, "Sharing visual features for multiclass and multiview object detection," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, 2004.

[46] D. Lowe, "Distinctive image features from scale-invariant keypoints," *Internatioal Jounrnal of Computer Vision*, vol. 60, no. 2, November 2004.

[47] N. V. Balaji, M. Punithavalli, "Machine Learning approach for object detection – a survey approach," *International Journal of Computer Science and Information Security,* vol. 8, no. 7, 2010.

[48] T. Liu, Z. Yuan, J. Sun, J. Wang, N. Zheng, X. Tang, HY. Shum, "Learning to detect a salient object," *IEEE Trans. On Pattern Analysis and Machine Intelligence,* vol. 33, no. 2, pp. 353-67, Feburary 2011.

[49] B. Li, M.Q.-H. Meng, "Computer-based Detection of Bleeding and Ulcer in Wireless Capsule Endoscopy Images by Chromaticity Moments," *Computers in Biology and Medicine*, vol. 39, pp. 141-147, 2009.

[50] L. Alexandre, N. N. Nobre, and J. C. Casteleiro, "Color and position versus texture features for endoscopic polyp detection," in *Proc. Int. Conf. Biomedical Engineering and Informatics*, Sanya, China, 2008 .

[51] Y. Deng and B. S. Manjunath, "Unsupervised segmentation of color-texture regions in images and video," *IEEE Trans. On Pattern Analysis and Machine Intelligence*, vol. 22, pp. 800-810, 2001.

[52] J. Yao, M. Miller, M. Franaszek, et al., "Colonic polyp segmentation in CT colonography-based on fuzzy clustering and deformable models," *IEEE Trans. Medical Imaging*, vol. 23, no. 11, pp. 1344-1352, 2004.

[53] S. B. Gokturk, C. Tomasi, et al., "A statistical 3-D pattern processing method for computer-aided detection of polyps in CT colonography," *IEEE Trans. Medical Imaging*, vol. 20, pp. 1251-1260, 2001.

[54] S. Hwang and M. E. Celebi, "Polyp detection in wireless capsule endoscopy videos based on image segmentation and geometric feature," in *Proc. IEEE Conf. Acou. Spe. and Sig. Pro.*, Dallas, 2010.

[55] S. Gross, M. Kennel, T. Stehle, J. Wulff, J. Tischendorf, et al., "Polyp segmentation in NBI colonoscopy," *Algorithmen — Systeme — Anwendungen Proceedings des Workshops, Bildverarbeitung für die Medizin, Springer Berlin Heidelberg*, vol. 22, no. 25, p. 252–256, 2009.

[56] Tanaka S, Oka S, Hirata M, et al., "Pit pattern diagnosis for colorectal neoplasia using narrow band imaging magnification," *Digestive Endoscopy*, vol. 18, no. S1, p. S52–S56, 2006.

[57] R. Lienhart, "Reliable Transition Detection in Videos: A Survey and Practitioner's Guide," *International Journal of Image and Graphics*, vol. 1, no. 3, pp. 469-486, 2001.

[58] A. Hanjalic, "Shot Boundary Detection: Unraveled and Resolved," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12, no. 2, pp. 90-105, February 2002.

[59] J. S. Boreczky and L. D. Wilcox, "A Hidden Markov Model Frame Work for Video Segmentation Using Audio and Image Features," in *Proc. of IEEE ICASSP*, Seattle, May 1998.

[60] Y. Qi, E. Hauptmann, and T. Liu, “Supervised classification for video shot segmentation,” in *Proc. of IEEE Int’l Conf. on Multimedia & Expo*, 2003.

[61] S. Lefevre, J. Hollera, N. Vincent, “A review of real-time segmentation of uncompressed video sequences for content-based search and retrieval,” *Real-Time Imaging*, vol. 9, pp. 73-78, 2003.

[62] I. Koprinska and S. Carrato, “Temporal video segmentation: a survey,” *Signal Process.: Image Commun*, vol. 16, no. 5, pp. 477 - 500, 2001.

[63] T. Kikukawa, S. Kawafuchi, “Development of an automatic summary editing system for the audio-visual resources,” *Trans. Electron. Inform.*, Vols. J75-A, pp. 204-212, 1992.

[64] B. Shahraray, “Scene change detection and content-based sampling of video sequences,” in *Proceedings of IS&T/SPIE*, 1995.

