City Research Online

City, University of London Institutional Repository

Citation: Williams, D. (2007). Visualisation of curved tubular structures in medical databases: An application to virtual colonoscopy. (Unpublished Doctoral thesis, City, University of London)

This is the accepted version of the paper.

This version of the publication may differ from the final published version.

Permanent repository link: https://openaccess.city.ac.uk/id/eprint/30945/

Link to published version:

Copyright: City Research Online aims to make research outputs of City, University of London available to a wider audience. Copyright and Moral Rights remain with the author(s) and/or copyright holders. URLs from City Research Online may be freely distributed and linked to.

Reuse: Copies of full items can be used for personal research or study, educational, or not-for-profit purposes without prior permission or charge. Provided that the authors, title and full bibliographic details are credited, a hyperlink and/or URL is given for the original metadata page and the content is not changed in any way.

City Research Online: http://openaccess.city.ac.uk/ publications@city.ac.uk

Visualisation of Curved Tubular Structures in Medical Datasets: An Application to Virtual Colonoscopy

David Williams

A thesis submitted in fulfilment of the
requirements for a PhD in Health Informatics

Centre for Health Informatics
City University
December 2007

Contents

Acknowledgement **5**

Declaration **6**

Abstract **7**

Glossary **8**

I Introduction and Existing Approaches **9**

1 Introduction **10**

1.1 Inflammatory Bowel Disease 11

1.2 System Overview 12

1.3 Research Focus 13

1.4 Structure of this Work 15

2 Data Acquisition **16**

2.1 Introduction 16

2.2 CT Scanner Technology 17

2.3 Patient Preparation 18

2.4 Data Representation 19

2.4.1 DICOM 20

2.4.2 Storing Data in Memory 21

2.5 Summary 22

3 Fundamentals of IBD Detection **23**

3.1 Introduction 23

3.2 Colon Segmentation Techniques 23

3.2.1 Inner Wall Segmentation 23

3.2.2 Outer Wall Segmentation 25

3.2.3 The Watershed Approach 25

3.3 Texture Analysis 26

3.4 Classification . . . 27
3.4.1 Determination of Classes . . . 27
3.4.2 Matching . . . 27
3.5 Summary . . . 28

4 General Purpose Medical Visualisation 30
4.1 Introduction . . . 30
4.2 Slice Examination . . . 30
4.2.1 Window Levelling . . . 31
4.3 Multiplaner Reformation . . . 32
4.3.1 Interpolation . . . 32
4.4 Volume Rendering . . . 34
4.4.1 Intensity Projection . . . 34
4.4.2 Shaded Surface Display . . . 35
4.4.2.1 Gradient Estimation . . . 37
4.4.3 Direct Volume Rendering . . . 37
4.5 Summary . . . 39

5 Visualising Tubular Structures 40
5.1 Introduction . . . 40
5.2 Optical Endoscopy . . . 40
5.3 Virtual Endoscopy . . . 42
5.3.1 Curved Planar Reformation . . . 43
5.3.2 Colon Unfolding . . . 44
5.3.3 Virtual Flythroughs . . . 45
5.3.3.1 The Virtual Flythrough Problem Domain . . . 45
5.3.3.2 Optimizations to the Raycasting Process . . . 46
5.3.3.3 Enhancements . . . 48
5.4 Motivation for a New Visualisation . . . 48
5.5 Summary . . . 49

II Volumetric Curved Planar Reformation 51

6 Generation of Volumetric CPR Images 52
6.1 Introduction . . . 52
6.2 Preparing the Dataset . . . 53
6.3 Principles of Volumetric CPR . . . 54
6.4 Comparison of Projection Methods . . . 56
6.4.1 Projected Mode . . . 56

6.4.2 Stretched Mode . . . 57
6.4.3 Straightened Mode . . . 61
6.5 Transfer Function Specification . . . 63
6.6 Restricting the Raycast Area . . . 64
6.7 Maximizing surface coverage . . . 66
6.8 Summary . . . 67

7 Integration with a virtual colonoscopy system 68
7.1 Introduction . . . 68
7.2 System Layout . . . 69
7.3 Image Precomputation . . . 70
7.4 Picking . . . 70
7.5 View Synchronization . . . 71
7.6 Summary . . . 73

8 Advanced Techniques and Uses 74
8.1 Introduction . . . 74
8.2 Overlays . . . 74
8.2.1 Surface Coverage Tool . . . 75
8.2.2 Translucency Window . . . 77
8.3 Visualising Behind Fluid . . . 78
8.4 Lighting Strategies . . . 79
8.4.1 Directional Local Lighting . . . 80
8.4.2 Positional Local Lighting . . . 81
8.4.3 Global Lighting . . . 82
8.5 Shape Driven Lighting . . . 85
8.5.1 Direction Driven Lighting . . . 85
8.5.2 Curvature Driven Lighting . . . 86
8.6 Summary . . . 88

III Evaluation and Conclusion 89

9 Evaluation 90
9.1 Introduction . . . 90
9.2 Methodology . . . 90
9.3 Quantitative Evaluation . . . 91
9.3.1 Image generation time . . . 92
9.3.1.1 MPR . . . 93
9.3.1.2 Flythrough . . . 93

9.3.1.3 Volumetric CPR 93
9.3.1.4 Unrolling 94
9.3.1.5 Comparison 95
9.3.2 Surface coverage 96
9.3.2.1 Determining the total area 96
9.3.2.2 Determining the seen area 97
9.3.2.3 Comparison 98
9.3.3 Memory requirements 98
9.3.3.1 Comparison 100
9.4 Formative Evaluation 100
9.4.1 System Layout 101
9.4.2 Non-Rotating Volumetric CPRs 102
9.5 Summative Evaluation 102
9.5.1 Radiological Perspective 102
9.5.2 IBD Perspective 103
9.6 Summary 104

10 Conclusion 105

11 Future Work 108
11.1 Introduction 108
11.2 IBD visualisation 109
11.2.1 Abnormality Based Transfer Functions 109
11.2.2 Using the context information 110
11.3 Polyp Recognition in the Volumetric CPR 111
11.4 Volumetric CPR for CT Angiography 113
11.4.1 Flow Visualisation 115
11.5 Hardware Accelerated Image Generation 115
11.6 Stereoscopic Image Generation 116
11.7 Summary 119

Bibliography 120

A Volumetric Curved Planar Reformation for Virtual Endoscopy 127

B Volumetric CPR as an Enhancement to Virtual Colonoscopy Systems 139

C O-Buffer Based IFT Watershed from Markers for Large Medical Datasets 144

D Feedback Questionnaires 161

Acknowledgement

I would like to thank my supervisors, Professor Abdul Roudsari and Dr Haralambos Hatzakis, for their guidance and support over the past three years. They kept me on track, helped keep the research focused, and provided advice along every step of the way. Without them I am sure I wouldn't be in the position I am today.

My colleagues at City University are owed a great deal for their friendship over the last three years. Juan Adriano, Sanaa Henni, and Qi Ma have been helpful and supportive during the more difficult parts of the research and the whole experience would have been less enjoyable without them.

Lastly I would like to express my appreciation for the team at Biotronics3D, specifically Sören Grimm, Ernesto Coto, and Elnaz Gederi, for their ideas and the numerous discussions we engaged in. They were also instrumental in providing the supporting infrastructure for this project to develop.

Declaration

I grant powers of discretion to the University Librarian to allow this thesis to be copied in whole or in part without further reference to me. This permission covers only single copies made for study purposes, subject to normal conditions of acknowledgement.

Abstract

Medical conditions affecting the colon are problematic to diagnose due to the difficulty in examining this particular internal organ. To date, the most widely used approach is to perform a colonoscopy; a procedure in which a small camera is inserted into the colon to examine its surface. This procedure is unpleasant and potentially dangerous for the patient, and is expensive and time consuming for the hospital. As a result, patients at risk of developing the conditions are not always screened as often as would be desirable.

Over the last few years a new approach known as *virtual* colonoscopy has been gaining popularity. The method uses information from a CT scan to reconstruct a 3D model of the colon which can then be examined without the patient needing to undergo a colonoscopy. This approach is now commonly used when screening for polyps (an indication of colon cancer) but can not be so easily used on conditions such as Inflammatory Bowel Disease (IBD) where information beyond the shape of the surface is required.

This thesis forms part of a larger project which aims to diagnose conditions such as IBD by using image processing algorithms on CT data and presenting the results to the user in an easy to interpret way. Specifically we are concerned with this *visualisation* stage of the system and so have developed a new visualisation approach which we call Volumetric CPR. This can be used to supplement the more traditional virtual flythrough visualisation and is applicable to IBD detection as well as screening for polyps.

Our technique builds on the concept of Curved Planar Reformation (CPR), which has proved to be a practical and widely used tool for the visualisation of curved tubular structures within the human body. It has been useful in medical procedures involving the examination of blood vessels and the spine. However, it is more difficult to use it for structures such as the colon because abnormalities are smaller relative to the size of the structure and may not have such distinct density and shape characteristics.

Our new approach improves on this situation by using volume rendering for hollow regions of the structure and standard CPR for the surrounding tissue. This effectively combines gray scale contextual information with detailed colour information from the area of interest. The approach is successfully used with each of the standard CPR types and the resulting images are promising as an alternative for virtual colonoscopy.

We also demonstrate how systems can effectively utilize this new visualisation in order to convey maximum information to the user. We show how overlays can be used to present surface coverage data and how sophisticated lighting models can improve the users understanding of the 3D structure. We also present details of how to integrate our visualisation into existing systems and work flows.

Glossary

AvIP	Average Intensity Projection
CAD	Computer Aided Detection
CARS	Computer Aided Radiology and Surgery
CEF	Contrast Enhanced Fluid
CFD	Computational Fluid Dynamics
CPR	Curved Planar Reformation
CPU	Central Processing Unit
CT	Computed Tomography
CTA	Computed Tomography Angiography
CTC	Computed Tomography Colonoscopy
DICOM	Digital Imaging and Communications in Medicine
DVR	Direct Volume Rendering
FPS	Frames Per Second
GPU	Graphics Processing Unit
HU	Houndsfield Unit
IBD	Inflammatory Bowel Disease
IFT	Image Foresting Transform
ITK	Insight Tool Kit
MIP	Maximum Intensity Projection
MPR	Multi Planar Reformation
MRI	Magnetic Resonance Imaging
PACS	Picture Archiving and Communication Systems
PVE	Partial Volume Effect
RAM	Random Access Memory
RGBA	Red, Green, Blue, Alpha
RSNA	Radiological Society of North America
SSD	Shaded Surface Display
TVCG	Transactions on Visualization and Computer Graphics

Part I

Introduction and Existing Approaches

Chapter 1

Introduction

Since its introduction in the 1970's, Computed Tomography (CT) has revolutionised the way in which many conditions are diagnosed and treated. The ability to examine structures inside the body, without resorting to surgery, has allowed clinicians to diagnose problems and plan corrective procedures with a minimum of risk to the patient. This PhD work forms part of a larger project which aims to use CT (and other modalities) to automatically detect the presence of Inflammatory Bowel Disease (IBD) in the colon. This PhD work is being undertaken as a joint venture between City University and a London-based medical imaging company called Biotronics3D; the eventual goal is to incorporate the research into Biotronics3D's suite of products. Biotronics3D own the intellectual property arising from this work but have been keen to publish it in recognised journals. They also hold a patent on the resulting system, and in return are providing data sets and expertise.

The purpose of this introductory chapter is to provide an overview of both the project as a whole and of the parts of it which constitute the PhD. We begin in Section 1.1 with a description of IBD in its two most common forms - highlighting the similarities and differences between each. We also give information about the prevalence and distribution of the condition. Most importantly, we observe than the current 'gold standard' for the diagnosis of the condition is a colonoscopy; an unpleasant procedure for the patient which involves inserting a camera into the colon. We note that the pain and discomfort associated with this procedure can hinder screening programs and this provides our motivation for finding a less invasive and more automated approach.

Then, in Section 1.2, we provide a high level overview of an automated system for the detection of IBD from CT data. The actual PhD only comprises a small part of this complete system but it is useful to consider an overview for two reasons. Firstly it is important to understand how different parts of the system connect and interact. Secondly, it was initially undecided which parts of the system would form the PhD. Preliminary research was therefore carried out on all parts, and we present this research in later chapters.

Eventually, visualisation was chosen as the area which would form the majority of the PhD work. The reasons behind this decision are discussed later but it had one important effect in that, because visualisation is the last component of the system, it is not possible to visualise IBD data until the rest of the system has been completed. Hence we introduced a more immediate goal of applying our visualisation research to the problem of screening for colorectal cancer, because in this case the raw data can be visualised. Section 1.3 describes this decision more fully.

Finally we describe the structure of the remainder of the thesis in Section 1.4.

1.1 Inflammatory Bowel Disease

The project is concerned with the detection of a family of conditions known as Inflammatory Bowel Diseases (IBD) and, within this family, the two most common conditions are Crohn's Disease [1] and Colitis [2]. In terms of symptoms the two have much in common; they are both characterized by an inflammation of the bowel wall due to an abnormal response in the body's defence mechanism. There are differences however; Colitis tends to be confined to the colon whereas Crohn's disease can affect any part of the gastrointestinal tract. Also, Colitis usually has a more uniform distribution compared to Crohn's disease (which may have patches of diseased and healthy tissue). Crohn's disease can also lead to the development of fistulae as a result of severe ulcers breaking through the bowel wall. These fistulae are channels which may tunnel through into other organs or even reach the skin and become external.

IBD as a whole primarily affects North America and Europe (with a prevalence rate of 149 per 100,000) although there are cases throughout the world. Young adults are most at risk and the condition affects both men and women equally. There are also strong indications that genetics play a part in who develops the condition as many sufferers have a close relative who has also been diagnosed with IBD. Due to the nature of the disease it can be quite difficult and time-consuming to diagnose accurately. One popular method involves the use of a Barium meal in which Barium (a soft, white, metal which is not easily penetrated by x-rays) is given to the patient in liquid form. This lines the gastrointestinal tract and causes it to show up much more clearly when x-rays are then taken. This is relatively comfortable (and safe) for the patient and the financial cost is quite low; however the quality of the resulting images are not particularly good.

A less pleasant, but ultimately more decisive method is to perform a colonoscopy; this involves inserting a camera into the colon to allow a visual analysis of the affected area to take place. It may also be necessary for samples of tissue to be taken for further analysis; this is known as a 'biopsy'. This is clearly an unpleasant and potentially dangerous procedure (one in 500 to 1000 cases results in colonic perforation and one in 2000 to 5000 cases results in death [3]), and this can create a reluctance on the part of the patient to take part (which hinders early diagnosis). For some more serious conditions (such as colorectal cancer) early diagnosis can be extremely important; for this purpose *virtual* colonoscopy is starting to show promise.

Virtual colonoscopy requires a CT scan to be performed on the patient's abdomen and from this data a 3D model of the colon is constructed. A doctor is able to perform a virtual flythrough of the colon and spot any polyps or other symptoms of colorectal cancer. A CT scan is fast and pain free, though there is usually some preliminary preparation to clear the bowel for optimal viewing and to enhance the visibility of any remaining fluid. With an optical colonoscopy (i.e. using a physical endoscope) there is a maximum depth to which the camera can be inserted; virtual colonoscopy does not suffer from this limitation but does have the limitation that tissue samples cannot be taken for later analysis.

There are also cost and time benefits associated with a virtual colonoscopy. Nicholson [4] observed that a virtual colonoscopy takes about 20 minutes whereas an optical colonoscopy takes about 30 minutes. However, when using the optical approach a patient spends all day at the hospital and may even require a bed due to the anaesthetic required for the procedure. Taking this into account puts the cost of a virtual colonoscopy at about 40% of that for the conventional approach.

It is, however, considerably more difficult to use a virtual colonoscopy for the detection of IBD (compared to colorectal cancer). Colorectal cancer causes noticeable changes to the surface of the colon which the doctor can observe, but the inflammation due to IBD can be difficult to spot with a virtual surface analysis. The aim of the project as a whole is therefore to develop ways of identifying and visualising infected tissue within the colon.

1.2 System Overview

The system for the detection of IBD is a pipeline into which the raw CT data is fed, and from which emerges a visualisation allowing the user to identify areas affected by IBD. Each stage of the pipeline implements image processing or visualisation operations and feeds the output into the next stage. Figure 1.1 provides an overview of the system in which new research areas are marked in blue, areas with some existing research are marked in red, and areas which are largely solved are marked in white. A description of each stage is given below, although full details of the related work are reserved until Chapters 3 to 5.

Preprocessing: Preprocessing involves cleaning up the image to reduce or eliminate common problems such as:

- Motion artefacts which result from a patients breathing or heart beat.
- Partial volume effect due to material boundaries being sampled at too low a frequency.

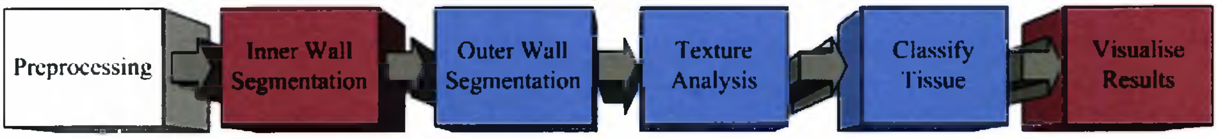


Figure 1.1: An overview of the IBD detection system. White blocks indicate significant existing research, red blocks indicate some existing research, and blue blocks indicate little existing research within our problem domain.

- Random noise due to equipment limitations.

Inner Wall Segmentation: Several approaches for the segmentation of the inner wall (the boundary between air in the colon and colon tissue) have been previously developed and applied within the context of virtual colonoscopy for polyp detection. These approaches can be applied to our new problem domain largely unchanged.

Outer Wall Segmentation: Segmentation of the outer wall (the boundary between colon tissue and other tissue) is considerably more difficult because there is not such a clear difference in intensities. It is an important research area because, once solved, it allows all parts of the scan which correspond to colon wall tissue to be identified.

Texture Analysis: Other researchers have successfully applied texture analysis to the identification of tissue type, for example in the brain. Within this project it will be used to determine properties of the tissue such as roughness, uniformity, and contrast, which can then provide an indication of whether the tissue is healthy.

Classify Tissue: Numerous classification techniques have been developed in the past and include neural networks, Bayes classifiers, and support vector machines. Within our project the chosen classification method will be responsible for assigning to each part of the colon wall a probability of being healthy or not, most likely based on a series of training data sets.

Visualise Results: Visualisation of the colon has been actively researched from the perspective of viewing the *surface*, but still has a lot of problems as will be discussed in Chapter 5. There has also been little work on allowing the user to see *behind* the surface; this is important for IBD visualisation.

1.3 Research Focus

The first six months of the PhD were spent performing a literature review and researching all of the areas described in the previous section. After this period it was decided that the PhD work should focus on the *visualisation* aspect due to existing expertise in this area. This visualisation research is the primary focus of this thesis, but this choice of visualisation as a research area introduces some additional problems in that it is the final stage of the system as depicted by Figure 1.1. Hence its usefulness in visualising IBD data cannot be easily determined until the earlier stages are complete.

However, there are numerous other colon screening problems which require the visualisation of the colon surface. One of the most important is colorectal cancer, so we have chosen to use this as another problem which our visualisation should address. It has been the subject of significant previous research and so datasets and diagnoses are widely available. Actually, the visualisation which we eventually develop in Part II is even more general than this and can be applied to *any* tubular structure in the body (such as the trachea or aorta) but, in the interests of keeping as close as possible to our original problem of IBD visualisation, we are mostly concerned with applying it to the colon.

Colorectal cancer is the second most common cause of cancer related death in the western world (following lung cancer). It generally affects the more senior members of society with most cases occurring in patients in their 60s and 70s, but can strike younger people if there are aggravating conditions such as a family history of early colon cancer. There is also a correlation between occurrence of colorectal cancer and lifestyle aspects such as smoking, alcohol consumption, and diet. One of the early indications of colorectal cancer is the formation of polyps on the surface of the colon. Hence it is common practice for those deemed to be at risk of developing the disease to undergo regular screening program to assess any developments. If caught at an early enough stage these polyps can be removed before they become malignant – greatly increasing the patients chances of survival. As with IBD, the 'gold standard' for the detection of polyps is a colonoscopy.

In summary, the overall goals of this PhD research are to:

- Develop a visualisation method for the screening of CT colonoscopy datasets. The visualisation should be suitable for:
    - Identifying polyps which are an indication of colorectal cancer.
    - Recognising regions of tissue which are inflamed due to IBD.
    - Displaying the output of our IBD detection pipeline.
- Integrate our visualisation into the Biotronics3D virtual colonoscopy system such that it can be used alongside other existing approaches.
- Evaluate how well our approach compares to other solutions to similar problems. This evaluation will also be done in partnership with Biotronics3D.

We provide further details on the requirements of our system in Chapter 5, once we have reviewed the existing work in the field.

1.4 Structure of this Work

This thesis is structured as follows. The remainder of Part I (comprising Chapters 2 to 5) covers all the background material and related research for the project. Chapter 2 begins this with a description of the operation of CT scanners and an explanation of how the resulting data is stored. This is important because it forms the input to our system. Chapter 3 then describes all the background research relating to aspects of the system other than visualisation, and so describes segmentation approaches, techniques for texture analysis, and various solutions for classification. Chapters 4 and 5 then provide a much more detailed review of existing medical visualisation approaches; Chapter 4 describing the underlying concepts and Chapter 5 describing their application to virtual endoscopy. Chapter 5 finishes by providing motivation for a new visualisation approach.

Part II then presents the *Volumetric CPR* as our new solution for the visualisation of curved tubular structures. Chapter 6 describes the basics of the algorithm before Chapter 7 discusses the problems and solutions with integrating our new approach with existing visualisation techniques. Finally, Chapter 8 demonstrates some more advanced ways in which it can be used to provide the user with additional information.

After this, Part III brings the thesis to a close with an evaluation of the system and our conclusions about its usefulness. We begin this in Chapter 9 with an evaluation of our technique in which we demonstrate ways in which it is technically superior to previous approaches. We also present feedback from radiologists giving their impression of the system. Our conclusions are then presented in Chapter 10 along with a list of what we consider to be our key contributions to the field. Finally we give some insight into potential extensions and applications of the work in Chapter 11.

Chapter 2

Data Acquisition

2.1 Introduction

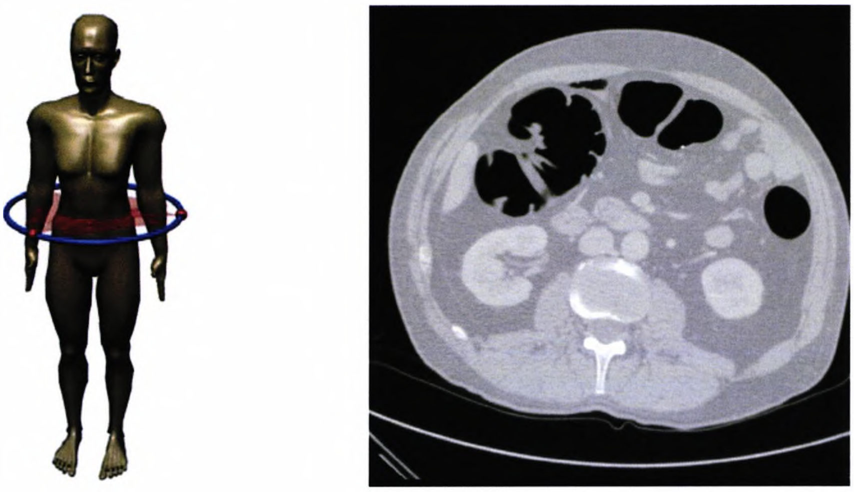
When a radiologist wishes to obtain a scan of a patient there are several types, or *modalities*, to choose from, each with its own characteristics and advantages [5]. Among the most common modalities are:

Computed Tomography: Rotates an x–ray source around the body to determine the degree to which x–rays are absorbed at each point. Effective for examining high density structures such as bone though it can also be used on tissue. Has the disadvantage that the patient is subjected to relatively high levels of radiation.

Magnetic Resonance: Rather than using high frequency ionizing radiation such as x–rays, MRI scanners use lower frequency radio waves in conjunction with a magnetic field in order to generate images. For this reason they are considered to be safer than CT scanners but have been in use for a shorter period of time. MRI scanners are more effective than CT for examining soft tissue within the body.

For the automatic classification of colon wall tissue through texture analysis it is anticipated that MRI will be required because of the superior imaging of soft tissue. However, for our more short–term goal of colonic polyp detection CT is sufficient and has been widely used for this purpose [6, 7, 8, 9, 10, 11]. Because this thesis is focused on the visualisation aspect of the system, the actual modality of the underlying data is not important - our technique is sufficiently generic as not to be tied to a particular modality.

For this reason we have been able to develop our visualisation system based on CT data; this is significantly easier to come by because MRI imaging of colon polyps is not a common procedure. In this section we explain the process of acquiring a CT data set and describe its representation in different parts of our system.



(a) (b)

Figure 2.1: A single CT slice taken through the abdomen (a) results in an image like that shown in (b).

2.2 CT Scanner Technology

A CT scanner is a medical imaging device which creates a number of 2D 'slices' through the object being scanned. Each slice is created by rotating an X-Ray source and a detector around the body and taking a number of X-Rays from different directions. These X-Rays are then combined to form a 2D slice through the object using an approach known as 'Tomographic Reconstruction'. This is illustrated in Figure 2.1.

Current generation scanners typically create slices with a resolution of 512×512 pixels and with a bit depth of 12 bits per pixel (stored as 2 bytes). This means each image occupies half a megabyte, but the next generation of scanners are already becoming available; these have a resolution of 1024×1024 and require approximately two megabytes per image. Each pixel is a 12 bit number representing the radiodensity at the corresponding location; this is converted to a greyscale intensity using a technique called 'Window–Levelling' which will be described later.

A material's radiodensity is a measure of the extent to which it absorbs x-rays and is measured in Houndsfield Units (HU). The HU scale is defined such that, at standard temperature and pressure, air has a radiodensity of $-1000HU$ and distilled water has a radiodensity of $0HU$. Given this scale, some typical radio-densities for materials found in the body are given in Table 2.1 (as originally specified in [5]). It can be seen that the range (approximately $-1000HU$ to $1000HU$) easily fits within the 12 bits allowed for each pixel as 12 bits allows $2^{12} = 4096$ distinct values.

As mentioned previously, a scanner will usually take a number of slices at a time. This might be just a few in the case of examining extremities such as the hand or foot, but in the case of a full-body scan (used in CT Angiography, for example) it could be up to

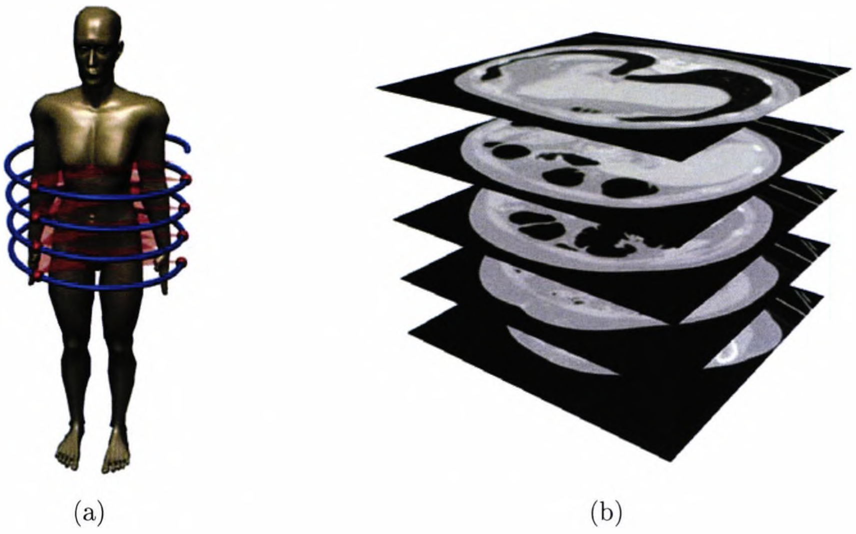


Figure 2.2: A scanner will typically take a series of slices (a), often several hundred. When stacked on top of each other (b) these slices conceptually form a 3D volume.

2000. These slices are usually about $1mm$ apart but next generation scanners are again expected to improve this and decrease the slice distance to $0.5mm$. This will result in higher quality and higher resolution scans.

Figure 2.2 shows a series of scans taken through the abdomen. It is important to realise that, although so far we have referred to a 'series of slices', it is possible to consider these slices to form a 3D volume of data. This is directly analogous to the way a series of 1D lines can form a 2D image on a computer monitor. For the majority of this work we consider the data to be in this conceptual form and as such each 3D element of the data, or *voxel,* is referenced by a set of (x, y, z) coordinates.

2.3 Patient Preparation

Before being put through the CT scanner for the purpose of virtual colonoscopy, it is usually necessary for the patient to undergo some preparation in order to ensure that material in the colon does not obscure polyps or other abnormalities. The exact procedure

Material	Radiodensity
Bone	400 – 1000
Soft Tissue	40 – 80
Water	0
Fat	-60 – -100
Lung	-400 – -600
Air	-1000

Table 2.1: Radiodensity in Houndfield Units for materials commonly found in the body.

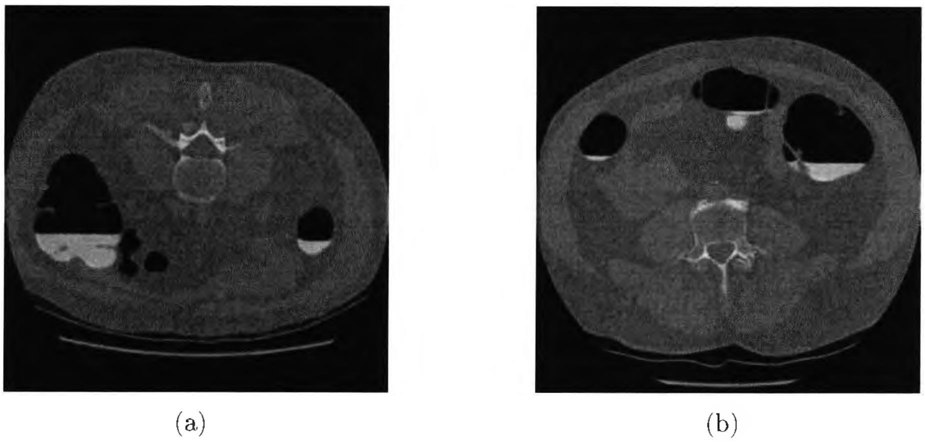

(a) (b)

Figure 2.3: The same slice through the abdomen with the patient lying in (a) prone and (b) supine position. This causes the contrast enhanced fluid to move, ensuring that every part of the colon surface is visible in at least one scan.

for this varies, but basically consists of removing as much material as possible from the colon and/or tagging any remaining material so that it can be identified in the CT scan.

The patient is given some form of laxative 24 hours before the examination and then at regular intervals until the examination is performed. The patient may also be given a drink containing barium – a high density element which shows up bright white on CT scans. This is absorbed into any remaining material allowing it to be clearly identified and, in some systems, to be removed automatically via an 'electronic cleansing' process. However, when performing such an electronic cleansing there is a risk that it will be too aggressive (thereby removing polyps) or too conservative (thereby leaving a layer of material on the surface of the colon – potentially hiding polyps). For this reason electronic segmentation may sometimes not be performed, and the patient may instead have *two* scans in the prone and supine positions (laying on the front and the back) as shown in Figure 2.3.

During the scan the colon is inflated with Carbon Dioxide in order to prevent opposing walls from touching each other. This ensures that the maximum amount of the wall can be seen and also prevents the flight path of the virtual camera from being blocked. However, in some cases patient discomfort may prevent the colon being inflated very far.

2.4 Data Representation

When operating on large medical datasets it is important to have an understanding of how the data is stored. This has a direct influence on the operations which can be performed on the data and the efficiency with which they can be carried out. There are several aspects to consider:

- How is the data stored on disk?

- How is the data accessed from remote locations?
- How is the data stored in memory by the application?

For the purposes of this work we are mostly interested in how the data is stored in memory by the application, as this has the most effect on the performance and behaviour of our system. However, in the interests of completeness, we also look at other aspects of data storage.

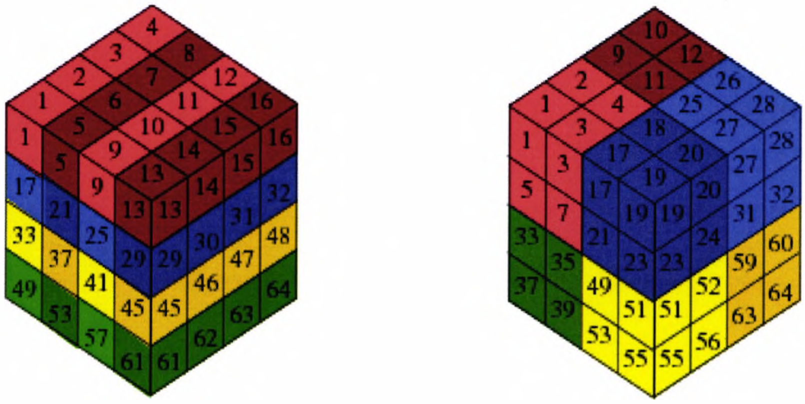
2.4.1 DICOM

Digital Imaging and Communications in Medicine (DICOM) is a standard [12] devised by the National Electrical Manufacturers Association (NEMA) to provide a means of handling medical images between different compliant devices. The standard describes many aspects including the transmitting, storing, and printing of data and thereby aids integration of various devices with hospital Picture Archiving and Communication Systems (PACS).

A medical imaging application such as the system which is described in this thesis will typically get its data from one of two places; a PACS server or a local file system. In turn, a PACS server may be running on a remote server across a network or may be running locally on the machine. However, the actual location of the PACS server is not important to the client application as it will receive its data via the DICOM protocol as defined by the DICOM standard [12] and which is built on the TCP/IP network protocol [13]. Due to the large size of most medical datasets, this can place a heavy burden on the network but allows radiologists the flexibility of accessing the data from many locations.

Alternatively, the data may reside on a local file system for a number of reasons. Firstly, in a simple setup a PACS system may not be present as the machine storing the data may be the same one that is used for analysis. Even when a PACS server is in use, it is quite common for medical imaging applications to cache datasets locally in order to avoid the overhead of having to retrieve them across the network the next time they are wanted. Lastly, data may be placed on a portable medium such as CD or USB stick so that it can be transferred beyond the hospital network.

The DICOM file format for locally stored data is described by Section 10 of the standard [12]. It consists of two parts, the header and the image data. The header is a block of text which contains information about the patient (name, date of birth, etc), scanner parameters (modality, pitch, orientation, etc) and image properties (resolution, bytes-per-pixel, etc). Information about the scanner parameters and image properties can be used to make sure the image is displayed properly and to automatically choose the correct protocol, while information such as patient details is typically overlaid on the image so that is is always visible to the clinician.



(a) (b)

Figure 2.4: The numbers on each voxel show the memory offset at which each voxel is stored. A linear layout (a) places voxels, rows, and slices consecutively in memory. The block memory layout (b) groups adjacent voxels in order to improve cache coherency.

The image data itself is separate from the header and for efficiency is stored as a binary representation rather than as text. Data can optionally be compressed using lossy or lossless techniques such as JPEG or Run Length Encoding (RLE) respectively.

2.4.2 Storing Data in Memory

The simplest and most common structure for storing volume data in memory is a linear one. With this approach pixels are stored consecutively to form lines, lines are stored consecutively to form slices, and slices are stored consecutively to form the volume as shown in Figure 2.4(a). The advantage of this approach is that for any given (x, y, z) position in 3D space it is easy to calculate the offset from the start of the data structure to the desired voxel as follows:

$$offset = x + y \times xdim + z \times xdim \times ydim \tag{2.1}$$

where $xdim$ and $ydim$ are the dimensions of the volume in the x and y directions respectively. The disadvantage is that two voxels which are adjacent to each other in 3D space may be separated by a significant amount in physical memory, e.g. two voxels on different slices may be about 1Mb apart. Many operations (such as gradient calculation or trilinear interpolation) operate on adjacent voxels and it is therefore beneficial if adjacent voxels are close by in memory because this significantly increases the cache coherency of the algorithms. It will be noted in Section 5.3.3.1 that the endoscopy renderer also makes use of only a relatively small amount of data per frame and so has the potential to benefit from improved cache coherency.

A block memory layout attempts to resolve these problems (Figure 2.4(b)). The volume is broken up into sub–volumes, or blocks, and the content of each of these are stored linearly. However, the blocks themselves can be stored at arbitrary positions in memory. By carefully choosing the size of the blocks (in our case $32 \times 32 \times 32$ - though Figure 2.4(b) shows $2 \times 2 \times 2$ for simplicity) to fit into cache we can decrease the chance that a desired voxel needs to be fetched from main memory, hence speeding up many operations. We also potentially gain the ability to load larger volumes as one contiguous block of memory is no longer needed.

The main drawback of this approach is that the addressing calculations (especially when crossing a block boundary) are significantly more complicated. Details of the problems and solutions which have been developed are given by Grimm *et al* in [14, 15] along with an evaluation of how effective the block technique actually is.

2.5 Summary

Understanding the nature of the data with which we are working is crucial if we wish to develop algorithms which operate in an efficient and effective way. In this chapter we have introduced the principles behind CT data and have explained how a CT scan is essentially a 3D volume of radiodensity values. Such volumes are typically very large and so care must be taken when storing or transferring them and, most importantly, when loading them into memory. The block based structure we have presented is at the heart of the Biotronics3D system and allows us to achieve good performance with the visualisation we present in Part II.

Chapter 3

Fundamentals of IBD Detection

3.1 Introduction

Given CT data in the form described by Chapter 2, we now wish to apply our system to it in order to assess the probability of IBD being present. This involves following the series of stages which were presented in Figure 1.1, and the purpose of this chapter is to present the background research necessary for an understanding of how these stages can be implemented (or have been implemented in similar systems).

Initially the background research covered the whole of the IBD detection problem domain, from colon segmentation through to the visualisation of the results. Once visualisation was chosen as the main area of study it was necessary to perform more detailed background research in this particular field. Hence this chapter will cover all previous work *except* that relating to visualisation, while Chapters 4 and 5 discuss the previous work on visualisation in more detail.

3.2 Colon Segmentation Techniques

Accurate segmentation of the colon from the CT data set will be crucial to the overall performance of the system because this determines the data which will be worked with. The segmentation of the inner colon wall (the boundary between colon tissue and air) has already been the subject of significant research because it is needed when using virtual colonoscopy to identify colorectal cancer. Extracting the outer colon wall (the boundary between colon tissue and the rest of the body) will require more research on our part, though we already have some insight into how this might be done.

3.2.1 Inner Wall Segmentation

Numerous works on this segmentation task have all taken a similar approach in the stages used in the segmentation process, but have implemented each stage differently. This is

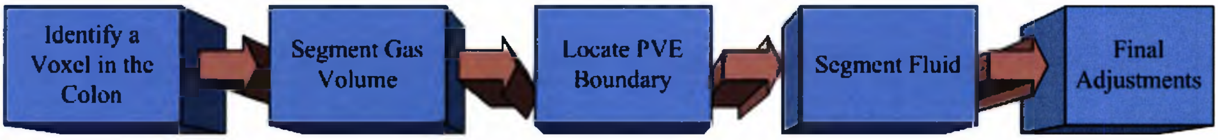


Figure 3.1: The stages involved in the segmentation pipeline.

actually somewhat of an over generalization but will be a useful basis on which to compare the techniques. Figure 3.1 shows these stages and is based mainly on the work of Sato *et al* [16] and Ge *et al [17]* but also partly on the work of Lakare *et al* [18] and Li *et al*[19].