[65] W. Xiong, J.C.-M. Lee, M.C. Ip, “Net comparison: a fast and elective method for classifying image sequences,” in *Proceedings of SPIE Conference on Storage and Retrieval for Image and Video Databases III*, San Jose, CA, 1995.

[66] T. N. Pappas, “An adaptive clustering algorithm for image segmentation,” *IEEE Trans. Signal Process*, vol. 40, p. 901 – 914, 1992.

[67] A. Hampapur, R. Jain, T.E. Weymouth, “Production model based digital video segmentation,” *Multimedia Tools Appl*, vol. 1, no. 1, p. 9 – 46, 1995.

[68] P. Aigrain, P. Joly, “The automatic real-time analysis of film editing and transition elects and its applications,” *Comput. Graphics*, vol. 18, no. 1, p. 93 – 103, 1994.

[69] J. S. Boreczky, L. D. Wilcox, “A hidden Markov model framework for video segmentation using audio and image features,” in *Proceedings of International Conference on Acoustics, Speech, and Signal Processing*, Seattle, 1998.

[70] J. Oh, S. Hwang, W. Tavanapong, P. C. de Groen, and J. Wong, “Blurry frame detection and shot segmentation in colonoscopy videos,” in *In Proc. of IS&T/SPIE Symp. Electron. Imag.*, San Jose, CA, 2004.

[71] W. S. Cleveland, E. Grosse, and W. M. Shyu, Local regression models, New York: In J. M. Chambers and T. J. Hastie (eds.), Statistical Models, S. Chapman and Hall, 1992.

[72] “Polynomial curve fitting,” MathWorks, [Online]. Available: http://www.mathworks.com/help/techdoc/ref/polyfit.html.

[73] P. B. Cotton and C. B. Williams, Practical Gastrointestinal Endoscopy, The fundamentals, Fifth Edition ed., Blackwell Publishing, Online book-preview, 2003, pp. 151-152.

[74] W. K. Pratt, Digital Image Processing, John Wiley & Sons, Inc., 1991.

[75] I. H. Witten and E. Frank, Data Mining: Practical machine learning tools and techniques, 2nd Edition ed., San Francisco: Morgan Kaufmann, 2005.

[76] J. R. Quinlan, C4.5: Programs for Machine Learning, Morgan Kaufmann Publishers, 1993.

[77] J. Oh, S. Hwang, Y. Cao, W. Tavanapong, J. Wong, and P. C. de Groen, “Measuring objective quality of

colonoscopy," *IEEE Trans. on Biomedical Engineering,* vol. 59, no. 9, pp. 2190-2196, September 2009.

[78] J. Yuan, J. Li, and B. Zhang, "Learning Concepts from Large Scale Imbalanced Data Sets Using Support Cluster Machines," in *ACM Multimedia*, Santa Barbara, California, 2006.

[79] D. W. Marquardt, "An algorithm for least-squares estimation of nonlinear parameters," *Journal of the Society for Industrial and Applied Mathematics*, vol. 11, no. 2, p. 431–441, 1963.

[80] G. Isabelle , E. André, "An introduction to variable and feature selection," *The Journal of Machine Learning Research,* p. 3, 3/1/2003.

[81] R. Collobert, S. Bengio, et al., "Torch: a modular machine learning software library," IDIAP, 2002.

[82] H. Trevor, T. Robert and F. Jerome, The elements of statistical learning: data mining, inference, and prediction, 2nd Edition ed., Springer, 2009.

[83] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proceedings of Imaging Understanding Workshop*, 1981.

[84] A. Zelinsky, G. R. Bradski, Learning OpenCV---Computer Vision with the OpenCV Library, 2008.

[85] C. Harris and M. Stephens, "A combined corner and edge detector," in *Proceedings of the 4th Alvey Vision Conference,* 1988.

[86] J. Shi and C. Tomasi, "Good Features to Track," in *9th IEEE Conference on Computer Vision and Pattern Recognition*, 1994.

[87] S. Stanek, W. Tavanapong, J. Wong, J. Oh, R. Nawarathna, J. Muthukudage, P. C. de Groen., "SAPPHIRE middleware and software development kit for medical video analysis," in *24th International Symposium on Computer-Based Medical Systems,* Bristol, U.K., 2011.

[88] "Cplusplus.com," [Online]. Available: http://www.cplusplus.com/reference/clibrary/ctime/clock/.