The initial stage identifies a voxel in the colon which can be used as a seed for a region growing algorithm. It is important to ensure that such a seed is indeed in the colon and not in some other gas filled structure such as the stomach, or even outside the body entirely. The following criteria can be used for this purpose:

- Voxel should have a value of less than -800HU to ensure it is gas.
- Voxel should not be connected to corners of the volume in order to eliminate external voxels [20].
- Voxel should be in the lower $\frac{3}{4}$ of the volume to eliminate the stomach [20].
- Voxel should be in an elongated organ [20].
- Voxel could be on the horizontal *Partial Volume Effect* (PVE) boundary between gas and fluid [16].

Note that in the work of [16] the location of the PVE boundary is found much sooner than indicated in Figure 3.1 because this is used to help identify a voxel in the colon. A segmentation of the gas volume is done in most cases using a region growing method but it is also possible to use thresholding to get an approximation. Near the boundary between the tissue and the gas Sato *et al* apply gradient measurements to determine how PVE voxels should be classified [16]. An alternative solution to the segmentation problem is the use of segmentation rays [18] which shoot through the volume and attempt to match the intensity profiles of the rays to predetermined tissue profiles. This is also an effective way of identifying PVEs.

A key issue when segmenting the colon is to also count areas of *Contrast Enhanced Fluid* (CEF) as being internal to the colon and two effective methods have been found for doing this. The method described by Wyatt [20] is based on curvature analysis and takes advantage of the fact that the surface of the fluid will have a relatively flat surface. Such flat surfaces do not occur naturally in the colon and so, after approximating the surface and calculating its normal at adjacent points, areas where the normals point in the same direction are classified as fluid surfaces. The fluid itself lies below this surface and is extracted by region growing.

An alternative approach [16] is focused on detecting the horizontal property of the fluid surface. A $1 \times 1 \times N$ vertical filter is applied at different points in all the columns of the volume and gives maximum response when the upper part is in gas and the lower part is in fluid. This maximum response corresponds to the location of the horizontal gas-fluid boundary. Again the fluid lies underneath this boundary and is segmented by region growing.

3.2.2 Outer Wall Segmentation

While the segmentation of the inner wall has been the subject of significant previous research (due to its importance in virtual colonoscopy) the segmentation of the outer wall has been largely neglected. Finding this outer wall will be important in terms of the larger project and will require significant research resources. However, as described in Section 1.3 we will be focusing on the problem of polyp detection as an application of the work in this thesis and so we can avoid the need to pursue outer wall segmentation.

3.2.3 The Watershed Approach

An approach known as the watershed transform [21] has also been used successfully in many image segmentation problems and has been used in commercial medical products (such as JVision from TIANI MedGraph, though this is no longer on the market). Like region growing, it allows the user to place initial markers to indicate areas which parts of the object are to be segmented, but unlike region growing it also allows markers to designate areas which *do not* belong in the segmented object. The algorithm itself is also significantly more complex than region growing.

It treats the greyscale image as a topographic surface in which white areas have a high altitude and dark areas have low altitude. The aim is to divide the image into catchment basins such that a point belongs to a catchment basin if a drop of water on that point would run down into that catchment basin. The watershed lines of the image are then given by the points between the catchment basins. This is shown in Figure 3.2.

The algorithm works by starting a flooding process at the lowest points on the topographic surface and gradually flooding the surface until it is entirely submerged. When water reaches a point on the surface that point becomes part of the component in which the flooding started. If the water from two separate components meets an imaginary dam is built and this becomes a watershed line.

Rather than actually starting the flooding at the lowest point of the surface the markers can be used as starting points instead. This helps avoid the over segmentation which can otherwise occur as a result of there being many local minima.

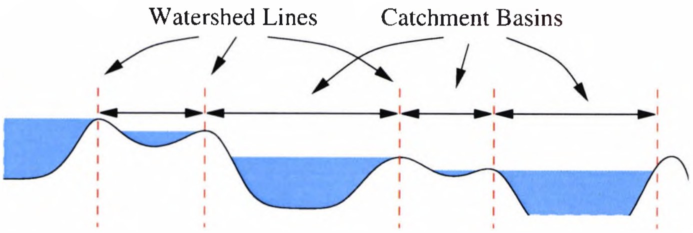


Figure 3.2: The watershed approach treats the image as a topographic surface which gets split into catchment basins separated by watershed lines. The catchment basins correspond to segmented components.

3.3 Texture Analysis

When automatically screening for IBD the concept of texture is expected to be a key property for identifying which areas of tissue are healthy and which are not. Texture is essentially a property of a voxel's local area and includes concepts such as roughness, uniformity, and contrast. It is also important to realise that texture is highly dependent on the scale at which an object is examined, for example from space the earth may appear to have only a few textures (desert, forest, ocean, ice, etc) whereas a closer look at a city will reveal many more (pavement, road, roofs, etc). Texture analysis can be computed using different approaches including statistical, fractal, and geometrical techniques.

A good coverage of many of the available statistical techniques is given by Haralick and Shapiro [22] while a more detailed description of the working of some algorithms can be found in other works. A widely used set of statistical techniques make use of the co–occurrence matrix; this was originally proposed by Haralick [23] but a good explanation (with an example) is given by Gonzalez [24]. Many properties which can be computed from co-occurrence matrices are given by Gonzalez [24] and Tuceryan [25].

Fractals are a means of expressing the degree of self-similarity which a texture exhibits at different scales. A mathematical analysis of their properties was put forward by Mandelbrot [26] and their use in texture analysis is also discussed by Tuceryan [25]. A rough texture is represented by a large fractal dimension, and conversely a fine texture has a small fractal dimension.

Geometrical techniques exploit the property that some texutres are composed of many primitive elements which join together according to a set of predefined rules. Once such elements and rules have been identified they can be used to not only recognize but also to generate corresponding textures. The rules essentially form a grammar over the primitives and specify, for a given sequence of primitives, which primitives may follow it.

3.4 Classification

The completion of the previous stage results in a set of numerical feature values being assigned to each voxel. It is the purpose of the classification stage to determine, based on a set of features, whether the voxel represents healthy or unhealthy tissue. If this cannot be done reliably it indicates that the feature set is not sufficient and a different feature set must be chosen.

A classification procedure will typically begin by identifying the main classes associated with the data set. This will include the class associated with healthy tissue, and potentially a class associated with each area of unhealthy tissue. It remains to be seen whether different areas of unhealthy tissue actually fall into the same class. Once these classes are formed, some algorithm is applied to match each voxel to its most appropriate class; this results in the final set of classifications.

3.4.1 Determination of Classes

The classic approach to class determination is the K-means method [27] and this has been applied in many different types of classification problem. It requires the user to specify a required number of classes which are to be found and aims to identify the class centres (or centroids) which best match the distribution of the data. K-means can be a computationally expensive algorithm but Moore and Pelleg have developed an optimization which makes use of kd-trees [28]. This repeatedly splits the data set in half and builds a tree in which the data points are held in the leaves. This structure essentially speeds up the process of computing the centre of mass for each group of data points as centre of mass information can be cached in the trees nodes and used in successive iterations. Moore and Pelleg claim an improvement of a factor of 170 in some cases.

Another limitation with the K-means algorithm is its requirement for the number of classes to be specified in advance. In a separate paper Moore and Pelleg also address this issue [29] to produce a new algorithm which they call X-means (this also uses the kd-tree optimization from their previous work). Their approach allows the user to specify a range of values for the number of classes (K), they then split some centroids into two children (while keeping K in the desired range) and run K-means locally.

3.4.2 Matching

Having completed the class determination stage explained previously we are left with a set of data points and a set of classes to which they may belong; the remaining task is to classify each point by assigning it to the class to which it most likely belongs.

A simple and commonly used approach to this problem is the minimum distance classifier [24] which simply assigns each point to its closest centroid according to the

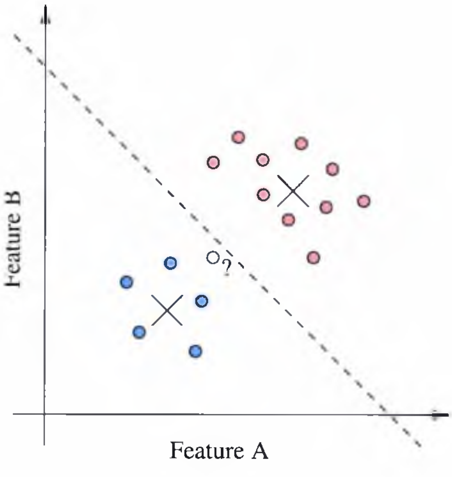


Figure 3.3: A Bayes classifier makes use of prior probability to recognise that a new point is more likely to be red than blue.

Euclidean distance. This is appropriate for clusters in the data set which have a small spread and where the centroids are a significant distance apart. It remains to be seen whether this condition holds true for our data set.

Figure 3.3 shows a situation in which a minimum distance classifier may not work in the desired manner. The data point marked with a question mark would, according to a minimum distance classifier, be classed as blue. However, given that it is so near the boundary and given that so far there are twice as many red data points as blue, it is arguable that it should be classed as red.

Before a data point is examined the probability that it will fit a particular class is called the prior probability (in our example the prior probability that the point will be red is 2/3 and the prior probability that it will be blue is 1/3. A Bayes classifier [24] takes account of the prior probability when performing classification; this allows it to minimise the average error.

3.5 Summary

This chapter has served as a review of existing work in the fields of segmentation, texture analysis, and classification as applied to the virtual colonoscopy problem domain. We began by noting that there has been significant previous research on segmenting the inner wall of the colon (needed when screening for polyps) but little previous research on segmenting the outer wall. Thus research on outer wall segmentation will need to be undertaken if we are to automatically diagnose IBD, but such research is not the focus of this thesis. We also note that texture analysis has the potential to be a powerful approach for diferentiating between healthy and unhealthy tissue, but will need to be coupled with a suitable classification approach.

We have omitted existing work on visualisation in this chapter because it is the focus of our research and so it will receive a more detailed review in the next two chapters. Understanding other stages of the system is important, however, because it can have a direct effect on the way visualisation is performed. For example, if contrast enhanced fluid cannot be successfully segmented than the visualisation must be capable of dealing with this.

Chapter 4

General Purpose Medical Visualisation

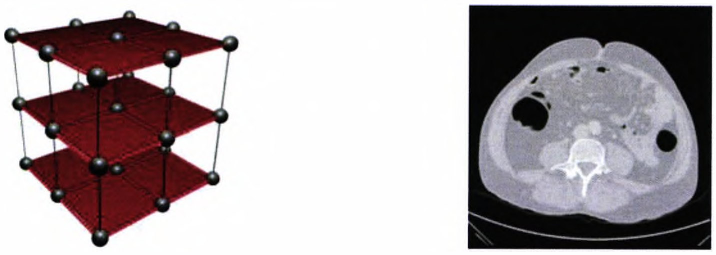
4.1 Introduction

The previous chapter described existing work in the fields of image segmentation, image processing, and image recognition. Undertaking this review of existing work was necessary in order to determine where the focus of the PhD work would be - we chose to focus on visualisation and so the existing work in this field is covered by this chapter and the next one. This chapter looks at medical visualisation in general while the next one examines its application to the specific problem of virtual endoscopy.

Given the nature and the volume of the data described in Chapter 2, it should be evident that it is not easy to examine in an effective and efficient way. To this end, numerous visualisation strategies have been developed which range from simply viewing the CT data as a series of 2D slices to performing complex 3D reconstructions. In this chapter we introduce some of the most common approaches and also describe some of the technical concepts and algorithms which are used to generate them. These same concepts will then be applied later when developing our new visualisation approach.

4.2 Slice Examination

At the most basic level the user simply examines the individual slices which comprise the CT volume (see Figure 4.1). These are mapped to the screen and the user cycles through slices using the scroll wheel or keyboard. Typically the user might be able to zoom in on regions of interest and pan the image in the case that it is too large to fit in the viewport. There is usually also an option to perform *Window-Levelling* - a useful operation which allows the brightness and contrast of the image to be adjusted to maximize the visibility of important structures.



(a) (b)

Figure 4.1: A slice examination displays the original 2D slices which comprise the 3D volume. Compare this to Figure 4.3.

4.2.1 Window Levelling

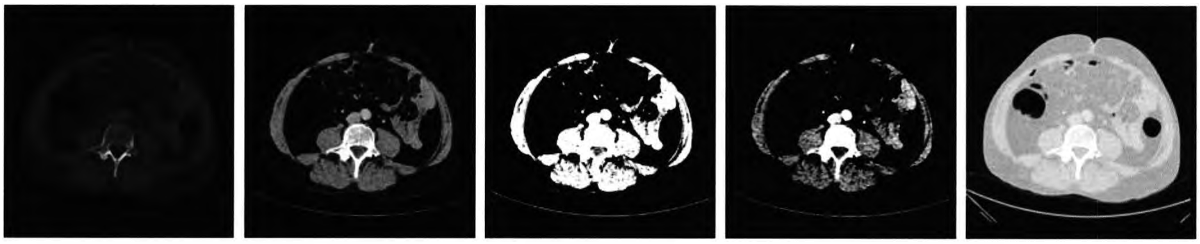
The requirement for window–levelling [30] is bourne out of the fact that 12-bit CT data contains up to 4096 different greyscale levels. This is greater than the number which can be individually discriminated by the human eye, and more than can be accurately displayed by most display devices. This means that mapping the lowest HU value to black and the highest to white causes structures which differ by just a few tens of HUs to appear homogeneous. One solution is to define a *level* centred on the structures HU value and a *window* which then covers a range of values each side. HU values above or below this range are mapped to white and black respectively while those within the range are mapped to shades of grey.

An example may make this clearer. As was seen earlier in Table 2.1 the tissue, fat, and water densities are all concentrated within a relatively small range - with a significant boundary from air and bone. There are several standard window levels which are commonly applied to CT datasets for different applications, the one for tissue uses a level of 40 and a window of 400. This means the total range for HU values which are mapped to greyscale values is $\{-160, 240\}$ and anything outside this range is mapped to black or white. The result of performing this operation is shown in Figure 4.2(b) while images resulting from other window levels are shown in the rest of Figure 4.2.

In general, given an input density, a window, and a level, the output greyscale value can be computed as:

$$greyscale = \begin{cases} 0 & if\ density < lower \\ 1 & if\ density > upper \\ \frac{density - lower}{upper - lower} & otherwise \end{cases} \tag{4.1}$$

where:



(a) w=1800, l=450 (b) w=400, l=40 (c) w=40, l=0 (d) w=90, l=36 (e) w=1600, l=-400

Figure 4.2: The choice of window-level has a significant effect on the image. Window levels are shown here are designed for bone (a), tissue (b), water (c), brain (d) and lung (e).

$$\begin{aligned} lower &= level - \frac{window}{2} \qquad (4.2) \\ upper &= level + \frac{window}{2} \qquad (4.3) \end{aligned}$$

and $greyscale = 1$ represents white and $greyscale = 0$ represents black. This concept of window-levelling is applicable to many visualisations which involve greyscale values, including Multiplaner Reformation and Curved Planar Reformation which are both discussed shortly.

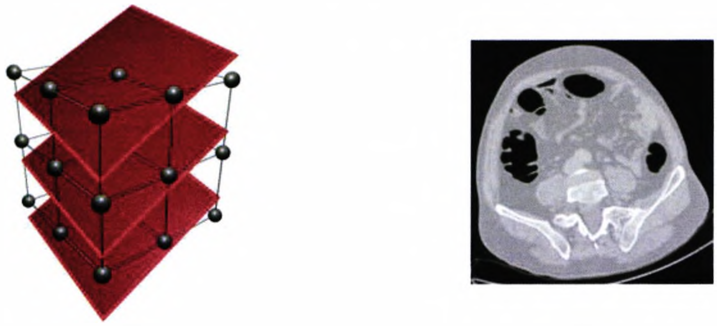
4.3 Multiplaner Reformation

Unfortunately a slice by slice examination has the limitation that interesting structures are unlikely to lie on exactly the plane defined by the slices. In the case of MRI technology it is possible to specify in advance the orientation at which slices should be taken through the body but this possibility does not exist with CT scanners – slices are always taken along the axial plane as shown previously in Figure 2.2.

A solution to this is to provide a visualisation which generates new slices with arbitrary orientation from the original axial slices (see Figure 4.3). This is known as *Multiplaner Reformation* (MPR) [31, 32, 33] and requires defining a plane in 3D space and determining the density at sample positions along the plane. It is important to realise that these samples do not necessarily correspond to the positions of voxels in the original volume, hence some method is needed to determine the density at an arbitrary location based on its neighbouring voxels. This process is known as *interpolation*.

4.3.1 Interpolation

Figure 4.4 shows eight adjacent voxels which form a cube or *cell*. In general we have a sample within the cell which is identified by an $\{x, y, z\}$ offset from the lower left hand



(a) (b)

Figure 4.3: MPR uses interpolation between voxels to allow slices to be displayed at an orientation different to that at which they were scanned.

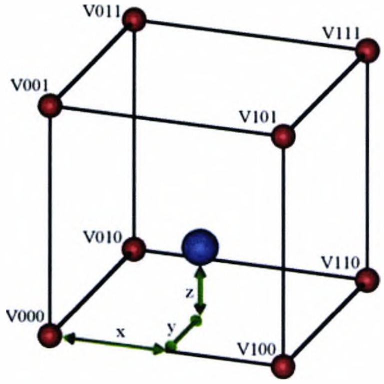


Figure 4.4: Interpolation is the process of determining the value of a sample at an arbitrary position from the value of its neighbouring voxels.

corner of the cell (voxel $V000$). One option to compute the density at the sample is to use the value of the voxel which is closest (called *nearest neighbour* interpolation) but for high image quality a better approach is needed.

Trilinear interpolation is a method of interpolation which weights the influence of each corner voxel by its distance from the sample point. Hence a sample point which is located near a corner voxel will have almost the same value as that voxel, whereas a sample point located in the middle of a cell will have a value halfway between the corner voxel values. The expression to calculate a value at a given sample point is given by Equation 4.4:

$$\begin{aligned} V_{xyz} = {} & V_{000}(1-x)(1-y)(1-z)+ \\ & V_{100}x(1-y)(1-z)+ \\ & V_{010}(1-x)y(1-z)+ \\ & V_{110}xy(1-z)+ \\ & V_{001}(1-x)(1-y)z+ \end{aligned} \tag{4.4}$$

$$
\begin{aligned}
&V_{101}x(1-y)z + \\
&V_{011}(1-x)yz + \\
&V_{111}xyz
\end{aligned}
$$

where x, y, and z are the offsets in the respective directions (see Figure 4.4).

It is worth noting that there are higher order interpolation filters available [34] which use a larger neighbourhood of voxels to give a smoother transition between values and to give a continuous first derivative. However, for most visualisation purposes trilinear interpolation strikes a good balance between image quality and image generation time.

4.4 Volume Rendering

The slice based approaches described previously have been in medical visualisation systems for many years and are widely used. However, they have the limitation that the user only sees a small amount of the information at a time – that which happens to lie on a 2D plane. We can improve this situation by introducing approaches based on *volume rendering* [35] which provide an overview of the whole of the dataset which is being examined.

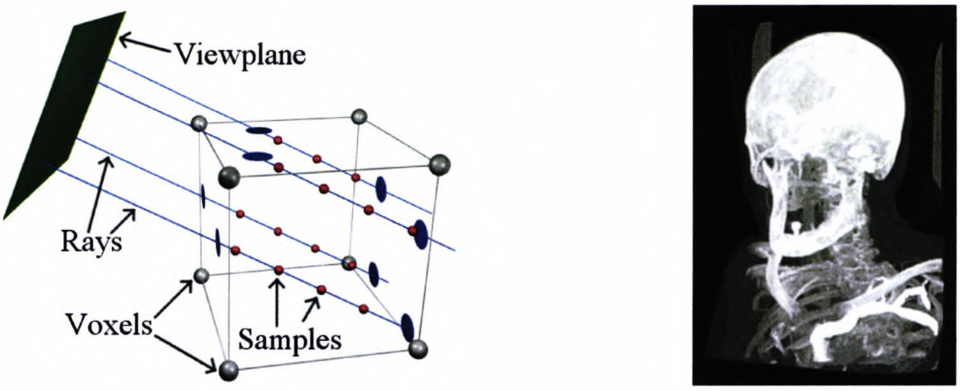
Volume rendering differs from slice based approaches in that, instead of defining a 2D slice through the volume, the user places a camera somewhere in 3D space and views the volume from the outside (or from the inside, in the case of the virtual endoscopy applications described in the next chapter). Volume rendering algorithms are responsible for determining the colour at each point in the final image by considering some or all of the voxels which lie behind it. These voxels can be combined and manipulated in different ways as discussed in the following sections.

4.4.1 Intensity Projection

The concept behind intensity projection is to determine the colour for each pixel in the final image by determining the average (AvgIP) or the maximum (MIP) of all the voxels which lie behind it [36, 37]. This can be done by *raycasting* (as shown in Figure 4.5(a)) as follows:

1. Create a *ray* which starts at the pixel and which travels in the direction in which the camera is pointing.

2. If the camera is outside the volume (as is usually the case for intensity projection) then compute the point where the ray intersects the volume and advance the ray to that point. If the ray does not intersect the volume then this pixel can simply be set to the background colour.



(a) (b)

Figure 4.5: Maximum Intensity Projection (MIP) generates images using raycasting (a). The results are particularly useful for visualising high density structures such as bone(b)

3. Advance the ray by small increments through the volume (the quality of the rendering increases with smaller step sizes but at the expense of greater rendering time).

4. At each point, use interpolation to determine the value at the ray's position; this is known as a *sample*. Interpolation is necessary because the ray position is unlikely to exactly coincide with a voxel position.

5. Combine samples to compute the final pixel intensity. If an AvgIP is being generated then the samples are averaged, whereas if a MIP is being generated the highest sample is used. In both cases window-levelling can be applied to modify the final intensity.

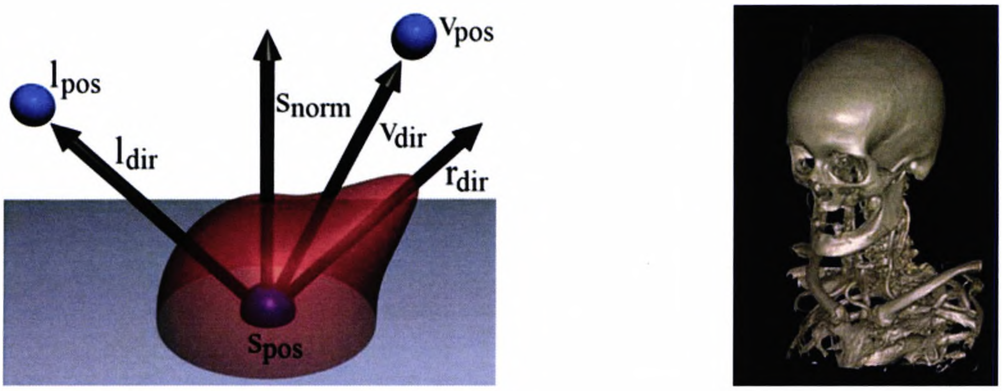
The result is an image like that shown in Figure 4.5(b).

One of the advantages of a MIP is it clearly shows high density structures within the body. Furthermore, it is not possible for a low density structure to occlude a high density structure regardless of how the image is rotated. MIP or AvgIP rendering are not typically used on the colon because it is a low density structure, but MIP is very often applied to the vascular system when a high density contrast agent has been injected.

4.4.2 Shaded Surface Display

A Surface Shaded Display (SSD) is another kind of volume rendering which shows surfaces in the dataset corresponding to a given isovalue which can typically be changed interactively. The images can again be generated by raycasting; the difference being that rays are terminated once the samples go above the isovalue rather than traversing the whole way through the volume.

Once the ray strikes the surface it is necessary to compute the shading at that point in order to determine the final pixel colour. In order for an image to truly look 3D it



(a) (b)

Figure 4.6: The Phong lighting model shown in (a) can be used to compute lighting intensities at each point on the surface. The result is a images like that shown in (b).

is necessary to apply a reasonably realistic lighting model; without this the scene often looks flat. The Phong lighting model [38] is probably the most widely used model in 3D graphics because it strikes a good balance between speed, accuracy, and quality. It is a local model, meaning only direct reflections are taken into account and light will not bounce off multiple surfaces.

Figure 4.6 shows the necessary vectors for Phong lighting where s_{norm} is the surface normal (a vector perpendicular to a given point on the surface), l_{dir} is a vector pointing towards the light source, v_{dir} is a vector pointing towards the eye point, and r_{dir} is the direction light would be reflected off a perfect mirror. The Phong lighting model takes several parameters (each of which is a colour) as inputs:

l_{amb}, s_{amb}: The *ambient* colour of the light and surface respectively. This is independent of light positions or orientations and just serves to brighten the scene.

l_{dif}, s_{dif}: The *diffuse* colour of the light and surface respectively. This is brightest when a surface is oriented to face towards the light source.

l_{spec}, s_{spec}: The *specular* colour of the light and surface respectively. This creates specular highlights to give the impression of a shiny surface.

The ambient (amb), diffuse (dif), and specular ($spec$) components are then calculated as follows:

$$amb = l_{amb} \times s_{amb} \tag{4.5}$$

$$dif = l_{dif} \times s_{dif} \times max(l_{dir} \cdot s_{norm}, 0) \tag{4.6}$$

$$spec = l_{spec} \times s_{spec} \times max(r_{dir} \cdot v_{dir}, 0)^{exp} \tag{4.7}$$

where *exp* is called the *specular exponent* and controls the size of the specular highlight. The final intensity of the light is now given by the sum of the ambient, diffuse, and specular components:

$$intensity = amb + dif + spec \tag{4.8}$$

In some applications of Phong lighting the diffuse and specular components are attenuated according to their distance between the surface and the light source, but in volume rendering there is often no need to do this as distances are relatively short.

4.4.2.1 Gradient Estimation

As can be seen from Equation 4.6 we need a value for s_{norn}, the surface normal. Although not shown explicitly, s_{norm} is also needed for the calculation of r_{dir} which is used in Equation 4.7. Within the domain of volume rendering it is usual to compute the gradient at an arbitrary point by computing the gradients at the corners of a cell and using trilinear interpolation (similar to the way in which arbitrary samples are computed). What is required, then, is a method of determining the gradients at voxels.

One popular solution to this problem is the central difference method [39] which is fast but has relatively low accuracy. For a given voxel, the gradient in the x, y, or z direction is found by computing the difference in voxel values for the nearest voxels in those directions. That is:

$$\begin{align}
xgrad_{(x,y,z)} &= val_{(x-1,y,z)} - val_{(x+1,y,z)} \tag{4.9}\\
ygrad_{(x,y,z)} &= val_{(x,y-1,z)} - val_{(x,y+1,z)} \tag{4.10}\\
zgrad_{(x,y,z)} &= val_{(x,y,z-1)} - val_{(x,y,z+1)} \tag{4.11}
\end{align}$$

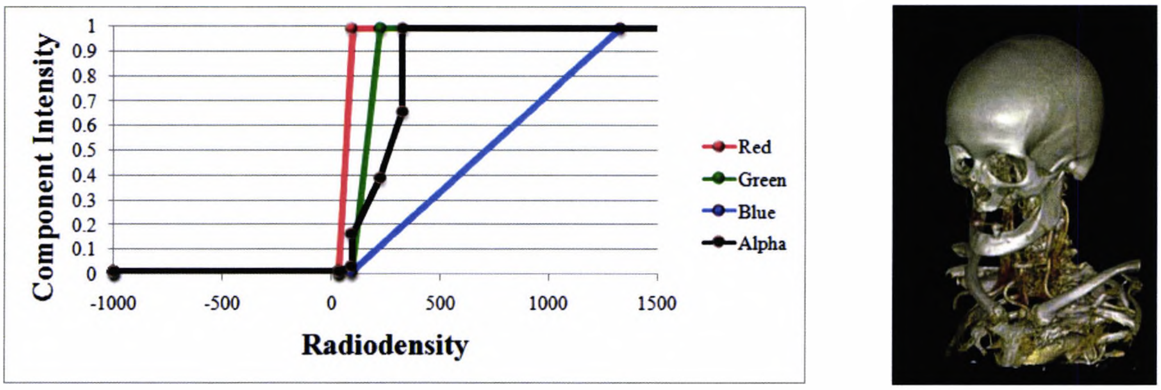
The normal is then given by:

$$N_{(x,y,z)} = \begin{bmatrix} xgrad_{(x,y,z)} \\ ygrad_{(x,y,z)} \\ zgrad_{(x,y,z)} \end{bmatrix} \tag{4.12}$$

There are many other gradient estimation schemes available (see [39, 34] for some good coverage) but only a few are widely used. We will cover others as we use them in specific applications.

4.4.3 Direct Volume Rendering

Direct Volume Rendering (DVR) is one of the most flexible volume visualisation approaches. It can mimic some simpler approaches (such as SSD) but also allows more



(a)

(b)

Figure 4.7: The graph in (a) shows the intensity of the red, green, blue and alpha channels corresponding to given radiodensities. Applying this transfer function to the dataset gives the image in (b).

detailed 3D images of the internal structures of the volume to be presented. As with previous approaches the image generation process can be based on raycasting to obtain and combine samples for each pixel, but the way in which these samples are obtained and combined is much more sophisticated.

Rather than combining raw values, a *Transfer Function* is applied to each sample. This transfer function maps a greyscale value into a colour value consisting of red, green, blue, and alpha (opacity) components. For example, when given a density value corresponding to bone the transfer function might return a colour with high red, green, and blue components (so it appears whitish) and a high alpha component (so it is fully opaque). On the other hand a density value corresponding to tissue might be given a reddish colour and a lower alpha so that it is possible to see what lies beyond it.

Transfer functions are easily implemented using a precomputed look-up table; a one-dimensional array of $\{R, G, B, A\}$ values where the density can be used as an index. For the benefit of the user the transfer function is usually shown on a graph (there are several variations of this) and the user is often allowed to modify this graph (and, in turn, the underlying transfer function) in order to explore the data. Figure 4.7(a) shows a transfer function designed for CT head datasets, while in Figure 4.7(b) the results of applying this can be seen.

Given a number of $\{R, G, B, A\}$ samples taken along a ray it is necessary to perform compositing on them in order to generate the final pixel colour. We keep track of the current ray colour RGB_{ray} and the current ray opacity α_{ray} and, for each successive sample which is obtained in a front-to-back order, we modify them by the colour of the sample RGB_{sample} and opacity of the sample α_{sample} according to the following recursive formula:

$$RGB_{ray} = RGB_{ray^{-1}}.\alpha_{ray^{-1}}(1-\alpha_{sample}) + RGB_{sample}.\alpha_{sample} \quad (4.13)$$

$$\alpha_{ray} = \alpha_{ray^{-1}}(1-\alpha_{sample}) + \alpha_{sample} \quad (4.14)$$

Once the ray opacity reaches a value of 1 there is no need to cast the ray any further because successive samples will not be visible. Cutting short the raycasting according to this criteria is known as *early ray termination* [40].

4.5 Summary

This chapter has introduced several important visualisation concepts including raycasting, surface shading, and transfer functions. Application of these concepts has allowed us to develop visualisations which move away from simple 2D slices and towards highly detailed 3D renderings of medical datasets, as shown in Figure 4.7(b). So far we have illustrated the concepts in the domain of general purpose medical visualisation, but in the next chapter we show how they are applied to the specific problem of visualisation of tubular structures.

Chapter 5

Visualising Tubular Structures

5.1 Introduction

The visualisations described in the previous chapter are for general purpose use and, to an extent, can be used for the purpose of examining structures such as the colon. However, there are other visualisation approaches which are designed specifically for this purpose and which isolate the relevant parts of the data to avoid an overly noisy or confusing image. Such approaches are typically still based on concepts such as raycasting, transfer functions, and shaded surfaces which is why it was useful to review these concepts in the previous chapter.

Before describing these virtual endoscopy visualisations more fully, this chapter actually begins with a more detailed description of *optical* (or *conventional*) endoscopy than has been presented so far. We describe the process and highlight the key advantages and disadvantages of this 'gold standard' so that the virtual approaches can be readily compared. We then present existing work on virtual endoscopy solutions before focusing on the widely used *virtual flythrough*. We finish the chapter with a discussion and an explanation of why a new visualisation is needed.

5.2 Optical Endoscopy

'Endoscopy' is a medical procedure in which a small video camera is inserted into the body to allow the examination of an internal structure. The camera is attached to a long, thin, fibre-optic cable which can be controlled by a handle on one end. This allows the physician performing the examination to maneuver the camera into a position where it can see the structure of interest. The fibre optic cable also serves as a means to transmit light to the camera so the structure can be illuminated.

The procedure is most commonly performed for diagnostic reasons but it is also possible to perform non–invasive surgery, in which case the endoscope will be equipped with

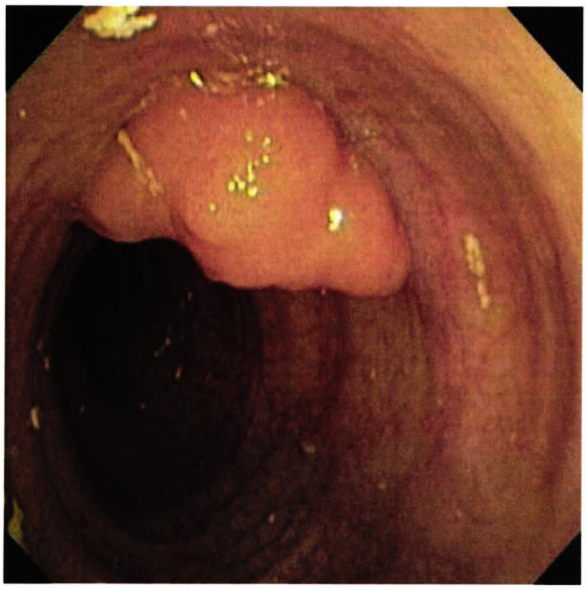


Figure 5.1: An image from an optical colonoscopy (from http://www.gastrolab.net/).

surgical instruments. In many cases it can be inserted through a natural opening in the body, but sometimes a small incision may be made and the endoscope is inserted through that.

As has already been explained, an endoscopy of the colon, or *colonoscopy*, is the 'gold standard' for detecting both polyps and indications of IBD. This usually involves a degree of preparation to ensure the colon is clear of obstructions and patients are typically asked to take laxatives to achieve this effect. Also, the colon is usually inflated with Carbon Dioxide to make exploration as easy as possible. An endoscope in then inserted into the colon through the rectum and follows the colon as far as possible without causing too much patient discomfort. Figure 5.2 shows an example of a polyp in the colon as seen through an endoscope.

As an approach to exploring the colon a colonoscopy has a number of desirable characteristics:

Detailed Images: Because colonoscopy images come from a real camera placed inside the colon they can achieve a high level of image quality. This allows small features such as blood vessels below the surface to be clearly seen; these can be a useful indicator in determining the presence of IBD.

True Colour Images: Unlike the data from CT scans, images obtained through colonoscopy contain true colours. This can be useful because inflamed tissue is often a different colour to healthy tissue.

Biopsy: The ability to take a small sample of tissue for later examination is a useful tool in assessing the presence of IBD or colorectal cancer.

Despite being widely used, colonoscopy suffers from a number of problems [41, 3]. These include:

Preparation: The preparation procedure is unpleasant for the patient and may require an overnight stay in hospital.

Sedation: All procedures involving the use of anesthetics carry the risk of complications. As well as being a risk to the patient, this also means a trained anesthetist must be on hand which is a drain on resources.

Limited Depth: There is a limit on how how far into the colon the endoscope can reach - typically it is unable to examine the small bowel. This can make the diagnosis of conditions in this area problematic.

Surface Coverage: The endoscope has only limited flexibility to examine the reverse side of folds, for example.

Risk of Perforation: One in 500 to 1000 cases results in colonic perforation and one in 2000 to 5000 cases results in death.

After Effects: It is typically a few hours before the patient can resume normal day–to–day activities.

Even if optical colonoscopy is a useful tool for examining the colon, it is not an effective way of visualising the output of our IBD detection system. It is conceivable that in the future it might be possible to overlay such information on the endoscope in real time, but such a system would require extremely sophisticated registration approaches.

Because of these various shortcomings, *virtual* colonoscopy has been used as an alternative for a few years. Virtual colonoscopy involves viewing a reconstruction of the colon built from a CT scan and has successfully been used for the purpose of screening for colon polyps [42, 43, 44]. The remainder of this chapter is dedicated to presenting the existing solutions to virtual colonoscopy, before explaining why they are insufficient and providing motivation for our new approach.

5.3 Virtual Endoscopy

Unlike the general purpose visualisations described in Chapter 4, the majority of dedicated virtual endoscopy visualisations are dependent on additional information which is typically precomputed ahead of time. This precomputation involves *segmenting* the structure of interest (in our case, the colon) and optionally computing some kind of path or centreline through it. Our segmentation is based on the techniques described in Section 3.2 while our centreline generation uses the approach described by Bitter *et al* [45]. In this section we look at the common visualisations which can be applied to the problem of visualising the colon.

5.3.1 Curved Planar Reformation

Curved Planar Reformation is a technique which removes one of the limitations of Multi–Planar Reformation for medical volume exploration by allowing the user to specify *curves* rather than planes and mapping these onto the screen. This curve may also be generated automatically as the result of a previous segmentation step.

The curve is defined by a centreline and a *resampling vector* which specifies the direction in which the line is extended. This centreline is in turn defined by a series of points at sub–voxel resolution. The image generation algorithm works by taking successive centreline points along the centreline (oversampled if necessary) and resampling in the volume along the resampling vector. The way in which this resampling is performed and the way in which the samples are mapped to image space are dependent on the projection method used. Kanitsar *et al.* [46] identify three commonly used projection methods, which preserve spatial perception and isometry to varying degrees:

Projected CPR: Each centreline point maps to a scan line based upon its mathematical projection onto the image plane. There may be many points on the centreline which project to the same scan line and so compositing can optionally be used to combine them. The resampling direction is the same for all centreline points. This method has high spatial perception but does not preserve isometry.

Stretched CPR: Each centreline point maps to a scan line based upon its distance from the start point. This distance is measured *along the curve* (rather than linearly) and this effectively stretches the curve. The process removes any occlusion but at the expense of reducing spatial perception. As with the projected CPR the resampling direction is the same for all centreline points but isometry is preserved.

Straightened CPR: Each centreline point maps to a scan line based upon its distance from the start point. It always maps to the centre of the scan line which has the effect of straightening the centreline in image space. The resampling direction varies between centreline points and is always given by the normal to the centreline. To avoid artifacts due to excessive rotation, it is possible to use techniques described by Klok *et al.* in [47].

CPR is useful in many areas such as visualisation of CT spine data [48] and CT Angiography [49]. In the latter case the centreline is placed through the aorta and a straightened CPR is generated. This yields an image which can be rotated around the centreline to screen the whole of the vessel wall for calcium deposits.

The same approach can be used on other tubular structures such as the colon (see Figure 5.2), though polyps can only be seen when they are intersected by the curved plane. It is typically possible to rotate a CPR image around its centreline so that the

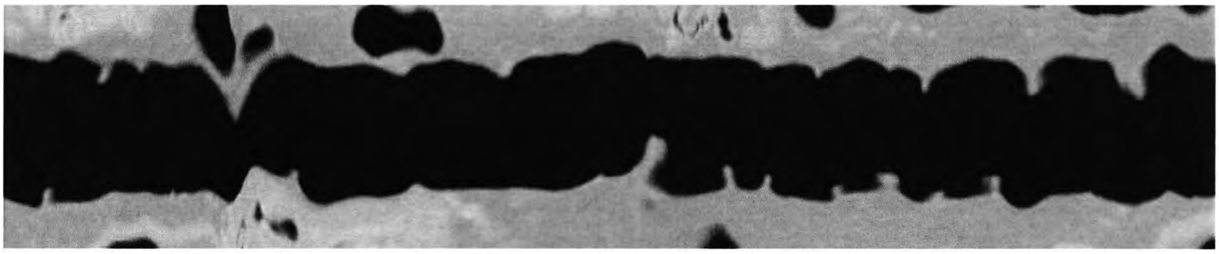


Figure 5.2: Curved Planar Reformation does not show much surface detail when applied to the colon. Compare this with Figures 8.6, 8.7, and 8.9 in Part II.

whole surface can be seen, but this is tedious and it is still difficult to differentiate between polyps and folds in two dimensions. Our new technique (presented in Part II) addresses this issue and makes CPR much more useful for such purposes.

5.3.2 Colon Unfolding

Colon unfolding [50, 51, 52, 53, 54] is a visualisation technique which allows the surface of the colon to be mapped to a single large image. Like the CPR it requires a centreline which defines a path through the colon and which is created during a precomputation stage. These points get mapped to successive scanlines in the final images and individual pixel colours are determined by raycasting.

The algorithm moves along the centreline and, for each centreline point, it builds a local coordinate system including a tangent and a normal vector. For each pixel in the scanline the algorithm rotates the normal around the tangent by some small amount and then casts a ray from the centreline point in the direction of the normal. This ray traverses space until it hits a surface as defined by a threshold value - at this point lighting calculations can be performed as described in Section 4.4.2.

In the work by Vilanova [52], two different forms of unfolding are presented which differ in the way the normal is rotated around the tangent. The first approach is to rotate by a fixed amount between pixels; this means that each scanline is the same length and hence results in a rectangular image which maps easily to the viewport. The second option is to rotate in such a way that the sample distance across the surface of the colon is consistent; this allows the width of the resulting image to vary depending on the width of the colon at that point. An example of an image generated with the second approach is shown in Figure 5.3.

Colon unfolding is considerably more effective at visualising a surface than CPR, and has some advantages compared to the flythrough approach which will be presented in Section 5.3.3. Unlike the flythrough, it allows easy visualisation of areas behind folds and this increases total surface coverage [53, 54]. One of the drawbacks is that it can suffer

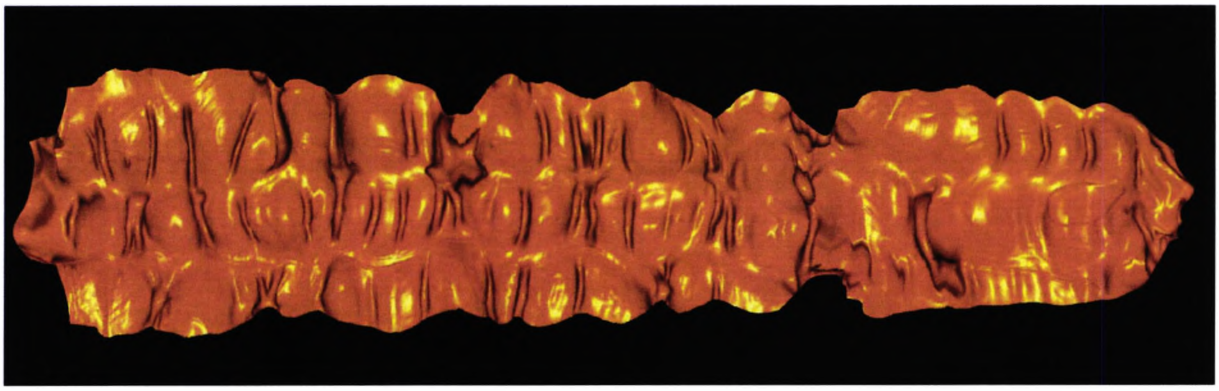


Figure 5.3: Colon unfolding allows easy visualisation of regions behind folds, at the expense of high distortion (image taken from the work by Vilanova [52]).

from relatively high levels of distortion which makes performing measurements difficult and can cause polyps to be missed.

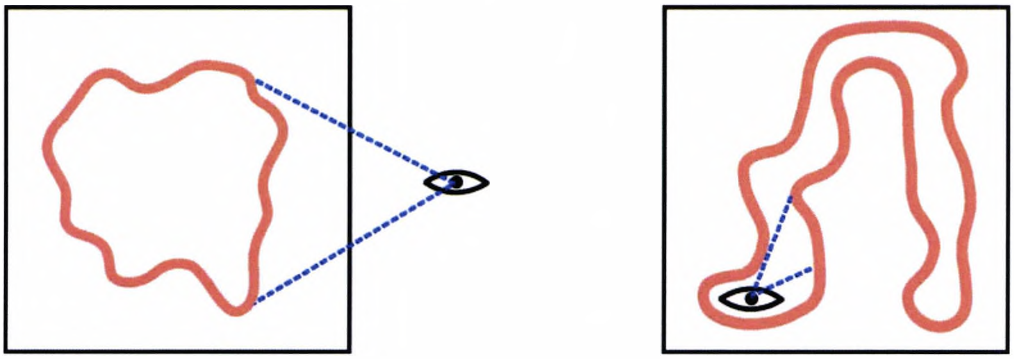
5.3.3 Virtual Flythroughs

Both CPR and colon unfolding provide powerful ways to get an overview of a structure being examined, but virtual flythroughs actually allow the camera to be placed *inside* the structure. The aim is to generate the kind of images which clinicians are accustomed to seeing from regular endoscopes (as shown in Figure 5.3.3.2), but without the discomfort and drawbacks of a real endoscopy being performed. Along with MPR, the virtual flythrough is the most widely used technique for screening for polyps in a CT dataset (and it forms an integral part of our system) so we take the time to explain its operation in detail.

5.3.3.1 The Virtual Flythrough Problem Domain

The image generation process is based on raycasting, as was introduced in Section 4.4.1. However, there are features and characteristics of the endoscopy problem which are unique to it (they do not appear in the general case) and which may help or hinder the process of writing a fast and efficient endoscopy renderer. In this section we give some thought to these features and characteristics.

Occlusion and Transparency: In most medical volume rendering applications the camera is placed outside the volume and at a distance, hence most of the volume is within the cameras field of view (see Figure 5.4(a)). It is also common for there to be a high degree of transparency, for example when viewing a head the skin and muscles may be shown as semi–transparent and the bone as opaque. This is very different from the colonoscopy situation as shown in Figure 5.4(b). For instance, in colonoscopy



(a) (b)

Figure 5.4: In the general case of volume visualisation (a) the majority of the volume can be visible at any given time. In the case of endoscopic rendering (b) there is more occlusion meaning a smaller amount of data has to be taken into account.

the colon wall in almost entirely opaque and the camera can only see a small part of the colon at a time (which in turn is only a small part of the volume). This allows for better cache utilization and hence a higher frame rate.

Simplified Transfer Function: In general a transfer function can map any greyscale value to any {colour, opacity} pair, though in practice the colour and opacity vary smoothly as the greyscale value changes. For the endoscopy problem the transfer function is even simpler because we are looking for an isosurface (or something which is almost an isosurface). The colour is usually constant (an orange–pink colour for the colon) and the opacity is 0 below the isovalue and 1 above it, although there is usually a small ramp around the isovalue to generate a smoother image.

Empty Space to Sample Depth Ratio: Due to the low transparency in the endoscopy problem (as mentioned earlier) rays penetrating the isosurface do not travel far before the contribution from samples ceases to contribute to the image, and so the rays can terminate early. This means the depth to which the rays sample is generally short compared to the distance they travel before they reach the surface. Hence it is worth finding an efficient method to traverse empty space.

Low Data Usage Per Frame: Due to the high occlusion and the position of the camera it is common for only a small amount of the data in the volume to contribute towards the image. It is desirable to use data structures which keep this data available in cache so it can be accessed as fast as possible.

5.3.3.2 Optimizations to the Raycasting Process

We now move on to discuss the optimizations to the general raycasting process which have been developed to enhance the performance of virtual flythrough renderers. Most

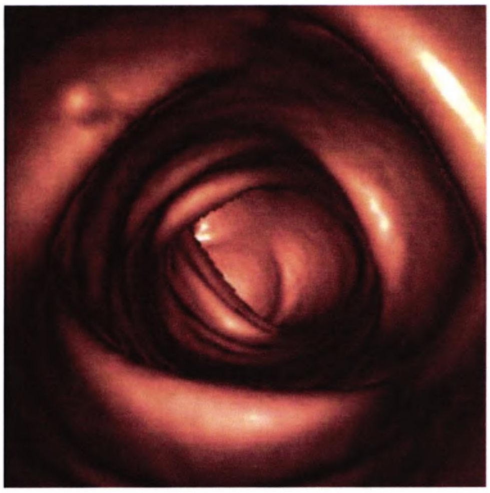


Figure 5.5: A virtual flythrough image of the colon as generated by the Biotronics3D Virtual Colonoscopy system.

of these are designed to take advantage of the properties described previously in Section 5.3.3.1.

Several groups have developed techniques for optimizing the process of rendering a virtual endoscopy. They tend to focus on empty space skipping; that is the process of getting the ray to the colon wall as quickly as possible as this is usually the most time consuming part.

One approach used by Lakare and Kaufman [7] is *distance rays*. These are cast for roughly one in four pixels in what was found to be an optimal pattern. For each ray the distance it traverses is measured and this is stored in the depth buffer. Other ray depths are then interpolated from this and the rays are able to start much nearer the surface. It was proven that this can not miss any feature more than 1 pixel across.

Yagel and Shi [55] propose the exploitation of temporal coherence to efficiently skip empty space. The depth image from each frame is used as a guide for the approximate depth of rays in the next frame. Small camera rotations and translations are handled by transforming the depth image appropriately and filling in any gaps which result. This technique can cause artifacts in the image and so it is not well suited for use on medical datasets.

Another technique used by Sobierajski and Avila [56] makes use of polygon rendering hardware. A polygon approximation of the isosurface is rendered to the screen and its depth values are stored. These can then be used as starting points for the ray casting process. However the task of generating the polygon representation is time consuming and reduces the flexibility of the system.

5.3.3.3 Enhancements

There have been numerous attempts to rectify some of the perceived problems with the virtual flythrough approach. One of the most frequently cited complaints [43, 44] is that the virtual flythrough suffers from limited surface coverage due to a restricted field of view and the presence of folds which can occlude parts of the surface. Several groups have have attempted to address this.

Tiede *et al* [57] make use of spherical panoramas via the QuickTime-VR technology developed by Apple Computer, Inc. For each position on the centreline, an image is rendered in six directions; up, down, left, right, forwards and backwards. This increases the surface coverage of the system because the rear side of folds is seen in the sideways images as the camera passes by and in the reverse image once it has travelled past. During the flythrough the user is able to monitor more that one view at a time in order to increase the amount of surface which is seen. Variations on this concept have been implemented in commercial products such as the Viatronix V3D Colon system [58] which includes a 'rearview' feature.

Some more recent work by Hassouna *et al* [59, 60] takes a different approach by providing a supplementary visualisation which provides a side on view of the colon, rather than modifying the flythrough. In this work the colon is split length-ways into two halves and a camera moves along the centreline and faces directly into the colon wall. They claim that this side on view significantly boosts the surface coverage by allowing the users to see behind folds.

Another of the problems with virtual colonoscopies as compared to the optical approach is the inability to perform *biopsies*. An optical endoscope will commonly be equipped with small surgical implements which can be used to obtain tissue samples; these may then be used for further analysis in order to help determine the most appropriate treatment. It is of course impossible to take real tissue samples from a CT scan but there has been research into the concept of *virtual biopsies* as an alternative [61]. The idea is to allow the user to examine inside any polyp shaped structures by using an alternative transfer function and compositing method. Other concepts such as translucency rendering [62] and profile flags [63] have a similar application.

5.4 Motivation for a New Visualisation

Having reviewed the existing approaches to virtual endoscopy we are now in a position to identify their weakness both generally and also with respect to our particular needs. We can then begin to understand the requirements of a new approach which will eliminate these problems and provide a useful solution to virtual endoscopy visualisation.

The most widely used visualisation for virtual endoscopy is the virtual flythrough as this is present in all systems which provide 3D reconstructions. It is also one of the oldest approaches and as such there has been plenty of time for researchers to identify weaknesses, leading to many papers expressing concern about the *surface coverage* of the approach. Surface coverage is important because any parts of the surface which are not visualised could potentially have polyps which are then not seen. It should be pointed out that it is useful not only to display every part of the surface, but also to ensure that all parts of the surface are visible for *as long as possible*. This increases the chance that any polyps are identified by the user.

The colon unfolding approach improves the surface coverage but instead suffers from relatively high levels of distortion which can have one of two effects. In some cases the curvature of the centreline causes rays to *converge*; meaning polyps can be stretched (causing them to look like folds) or to appear twice. In other cases the curvature of the centreline causes rays to *diverge*; in this case polyps can be missed completely. It would be desirable for a new visualisation to minimize this distortion as much as possible.

All existing 3D approaches have problems when contrast enhanced fluid is present in the colon. This contrast enhanced fluid will form a layer or deposit on the surface of the colon and prevent the visualisation of the area of the surface lying beneath it. This is usually addressed by the use of electronic cleansing routines which automatically identify and remove the contrast enhanced fluid from the dataset. However, these routines must be conservative in their segmentation to avoid accidentally removing structures embedding in the fluid.

Finally, most colon visualisations are designed for the purpose of detecting polyps. While this is our short term goal we must also consider the longer term goal of detecting IBD. This involves visualising the tissue which lies just behind the surface as well as visualising the shape of the surface itself. CPR allows the examination of tissue beyond the surface but provides little information on the surface's shape, and there are extensions to the virtual flythrough to incorporate additional information [62, 64], but a visualisation which combined all of these properties would be highly desirable. Part II of this thesis is concerned with such an approach which we call the Volumetric CPR.

5.5 Summary

This part of the thesis has served to introduce the field of virtual colonoscopy and, in particular, to present a review of the currently available visualisation approaches. We have also explained why existing approaches are not sufficient both for the purpose of detecting polyps and for the purpose of visualising IBD data. This serves as our motivation for why it is useful to develop a new visualisation approach which improves on those which are already available. Based on the literature we have reviewed and the existing systems we have examined, we have been able to determine that our visualisation should exhibit the following characteristics:

Maximise surface coverage: For detecting both IBD and colon cancer it is important to ensure the user sees as much of the surface as possible in order to ensure that any polyps or regions of unhealthy tissue are identified.

Minimise examination time: Reducing the amount of time spent on each examination reduces viewer fatigue and increases throughput.

Enhance polyp visibility: When detecting colon cancer, polyps should be clearly visible against their background.

Eliminate false positives: The technique should allow the user to differentiate between polyps and polyp–like–structures, such as fluid deposits. It should also avoid the same polyp being seen multiple times, as can happen with certain techniques due to distortion.

View surrounding tissue: When detecting IBD it is useful to view the surrounding tissue so the user can determine the extent of any colon wall thickening.

Provide a user friendly interface: User interaction with the system must be straight forward and it must integrate cleanly with existing work flows.

Because we are integrating our solution with the Biotronics3D colonoscopy package we have some additional constraints:

Standard PC: The visualisation must run on a standard Windows PC without excessive hardware requirements.

Limited RAM: We assume only 1GB of RAM to be present and much of this is used up with the dataset. This means our visualisation technique must be memory efficient.

Limited Processor: The approach must be computationally efficient in order to provide the viewer with images at an interactive frame rate.

Operate on slow hospital network: Many hospital networks are already under heavy load with the ever–increasing volume of data which is passed around. However, efficient network usage is something which is handled by the Biotronics3D application and is not of concern from a visualisation perspective.

We now present our new visualisation approach in Part II.

Part II

Volumetric Curved Planar Reformation

Chapter 6

Generation of Volumetric CPR Images

6.1 Introduction

The purpose of the previous part of the thesis was three–fold. Firstly we wanted to provide a review of the existing work related to the project, with an emphasis on the visualisation component. Secondly we wanted to provide a description of the important concepts which are used for generating images in medical visualisation. And lastly we wanted to provide a motivation for why a new visualisation is useful.

With these tasks out of the way, we now move on in this part of the thesis to describe our new approach and how it interacts with the rest of a virtual colonoscopy system. We begin in this chapter by describing the process of generating Volumetric CPR images, which are an extension of CPR. As explained in Section 5.3.1 there are three types of CPR; projected, stretched, and straightened. We extend all three with our volumetric approach, compare their properties, and conclude that the straightened version is the most appropriate for integration into our system. We also discuss how to maximize the surface coverage when using the Volumetric CPR, and present approaches for making sure that raycasting is only used for the relevant parts of the image.

Chapter 7 then describes how the Volumetric CPR was integrated with the existing virtual colonoscopy system and, in particular, with the virtual flythrough. This includes information on the possible layout options, aligning different camera models, and 'picking' (which is the process of determining a 3D position corresponding to a mouse click).

Finally, Chapter 8 builds upon the basic technique with some extensions which can be used to enhance it by presenting additional data in a clear and intuitive way. Specifically, we present a method of combining overlays with the Volumetric CPR, a technique for screening parts of the surface covered by fluid, and new lighting models which help convey the shape of the surface more clearly.

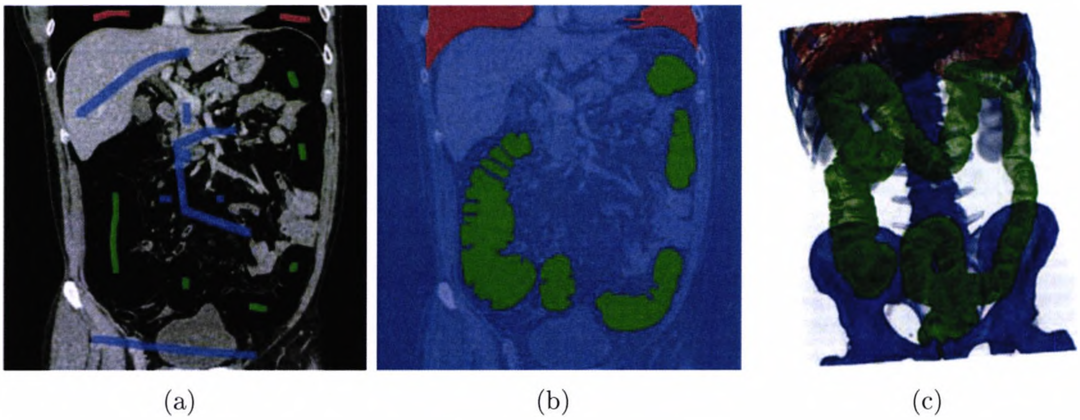

(a) (b) (c)

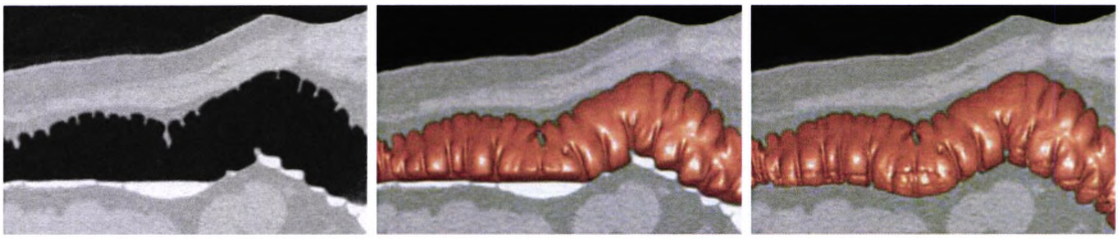
Figure 6.1: In this example, markers have been placed to identify the colon and the lungs, and in order to mark the rest of the body (a). Running the watershed algorithm then results in the segmentation shown in 2D and 3D in (b) and (c) respectively.

6.2 Preparing the Dataset

Like conventional CPR, Volumetric CPR requires a centreline through the structure which is to be examined. Such a centreline could be placed manually but this would be a tedious and error prone task. Instead we generate the centreline automatically from a segmented component of the volume. Although this thesis is focusing on the visualisation aspect of the system, we take the time in this section to briefly outline how the segmentation and centreline generation can be performed. It is relevant because, at the start of the PhD, work was carried out on segmentation which later resulted in a journal paper [65].

Segmentation can be performed using the watershed algorithm, the principles of which were explained in Section 3.2.3. Actually, the watershed algorithm is performed from markers to avoid the problem of over–segmentation which can otherwise affect medical images due to high levels of noise being present. When using this approach the user begins by placing markers inside the structure of interest. Different coloured markers are also placed outside the structure; this is shown in Figure 6.1(a) where the inside of the colon is marked with green markers and the outside is marked with blue markers. Running the watershed algorithm then results in the segmented components shown in Figures 6.1(b) and 6.1(c).

There are numerous ways of implementing the watershed algorithm and Felkel *et al* [66] provide a comparison and computational cost analysis of some common approaches. Our work [65] was based on improving the memory efficiency of Felkel *et al's* algorithms by using the data representation introduced in Section 2.4.2 for the data, label, and cost volumes. Our data structure allows large contiguous regions to be highly compressed, this has relatively little effect on the data volume but significantly reduces the memory required for the label and cost volumes. The effect on the running time is minimal because the data structure allows for efficient random access to the data. For further details of



(a) (b) (c)

Figure 6.2: A CPR of a section of the colon without volume rendering (a). Adding volume rendering (b) produces a much more useful image which reveals a previously invisible polyp. Electronic cleansing can be performed to remove contrast enhanced fluid (c).

the approach see our paper [65] which is included in this thesis as Appendix C.

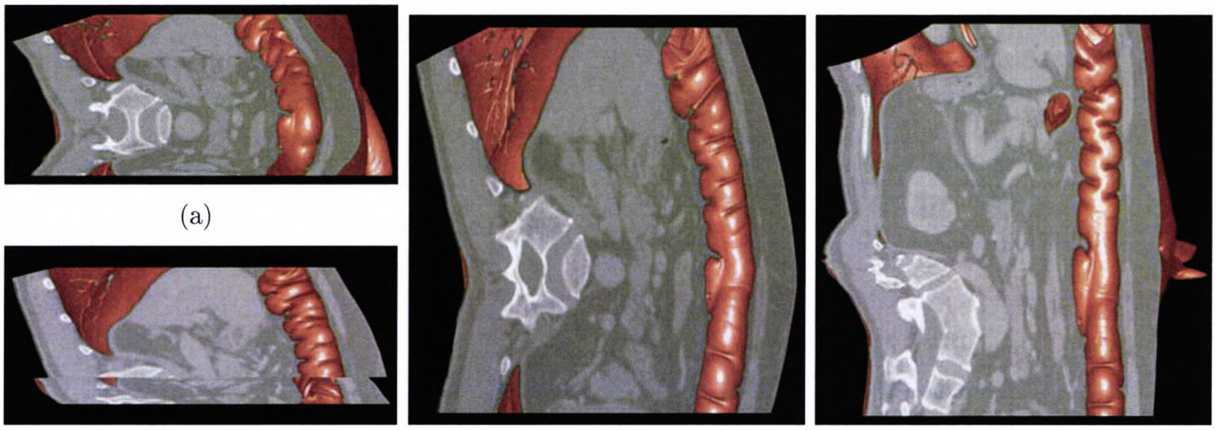
Our approach to centreline computation is based on the work of Bitter *et al* [45]. We begin by choosing a voxel in the colon and computing the most distant voxel from this initial one. This voxel will be at one end of the colon and becomes our start point. We then compute the most distant voxel from this start point which will be the other end of the colon - this becomes our end point. Dijkstra's algorithm [67] can then be used to find the shortest path between the start and the end points. However, the true shortest path will have a tendency to hug the colon wall as it goes round corners. To avoid this, we weight the path cost according to its distance from the colon wall; this results in a much more central centreline.

6.3 Principles of Volumetric CPR

Our proposed new visualisation is a hybrid 2D/3D approach which incorporates Volume Rendering into Curved Planar Reformation. We essentially perform CPR based on a centreline through the colon and then augment the image by performing volume rendering for the dark parts of the image - we call this new technique *Volumetric* CPR. As shown in Figure 6.2 this makes a considerable difference to the information contained in the images.

Although it is relatively straightforward to extend *MPR* with volume rendering it is significantly more difficult to do so for the *CPR* case. As already established there are three different projection methods available for mapping the curve to the image and it is very important to choose a suitable method for the data being examined. We describe the properties of the various projection methods in Section 6.4.

Regardless of the projection method chosen, our prototype implementation uses raycasting [68, 40] for the volume rendering, mainly due to its greater flexibility compared to other volume rendering algorithms and the fact that the projection is not necessarily orthographic or perspective for all CPR types. Specifically, it is an isosurface raycaster



(a)

(b)

(c)

(d)

Figure 6.3: The same piece of colon is shown above using different projection methods. Projection is used for (a) and (b); in the case of (b) the camera has been rotated to demonstrate how occlusion can occur. Occlusion cannot occur in the stretched or straightened cases shown in (c) and (d) respectively.

and so it is possible to take advantage of some of the existing techniques for space leaping [7, 8] and accurately finding the isosurface [69, 70].

The algorithm begins as a conventional CPR by identifying the first centreline point on the centreline and then resampling along the resampling vector. It maps each sample to a pixel and determines, according to some criterion, whether the pixel should get its colour from the sample which was just taken or through raycasting of the volume. If it decides on the former then the sampled value is modified by the window–levelling function and drawn to the image. If it decides on the latter then a ray is set up in the appropriate direction and ray–casting is performed before drawing the result. Hence there are three main questions which must be answered during the rendering process:

1. Which pixels should have their colour determined by the window–leveling function and which by raycasting?
2. In which direction are rays cast?
3. What is the criteria for terminating a ray?

A simple criterion for determining whether or not a pixel should be raycast is its resampled intensity value, as can be used when enhancing MPR in a similar way. If the intensity value is high then the window–levelling function is applied to give the final colour while if it is low then raycasting is used instead. This is a simple yet effective approach which is fast to check and results in images like that shown in Figure 6.2(b). An even better criterion which makes use of region growing is presented later in Section 6.6.

The direction of the rays is dependent upon the projection method being used. For the projected Volumetric CPR the ray direction is computed from the relative positions of the object and view plane, whereas for the stretched and straightened CPR it is computed from the shape of the curve. This is discussed in more detail in Section 6.4.

The rays are terminated once the resampled intensity value goes above a certain threshold, indicating that it is leaving the air–filled region. This usually means the ray has hit the surrounding tissue but in the case of virtual colonoscopy datasets it is common for a contrast enhancer such as Barium to also be present - this marks the stool so it can be easily recognized during the examination. Such contrast–enhanced fluid typically shows up as bright white in standard CPR rendering (Figure 6.2(a)) and, because it has a high density, it will cause ray termination and show up as flat surfaces in the volume rendered part of Volumetric CPR (Figure 6.2(b)) or during virtual flythroughs. We discuss this further in Section 8.3, but for the time being it is possible to perform electronic cleansing to remove the fluid and obtain images like that shown in Figure 6.2(c).

6.4 Comparison of Projection Methods

As with a conventional CPR, the choice of projection method makes a significant difference to the resulting image [46]. In fact the difference is even greater with the raycasting enhancements because the projection method has an effect on the direction in which the rays are cast; see Figure 6.3 for a comparison of the resulting images. In this section we look at the implications of using each of the projection methods.

6.4.1 Projected Mode

A projected Volumetric CPR takes each point along the centreline and projects it into image space. The values along the projected scan line are then computed by resampling into the volume along the resampling vector. If the curve folds back on itself then more than one point on the centreline will map to the same scan line. When this occurs a policy is required to determine what should be displayed; within our implementation the old value is simply overwritten which can cause occlusion to occur in the image. An alternative is to use some form of compositing but this usually results in confusing images which are hard to understand.

When operating in projected mode, our Volumetric CPR casts rays from points on the curve corresponding to empty voxels in the direction of the normal to the viewport. That is, all rays which are cast are parallel; this results in an orthographic 3D rendering. This is illustrated for the general case by Figure 6.4.1, which also shows the area the curve is projected onto and the places in which occlusion occurs.

A real example is illustrated by Figure 6.4.1 which shows the process of rendering a curved segment of a tubular structure. In this particular case the view plane is positioned such that no occlusion occurs. Figure 6.3(a) shows the result of rendering a colon segment with this setup.

In the middle of Figure 6.3(a) the rendering is undistorted and it is easily possible to identify the folds of the colon. However, toward the top and bottom of the image a form of distortion occurs. Referring to Figure 6.4.1 reveals why this is the case; it can be seen that near the centre of the image rays travel from the centreline to the *nearest* part of the surface whereas near the top and bottom of the image the rays travel a much greater distance to a part of the surface which is further away. This has the effect that near the top and bottom of the image the 3D rendering no longer corresponds as closely to the surrounding CPR which provides the context. For this reason, projected Volumetric CPR is most applicable to short lengths of tubular structures or those which have only a low degree of curvature.

A further issue occurs when we rotate the viewport or the curve to examine a different part as shown in Figure 6.4.1. Note that in both Figures 6.4.1 and 6.4.1 there is a point marked A from which rays are cast into the surface. This point is the same in both figures but the rays which are cast reach a different point on the surface. As we rotate the curve, the 3D rendered position corresponding to a given CPR position can change. Although this can look strange it is not disorientating because there is only a small change between each frame.

Figure 6.4.1 also illustrates how occlusion can occur which, as previously stated, is as a result of multiple points on the centreline mapping to the same scan line. The easiest approach is simply to overwrite the previous contents of the scan line with the new values; this results in images like that shown in Figure 6.3(b). It could be argued that this is actually the correct behavior because a part of the curve near the view plane is occluding a part of the curve which is further away but unfortunately this behavior is not guaranteed. If we had performed the Volumetric CPR generation by starting at the other end of the centreline (marked 'End' on Figure 6.4.1 rather than 'Start') then the far part of the curve would have been rendered over the near part of the curve. For some curve configurations there is no start/end pair which gives the correct ordering though a depth buffer can help resolve this.

6.4.2 Stretched Mode

The stretched mode builds and improves on the projected mode by eliminating the occlusion and some of the distortion which occurs. The occlusion is addressed by ensuring that no two points on the centreline map to the same scan line in image space. As with the projected Volumetric CPR, the algorithm takes successive centreline points along the centreline and maps them to image space, before resampling into the volume along the resampling vector to give values along the scan line. However, rather than simply projecting the centreline point, its distance from the start point along the centreline is used

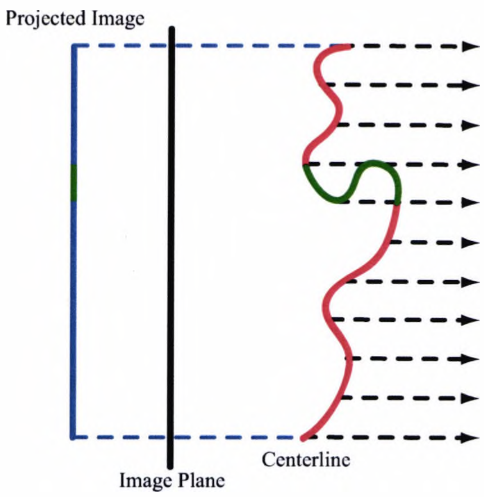


Figure 6.4: Ray pattern for the Volumetric CPR. The area shown in green is where occlusion occurs.

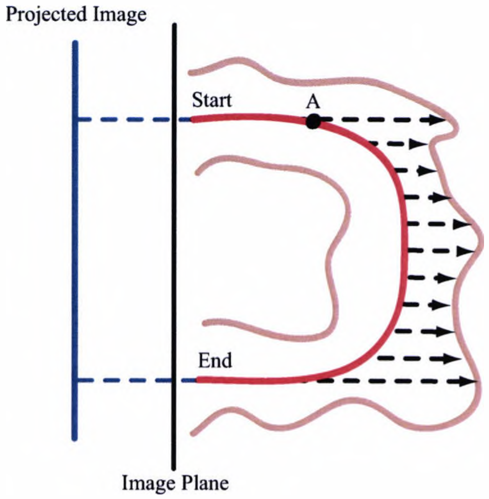


Figure 6.5: Rendering a projected Volumetric CPR of a tubular structure. This ray pattern gives an image like Figure 6.3(a) when applied to the colon.

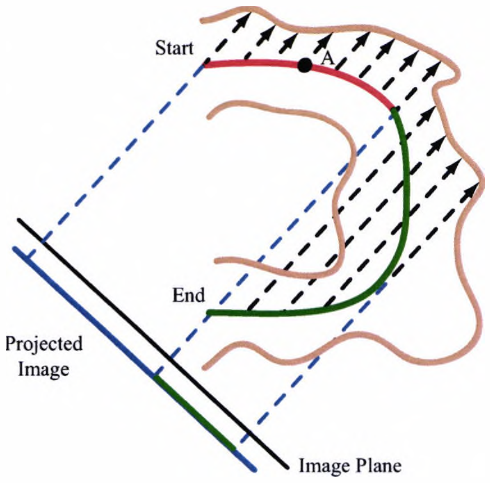


Figure 6.6: The same arrangement as Figure 6.4.1 but with rotation. This causes occlusion to occur on the green part of the image plane as the whole of the green part of the curve maps to this region.

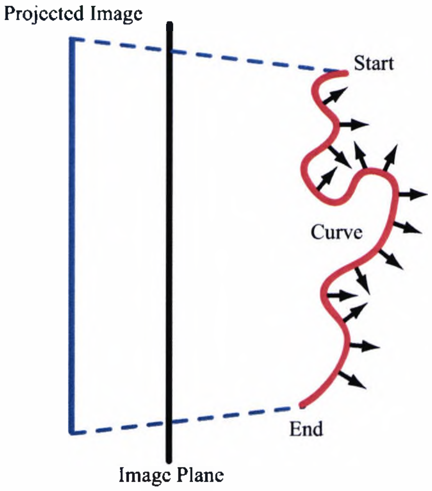


Figure 6.7: In a stretched Volumetric CPR rays are no longer sent in a parallel formation.

as an offset from the bottom of the resulting image. No two points on the centreline have the same distance from the start and hence no occlusion can occur.

The stretched Volumetric CPR algorithm is further modified from the projected version such that the rays are no longer always sent parallel to the view direction. Instead the direction is modified according to the curvature at the centreline point. This is illustrated by Figure 6.4.2. Note that the rays are still parallel *within any given scan line* but are not necessarily so between scan lines.

Modifying the ray direction in this manner leads to some reduction in distortion compared to the parallel ray approach. The issue which was illustrated earlier by Figures 6.4.1 and 6.4.1 (namely that the point on the wall which corresponds to a point on the curve varies as the view plane is moved) does not occur because the ray direction is no longer

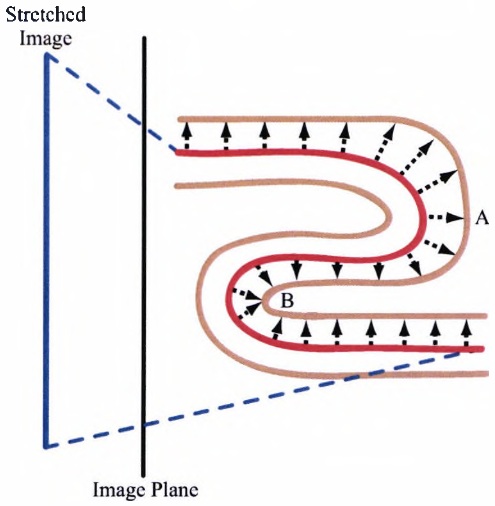


Figure 6.8: Ray spacing along the walls varies according to curvature.

dependent on the view plane or curve position.

However, a new artifact can present itself as shown by Figure 6.4.2. The centreline near point A is exhibiting a high degree of curvature which causes the rays to diverge; this in turn causes the spacing of the sample points along the wall to increase. Near point B the opposite occurs. In both cases the perceived effect is a form of distortion, near point A parts of the wall get compressed whereas near point B parts of the wall are stretched (or may even appear twice).

We are certainly not the first to have experienced these difficulties when mapping a curve to a plane and fortunately there are some known approaches which can help reduce the problem, or at least trade off different forms of distortion. During colon flattening Vilanova *et al.* *[54, 52]* made use of an adaptive sampling approach in order to ensure that samples are taken at equidistant intervals along the colon wall. A similar approach is applicable here but it should be noted that varying the frequency of samples along the centreline will mean that a given distance in image space will no longer correspond to a fixed distance in the CPR; whether this is a beneficial trade off will depend on the application.

A second approach is to curve the rays as they traverse space to perform a kind of non–linear raycasting [71, 53]. This can be used to prevent the rays intersecting but it requires some extra computational expense and a precomputation stage [72]. A real example of the stretched Volumetric CPR in action is given in Figure 6.3(c) and the ray pattern is shown in Figure 6.4.2.

There are some similarities between the raycast images generated by our stretched Volumetric CPR and those generated by colon flattening without the ray curving modification. Both techniques make use of the centreline as an origin for the rays but there are differences in the way they are cast.

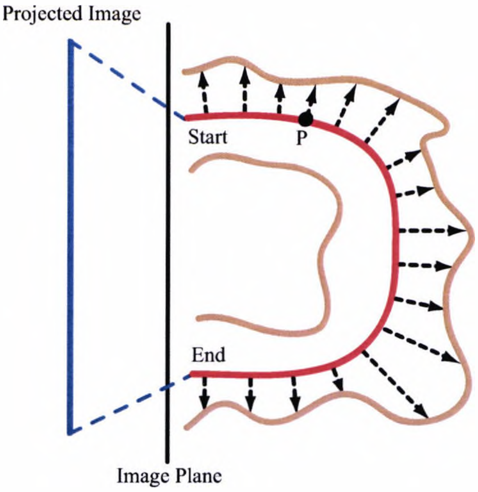


Figure 6.9: Rendering a stretched Volumetric CPR of a tubular structure. This ray pattern gives an image like Figure 6.3(c) when applied to the colon.

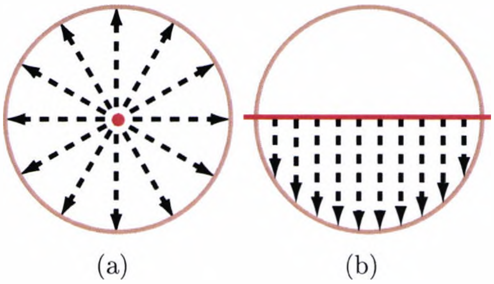


Figure 6.10: During colon flattening the rays radiate from the centreline (a) whereas during stretched Volumetric CPR they are all normal to the curve (b).

During colon flattening rays emanate from the centreline in all directions in a way which either maintains a constant angle between samples or which maintains a constant sampling distance along the wall of the colon. This is shown in Figure 6.10(a).

By contrast, the Volumetric CPR takes the centreline and generates the curve which corresponds to it. Rays are then cast from this curve as shown in Figure 6.10(b); this means rays are parallel within a scan line but not so between scan lines.

6.4.3 Straightened Mode

The stretched mode described previously allows an arbitrary centreline to be defined in 3D space. However, the curve is then formed from the centreline by extending it along the resampling vector; this vector is the same for all points and so the curve is always extended in the same direction (see Figure 6.11(a)).

For the straightened CPR we drop the restriction that the resampling vector is the same for all points on the centreline. Instead the resampling vector is now specified by the normal to the centreline at any given point and this results in a curve which twists

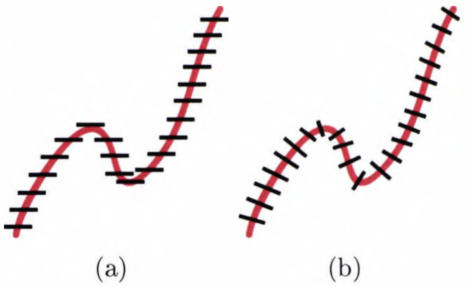

(a) (b)

Figure 6.11: For a stretched Volumetric CPR all resampling vectors are in the same direction (a). For the straightened Volumetric CPR they follow the normal to the centreline (b).

and turns in the full three dimensions as shown in Figure 6.11(b).

For each point on the curve there are actually an infinite number of normals which could be used and the choice of normal has a significant effect on the resulting image. A simple approach is to compute the normal using the Serret–Frenet equations but this has the disadvantage that twists can appear when the curve changes direction. Instead we choose to compute the normal n from the tangent t to the curve and an external reference vector r such that:

$$n = t \times r \tag{6.1}$$

Because centreline points are mapped to the centre of each scan line the centreline is straightened in image space. As with the conventional straightened CPR this has the benefit of making the images significantly easier to understand and to interact with as rotation in the projected and stretched mode can be non–intuitive.

A straightened Volumetric CPR of the same section of colon that was shown in Figures 6.3(a) to 6.3(c) is shown in Figure 6.3(d). A conventional straightened CPR suffers from some distortion at points which are not near to the centreline and this continues to be the case for the straightened Volumetric CPR. This is caused by the resampling vectors converging or diverging as they move away from the centreline.

In practice we have found the straightened version of the Volumetric CPR to be by far the most practical for screening the colon. One of the reasons for this is that the result is always a long, thin, image whose shape doesn't change as the user interacts with it. This means it can be easily displayed along the bottom of the screen, for example. Secondly the absence of occlusion (as is present in the projected version) means that arbitrarily long structures can be examined. Although some disorientation can occur in theory we have not actually seen any examples of it.

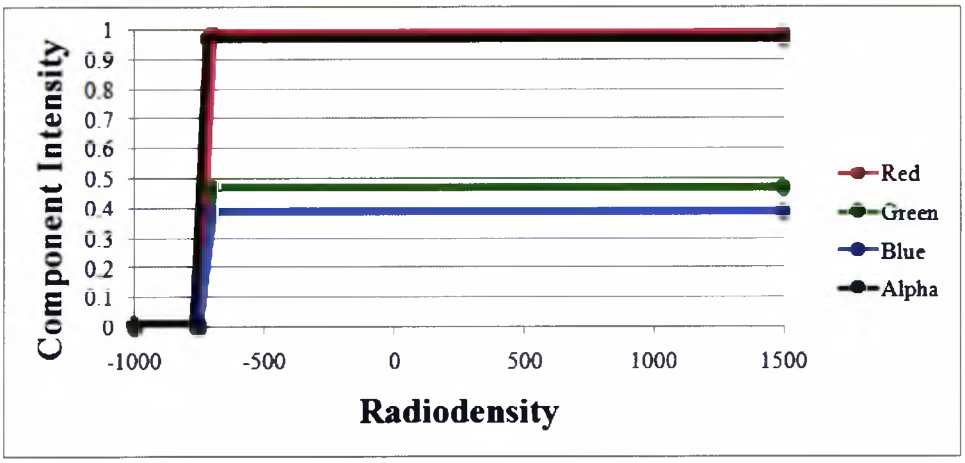


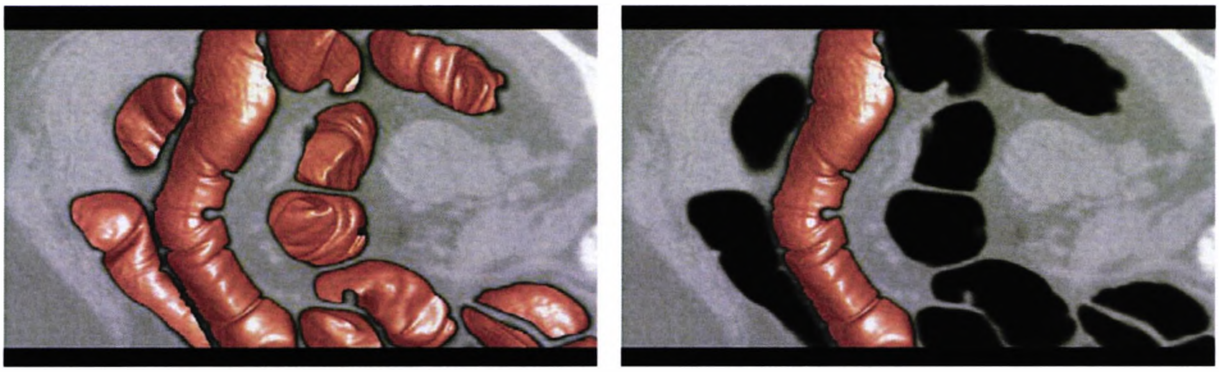
Figure 6.12: A transfer function defining an isosurface at -750HU and displaying it with a pink colour as shown in Figure 6.3.

6.5 Transfer Function Specification

As was discussed in Section 4.4.3, a *transfer function* is used to specify the relationship between radiodensity values of the voxels in the volume and the final colour in which such voxels should be rendered. However, the transfer function for the Volumetric CPR is much simpler than the one presented in Section 4.4.3 because we are essentially rendering an *isosurface*.

An isosurface is a surface in the volume consisting of all the points corresponding to a given isovalue. For example, within virtual colonoscopy datasets the isovalue of -750HU defines the isosurface which separates air in the colon from the surrounding tissue. Anything with a density lower than -750HU is considered to be air, and anything above is considered to be tissue. This is the value which the raycasting algorithm uses when taking samples to decide whether it has reached the surface. Figure 6.5 shows the transfer function which we have just described.

It can be seen from Figure 6.5 that the there is actually a slope in the graph, rather than the components instantly jumping from a intensity of zero up to their final values. This slight slope helps to improve the quality of the resulting image, and also ensures the Volumetric CPR exactly matches the appearance of the flythrough, as this uses the same transfer function. It should be noted that variations on the transfer function are possible; Section 8.2.2 presents a variation which allows the user to see just beyond the colon surface.



(a) (b)

Figure 6.13: Region–growing can be used to restrict the number of pixels for which direct volume rendering is performed. Without region growing (a) areas not of immediate interest are still augmented with direct volume rendering. Using region growing rectifies this (b).

6.6 Restricting the Raycast Area

The algorithm described in the previous sections gives some pleasing results but further refinements are proposed. Figure 6.13(a) shows that some of the areas in which raycasting has been used do not actually correspond to areas of interest, and there are two reasons why it is not desirable to perform raycasting for these areas. Firstly they can be a distraction; the clinicians attention should be focused on the area lying along the section of centreline section currently being visualised, rather than a section of the centreline which they may have already seen. Secondly, the raycasting of those areas can have a significant performance impact. This is especially true for areas which are outside the body as rays may travel a long distance before hitting a surface or leaving the volume.

What is needed is an importance–driven criterion for deciding which pixels should be raycast. There are several candidates; we describe two simple approaches and a more complex one which has proved very successful. During the Volumetric CPR generation the volume is resampled along the resampling vector starting from the current centreline point. It is trivial to limit the distance to which this resampling occurs and this results in an image which is clipped a short distance from the centreline. This clipping distance needs to be chosen carefully to ensure it is large enough to always cover the structure of interest and small enough to minimize other air–filled parts of the volume but in practice this approach works fairly well for the straightened CPR. For other CPR types it is less effective due to the curved nature of the centreline in image space.

A second simple solution is a scan line based connectivity approach. For each scan line a flag is kept indicating whether raycasting can currently be performed. This flag is initially true when rendering of the scan line is started and is set to false when the samples go above a certain threshold. This occurs when the scan line leaves the structure of interest and then no further raycasting is performed for that scan line. Again this

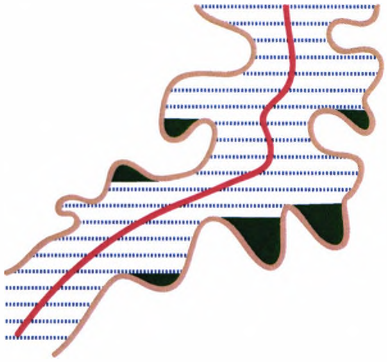


Figure 6.14: The scan line approach to clipping fails under many circumstances. Areas in green should be raycast but are not.

technique can be reasonable when applied to a straightened CPR with a centreline which is vertical in image space but fails with other types (see Figure 6.6).

Our solution is instead to only perform raycasting for those pixels which are connected to the centreline in image space. This is determined by a region growing approach using the pixel intensities as the criterion and the centreline's image space position as a set of seed points. This prevents raycasting of pixels outside the body and in other air–filled structures and also avoids the issue illustrated in Figure 6.6.

We have implemented the rendering using a two–pass approach. During the first pass the CPR is rendered as normal using any of the available projection methods but each centreline also gets projected to image space for use as a seed in the second pass. During the second pass, region–growing [73, 74] is performed from the seed points to determine the list of pixels which are connected to the centreline in image space. Raycasting is then used to determine the colour for each of the pixels in the list and the result is written over the previous value.

Breaking the rendering into two passes introduces some additional complications. A CPR image is usually an irregular shape and yet it is drawn to a rectangular viewport. Some viewport pixels therefore fall outside the data volume and it is conventional to draw these black; this causes problems during the second pass as the region–growing algorithm cannot tell which black pixels are air–filled structures and which simply fell outside the image. To avoid this situation the first pass sets a flag for each pixel of the viewport indicating whether it is inside the image. This can then be checked during the second pass. Figure 6.6 shows the different categories of pixel which result from this process.

The complete algorithm generates images like those shown in Figure 6.13(b) as compared to Figure 6.13(a) which does not use the new region–growing approach.

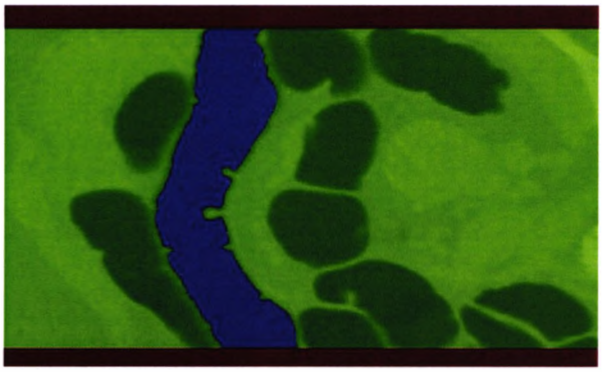


Figure 6.15: Pixels are tinted blue *if they were originally black, are inside the image, and are connected to the centreline; these pixels should be raycast. Pixels are tinted* red *if they were originally black but fell outside the image. The remainder of the pixels are tinted* green *and either were not originally black or were originally black but not connected to the centreline.*

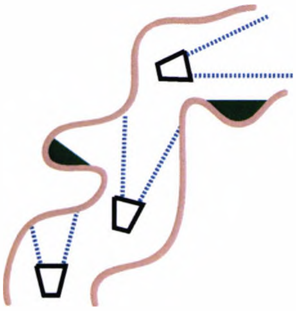


Figure 6.16: The virtual fly–through approach can miss the areas marked in green.

6.7 Maximizing surface coverage

An important aim for virtual endoscopic techniques is to maximize the surface area which is seen during the visualisation process. This increases the chance of the user identifying any features which are characteristic of the condition being diagnosed.

Virtual flythroughs, the most commonly used approach, do not guarantee that the whole surface is seen as it is possible for areas to be occluded e.g. behind folds of the colon wall as shown in Figure 6.7. Solutions to this problem include providing reverse or multiple views (which requires the user to watch two displays at once), or marking seen areas during the flythrough and allowing the user to go back to any areas which were missed.

The Volumetric CPR can also have problems with surface coverage; the most obvious of which is that only half the surface is seen at a time. This is because the curve essentially cuts the tubular structure in half and cast rays in only one direction (see Figure 6.17(a)). A simple solution would be to cast rays in both directions and show the results as two separate images (Figure 6.17(b)). This is a good approach but can fail in some situations

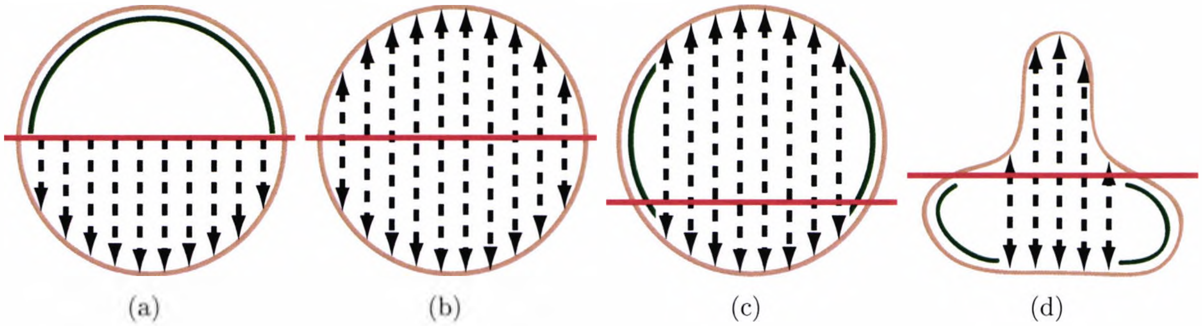


Figure 6.17: In Volumetric CPR only half the surface of the tubular structure is covered (a) as the areas marked in green are missed. This can be rectified by rendering two images with the rays going in opposite directions (b). However, this solution can fail if the centreline is not accurate (c) or if the cross-section is a more complex shape (d).

such as when the centreline is not perfectly 'central' or when the tube has a complex cross–sectional shape. These situations are shown in Figures 6.17(c) and 6.17(d) respectively, though we have not found that they happen in practice.

Another approach is to simply rotate the curve around the centreline, as with CT Angiography; either presenting this to the user as a video or allowing the user to control it interactively. It can be seen that in the case of Figure 6.17(d) this will allow all the surface area to be seen. This is true for all cross–sectional shapes which commonly occur during endoscopy.

6.8 Summary

This chapter has described the process of generating images for our new visualisation approach - the Volumetric CPR. By extending conventional CPR with raycasting we have obtained detailed colour images of the colon surface while also displaying the surrounding tissue. We have looked at the advantages of the different projection methods – all of them have been implemented but only the straightened version is exposed in the Biotronics3D system because it is the easiest to interact with from the users perspective. We have also presented the idea of region growing in image space as a method of reducing the amount of raycasting which is performed, and hence speeding up the image generation process. The work presented in this chapter has been accepted for publication in the IEEE Transactions on Visualization and Computer Graphics and the preprint of this paper in included in Appendix A.

Chapter 7

Integration with a virtual colonoscopy system

7.1 Introduction

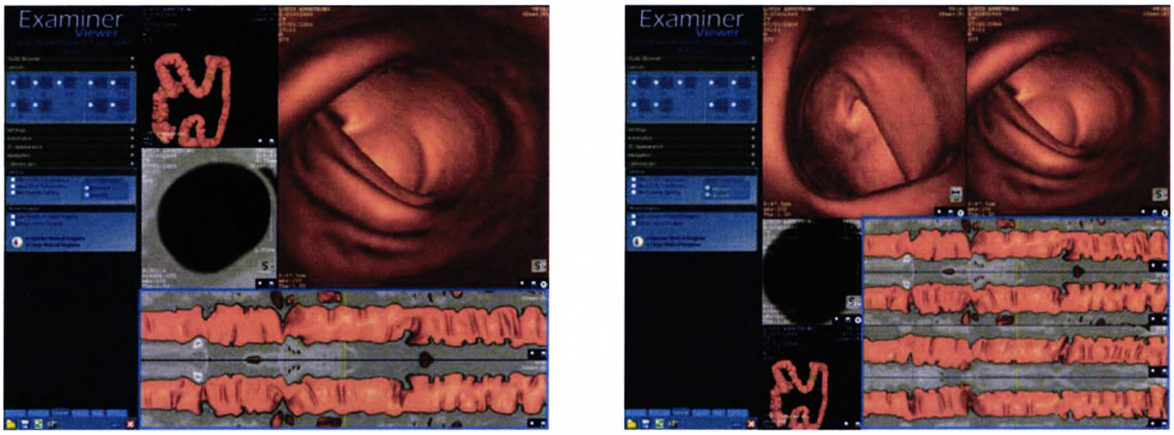
The Volumetric CPR alone has the potential to be a powerful and flexible tool for the screening of the colon wall and surrounding tissue. However, there are several reasons why it is useful to instead integrate the technique into a more complete solution in which it is used alongside other visualisation approaches. For example:

Access to 'base' data: While sophisticated 3D reconstructions of the data are useful for improving spatial awareness and providing an overview of a data set, many users also wish to see the 'base' data - that is, the original slices which comprise a volume. On discovering a polyp in a Volumetric CPR or flythrough they like to confirm the diagnosis in this way.

User familiarity: Because the Volumetric CPR is a new technique users are unlikely to immediately migrate to it until they have established its advantages for themselves. Many users are happy with the virtual flythrough [42, 43, 44] or were originally trained to use base data and so still feel most comfortable with this.

Measurements: Because of the spatial deformation which occurs when a curved structure like the colon is mapped to the screen, it is difficult to provide a meaningful measurement tool. Although it is possible to find the distance between two user specified points, there is not a linear relationship between distances in image space and distances in the volume. A measurement tool is useful because the size of polyps corresponds to their severity - those larger than 10mm typically need to be removed.

Fortunately it is possible to integrate the Volumetric CPR with other visualisations and this is what has been done for the Biotronics3D system. This chapter describes the trade



(a) (b)

Figure 7.1: Two of the possible layouts for the virtual colonoscopy system. In (a) two Volumetric CPRs facing in opposite directions are used in conjunction with a flythrough. In (b) an additional flythrough has been added (looking backwards to see behind folds) and two more Volumetric CPRs are present.

offs which needed to be made during this integration and describes how technical problems were overcome.

7.2 System Layout

Designing the layout of the virtual colonoscopy system required some thought because we were trying to provide the numerous visualisations which were required to satisfy all users, while at the same time trying to ensure that the interface did not become too cluttered or confusing. It was important to ensure that visualisations fitted together well to avoid wasting screen space, and also that they were sufficiently large so as to allow the user to examine them.

Due to technical constraints the Biotronics3D system does not allow user configurable views so it was necessary to provide one or more sensible layouts. There was an element of disagreement amongst radiologists about exactly what visualisations were required (these opinions were mostly gained at the Radiological Society of North America conference in December 2006). For example, some users were keen to have a reverse flythrough present while other were happy with just a forward flythrough, and some users were satisfied with two Volumetric CPRs while other wanted four.

Eventually the default configuration was established to be that shown in Figure 7.1(a), and a more complete layout was introduced which contained more visualisations and is shown in Figure 7.1(b). Other layouts exist (but are not shown here) and the user can switch between them at any point during the examination.

7.3 Image Precomputation

To allow the user to interact with a system in a smooth manner it is necessary to render at a high frame rate. That is, the system should be capable of generating updated frames in a short space of time so that the user can get immediate feedback on their interaction. Experimentation with the system has shown that this frame rate should ideally be over 30 frames per second to be perceived as smooth - lower frames rates are still acceptable but it is generally desirable to keep them as high as possible.

As we reported previously [75], the frame rate of the Volumetric CPR was about 8-9 frames per second. This is of course highly dependent upon properties such as image size and rendering quality, and higher frame rates could be achieved through optimizations and dedicated graphics rendering hardware, but this essentially means that the CPU has no time to dedicate to other tasks. If the Volumetric CPR runs at 8-9 frames per second on its own, then when combined with other processor intensive visualisations such as the flythrough the system will simply be too slow.

In order to circumvent this problem we have chosen to *precompute* the Volumetric CPR images. When rotation is not being performed the Volumetric CPR can essentially be considered to be a single long image of which the user only sees a small portion at a time. Previously we were rendering this small portion each frame, but when using the precomputation approach we instead store the whole image in memory and simply copy the desired part to the frame buffer when it is required. This is significantly faster than performing the rendering.

The approach has, however, a couple of drawbacks. Firstly the user looses the ability to rotate the Volumetric CPR, although full surface coverage can still be ensured by providing multiple images looking in different directions. Secondly there is a memory overhead due to the requirement that the whole image is stored. This can be significant when large datasets are loaded and is discussed further in the evaluation (see chapter 9).

7.4 Picking

'Picking' is the process of determining the 3D location in space corresponding to a 2D position in the final image. The 2D position would typically be specified by a mouse click and the 3D position in the dataset might be required for various reasons such as displaying its density or, as will be described in Section 7.5, performing view synchronization.

Picking in volumetric data can typically by implemented using raycasting, similar to that which is done during the rendering process. When the user clicks on a given pixel a ray can be cast from that pixel, through 3D space, until it hits the surface which is of interest to the user. This approach can also be used with the Volumetric CPR but, as described in Chapter 6, the setup of the ray position and direction can be a little complicated.

Because both the rendering and the picking are based on raycasting it is actually possible to do both at once. As we generate our Volumetric CPR image we simultaneously fill in a 'position' image; this is the same size as the output image but stores $\{x, y, z\}$ values in its three components rather than colours in $\{red, green, blue\}$ form. We are essentially saving the 3D position corresponding to each 2D pixel in the output and so picking now becomes a case of simply looking up the 3D location in the position image.

7.5 View Synchronization

As has been mentioned a virtual colonoscopy system will typically have many different views which must be synchronized if meaningful information is to be conveyed. For example, when a camera is placed in a given position in the colon we expect the flythrough to show the view from that point, the Volumetric CPR to show a side view of the surrounding area, the overview to show a marker for where the camera is, and the MPR to show a slice through the cameras location. Keeping all these views synchronized is non–trivial because they each employ different camera models. For example:

MPR and Flythrough: These two visualisations both require a camera position and a camera orientation to display their images. The position is simply a vector of $\{x, y, z\}$ values while the orientation consists of three vectors defining a local coordinate system. This coordinate system include a *forward* vector (representing the direction the flythrough will look) and *up* and *right* vectors which specify the rotation of the flythrough and provide sampling directions for the MPR.

Volumetric CPR: This again requires a position (which is the same as that for the flythrough) but, rather than using the forward vector as an orientation, it actually uses an angle of rotation *around* that vector. A 'zoom' factor should also be provided which controls how the centreline points map to scanlines; this is different from the zoom factor of the flythrough (which controls the field of view).

Overview: The overview camera is completely separate from the others in that it is placed outside the volume. However, it needs to display a marker at the flythrough camera's position, and might also rotate to avoid ambiguous situations.

The above points show that synchronization is not always trivial, though from the users point of view it should just seem to be automatic. The remainder of this section will deal only with synchronization between the flythrough and the volumetric CPR as this is the most relevant to our work.

Synchronization during the automatic flythrough is performed by rotating the flythrough camera to match the Volumetric CPR. Both flythrough and Volumetric CPR are based on the same centreline through the data and a rotation for the Volumetric CPR is specified by the user either by freely rotating it or (as is the case in our system) by choosing one one of several predefined rotations. The coordinate system of the camera is then derived from the local coordinate system of the centreline point by rotating both the up and right vectors around the forward vector. The result is that during the flythrough the camera twists and turns to maintain synchronization but this happens relatively slowly.

If the user stops the flythrough and clicks on a point on the Volumetric CPR then the flythrough camera must also jump to this point. This is done using the following procedure:

- First the system must identify the point p on the surface where the user clicked. As described in Section 7.4, this is done by simply looking at the 'position' image.
- Next the system must identify the closest point c_i on the centreline to where the user clicked. This is done by computing the distance between the surface point and each point on the centreline and determining the minimum.
- We now know where the camera should be positioned, and where it should be facing. The camera forward vector c_f is computed as the normalized vector from c_i to p. That is:

$$c_f = \frac{p - c_i}{||p - c_i||} \tag{7.1}$$

- The camera's right vector $c_r$is set to be the normalized tangent to the centreline. This ensures it is always facing in a direction which is normal to the centreline:

$$c_r = \frac{c_{i+1} - c_i}{||c_{i+1} - c_i||} \tag{7.2}$$

- The up vector c_u can then be computed as the cross product of the forward and right vectors:

$$c_u = c_r \times c_f \tag{7.3}$$

Note that if the user clicks on a part of the surface which is not in the centre of a Volumetric CPR scanline then (in our system) there will be a difference between the image presented by the flythrough and the Volumetric CPR because our Volumetric CPR will not rotate to centre the point whereas the flythrough will. This is not a limitation of the Volumetric CPR in the general case; just in our system as we have prevented rotation by precomputing the images.

7.6 Summary

Seamlessly integrating the Volumetric CPR into an existing virtual colonoscopy system allows it to work in collaboration with other visualisations which are present. It also allows it to benefit from other features of the system such as DICOM handling, PACS integration, and memory management. This chapter has explained the integration issues which we encountered and has explained how we overcame them, especially with a view to make the Volumetric CPR interact and synchronize with the virtual flythrough.

This endeavour has been successful and the Volumetric CPR now forms a key component of the Biotronics3D virtual colonoscopy package. As we will discuss in Chapter 11 there are also plans to incorporate it into other parts of the system so that it can also be used, for example, for CT angiography.

Chapter 8

Advanced Techniques and Uses

8.1 Introduction

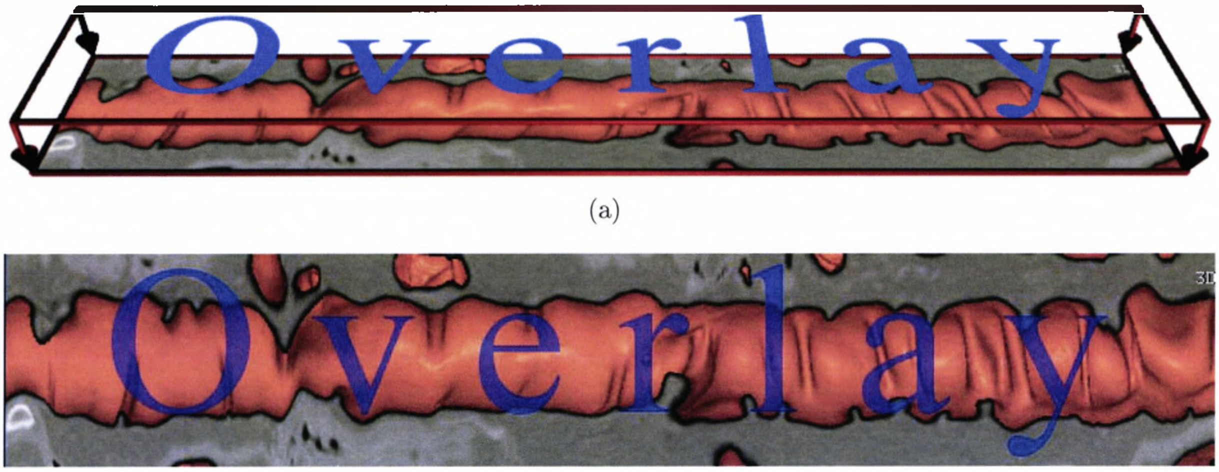
While the Volumetric CPR as presented in the previous chapters is a very powerful tool, we have identified some ways in which we can extend its functionality to make it even more useful. This chapter gives details on some of these techniques and their implementation. We begin by introducing the idea of overlays which superimpose additional information on top of a Volumetric CPR image. We show two uses for these overlays - conveying surface coverage data and translucency rendering - but it is likely that other uses could be developed in the future.

We also address the problem of performing lighting calculations to shade the surface of the colon in the Volumetric CPR; moving beyond the simple model which was used in the previous chapter. We introduce *positional* and *directional* lighting models and apply them both in a *local* and *global* context in order to enhance the users perception of the shape of the surface. Lastly we introduce the concept of *shape–driven lighting* as a method of encoding structural information into the colour of the light.

8.2 Overlays

It is often the case that the user wishes to selectively view additional information about the data set at the same time as they are examining an image. This is widely used in traditional visualisation approaches to convey, for example, information such as the patient's name, the scan details, and the position in the volume. Interactive techniques such as measurements will also involve drawing over the underlying image - all these techniques are implemented as overlays.

The volumetric CPR extends this concept by allowing *arbitrary* images to be overlaid. It also supports blending, such that the underlying image is not completely obscured by an overlay, but instead each pixel is a combination of both. The concept is illustrated



(a)

(b)

Figure 8.1: By performing compositing between the two images shown in (a) we are able to generate the image shown in (b).

in Figure 8.1. Figure 8.1(a) shows two images, a Volumetric CPR image and an overlay image above it. Figure 8.1(b) shows the result of performing a weighted blending of these, note that it is still possible to see structures underneath the overlay (for example, the polyp-like structure under the letter 'r').

In general it is possible to combine an unlimited number of images using a compositing equation similar to Equation 4.13 used earlier for combining samples during volume rendering. In practice a large number of images has an impact on the rendering speed and image clarity, and we have also found that one blended overlay is sufficient. This allows us to simplify the compositing expression to:

$$RGB_{result} = RGB_{vcpr} \times (A_{overlay} - 1) + RGB_{overlay} \times A_{overlay} \tag{8.1}$$

This is evaluated for each pixel in the image. Note that for our purposes the Volumetric CPR image is fully opaque (it has no alpha component). Information such as patient details can still be overlaid on top of the composited image; this is done by simply overwriting pixels rather than performing any blending. However, the user does have the option to turn this information on or off.

Having described how the overlays are implemented, we now present two examples showing how they can improve the diagnostic process.

8.2.1 Surface Coverage Tool

It was mentioned in Section 5.4 that one of the significant problems with a virtual fly-through is that parts of the surface may not be seen by the user. This happens for several reasons:

1. The colon has a large number of folds which can hide parts of the surface lying behind them. A partial solution to this problem is to do a flythrough in each direction, but this still fails in some cases (such as when there are two folds adjacent to one another).

2. The camera has a limited field of view (typically between 45° and 90°). Increasing the field of view beyond this tends to cause distortion in the resulting image.

3. It is difficult to get the camera to point in a optimal direction. Hence when going round corners the camera may have a tendency to point at the wall rather than down the colon, etc.

The Volumetric CPR can help with the surface coverage problem in two ways. Firstly, it can help boost surface coverage because it views the colon from a different direction. This allows it to see behind most folds and, although there are still situations where it still might miss parts of the surface, the improvements are very significant (see the evaluation in Chapter 9).

Secondly, the Volumetric CPR can make use of overlays to monitor which parts of the surface have been seen *by the flythrough,* and display this information to the user in real time. In order to do this, we first need a method of determining what has been seen by the flythrough, and secondly we need a method of providing this information to the overlay.

In order to achieve this we introduce the concept of a *seen volume*. The seen volume has the same dimensions as the data volume but only requires one bit per voxel; hence it only uses $\frac{1}{16}$ of the memory used by the data volume (as this requires two bytes per voxel). Each voxel can be in either of the states $\{seen, unseen\}$ and initially all voxels are set to unseen. During the rendering of the *flythrough* rays are cast until they hit a surface at which point lighting calculations are performed (as normal). However, when the ray hits the surface we also change the state of the voxel at the corresponding location of the seen volume to indicate that that particular point on the surface has been seen.

When generating the *Volumetric CPR* the process is reversed. Rays are cast and, when they hit the wall, the seen volume is tested to determine whether the flythrough has already seen that location. If so the corresponding pixel on the overlay is coloured to indicate this.

An example of the result can be seen in Figure 8.2.1. In this case the camera is moving from the left hand side of the image to the right; it can be seen that some areas behind folds are missed. Also some large patches are missed due to the camera not pointing in the right direction all the time, but instead accidentally facing a particular wall when travelling round corners. It can be seen that there are also missed regions which lie between two folds - in this case even doing a flythrough in both directions would not resolve the problem.

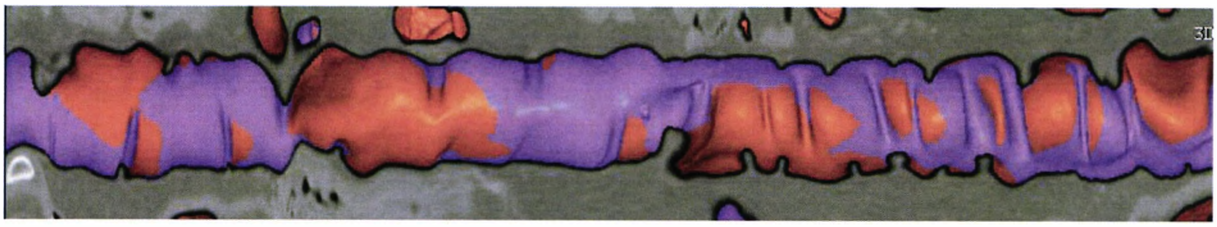


Figure 8.2: The regions marked in purple on the Volumetric CPR indicate the parts of the surface which have already been seen by the flythrough. As the flythrough proceeds the purple regions are updated in real time.

Using the surface coverage tool not only allows the user to observe this problem, but also to remedy it when it occurs. Simply clicking a given point on the Volumetric CPR causes the camera to jump to that point and align itself with the designated location. This allows the user to immediately examine the region which had been missed and the Volumetric CPR is updated to reflect this. This is significantly easier (and more immediate) than the approach used in other systems [58] in which the flythrough must be completed before a list of missed regions is presented to the user.

8.2.2 Translucency Window

As was explained earlier in Part I, the preparation for a virtual colonoscopy can involve administering to the patient a *Contrast Enhanced Fluid* (CEF) which tags any remaining material in the colon and causes it to show up bright white on the CT scan. One of the problems with this is that small deposits of tagged material on the surface can look like polyps, potentially causing false positives.

The Viatronix system [58] introduced a solution to this via the concept of *Translucency Rendering* [62]. Rather than having a transfer function which stops at a particular isovalue (as was shown in Figure 6.5) they define a transfer function which samples a few voxels into the wall and maps the density to a range of colours. Red or green indicate tissue while white indicates contrast enhanced fluid; the full transfer function is shown in Figure 8.3 and is based as closely as possible on the one given by Liang [62].

The work flow for using the Viatronix system is as follows. The user performs a flythrough in the colon until encountering a suspicious structure which looks like a polyp, at this point the user stops the flythrough and points the camera at the structure. In order to determine whether it really is a polyp the user activates the *translucency window*; a region of the screen in which translucency rendering is performed. After determining the nature of the structure the user resumes the flythrough. When used in this way, translucency rendering interrupts the flythrough by requiring the camera to stop.

When combined with the Volumetric CPR the translucency rendering can be used in such a way that it does not interrupt the work flow. By implementing the translucency

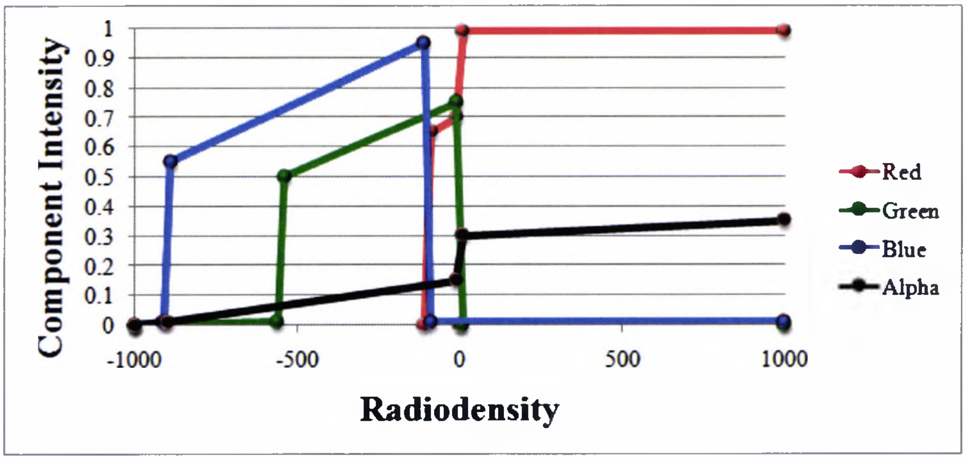


Figure 8.3: The transfer function for translucency rendering in the colon (as originally given by [62]).

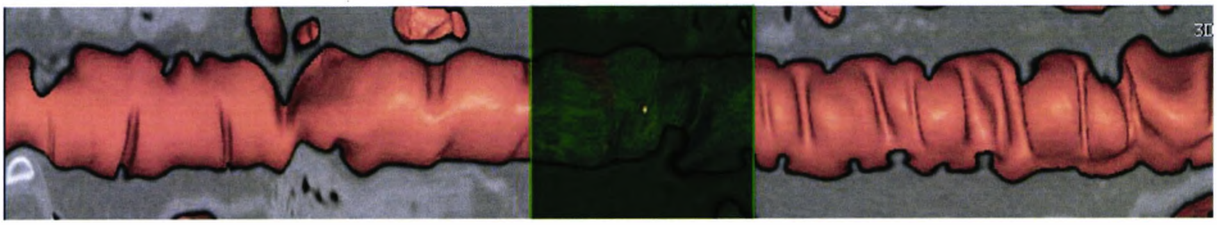


Figure 8.4: A Volumetric CPR with a translucency window applied. Note that tissue shows up green or red, while the deposit of fluid in the centre of the image shows up white due to its high density.

rendering as a overlay we can ensure that, at some point in time, every part of the surface hasthe translucency rendering applied without requiring the user to stop the flythrough. We choose to replace the central part of the Volumetric CPR with the translucency overlay (Figure 8.2.2) but, in general, any combination can be used.

Note that the translucency rendering has no meaning when applied to the greyscale part of the image; this is simply tinted green for aesthetic appeal.

8.3 Visualising Behind Fluid

As well as creating false polyps (which can be identified with our translucency window) contrast enhanced fluid can also obscure the users view of the colon surface. For this reason, such contrast enhanced fluid is typically removed via *electronic cleansing* algorithms, but there are some potential drawbacks:

- Care must be taken to avoid accidentally removing polyps which are inside the fluid as this would result in false negatives. Hence the segmentation algorithms need to be conservative, and as a result a thin layer of fluid can be left behind after segmentation is complete. Current 3D visualisation techniques do not provide effective ways of seeing behind such layers.

- Some doctors prefer to view the base data rather than take the risk that an overly aggressive segmentation might accidentally remove any polyps. In the case of most 3D visualisation techniques this causes problems when regions are completely blocked. For example, a flythrough will simply see a wall blocking the path while a colon unfolding will result in a blank image for the corresponding section.

However, assuming the segmentation stage is able to successfully compute a centreline, the Volumetric CPR can still generate a useful image when contrast enhanced fluid is present. In such an image the tagged material is still obstructing the users view of the surface but, crucially, the user is able to see the boundary between the material and the colon wall. Hence the user is aware of the material and by rotating the Volumetric CPR they are still able to screen for the presence of polyps.

8.4 Lighting Strategies

Up until this point, lighting within the Volumetric CPR has used a simplified version of the popular and well–established Phong reflection model [38] which was introduced in Section 4.4.2. In most volume rendering scenarios it is straightforward to calculate the necessary vectors (shown again in Figure 8.5). Given a point s_{pos} on the surface the view vector v_{dir} is simply the vector to the viewer's position v_{pos} from s_{pos}, the light vector l_{dir} is the vector to the light's position l_{pos} from s_{pos}, the normal s_{norm} is computed from the data using an approach such as Central–Difference or by application of the Sobel operator, and the reflection vector r_{dir} is calculated as:

$$r_{dir} = 2(s_{norm} \cdot l_{dir})s_{norm} - l_{dir} \tag{8.2}$$

Unfortunately, when applied to the Volumetric CPR, this model does not give realistic results due to the distortion of space which invariably occurs when an arbitrary curved structure in three dimensions is mapped to a two dimensional plane. The problem occurs because, although the lighting may be correct in the world space in which it is usually computed, it ceases to be correct once the curved structure undergoes a non–linear deformation in order to straighten it. Surfaces which were facing each other in object space may no longer face each other, and the distance between points may change. Hence both the angle at which light hits a surface and distance which it must travel are different between object and image space.

We now examine the lighting models we have developed. We break the models down into two categories; *local* models in which each centreline point uses its own 'local' pa-

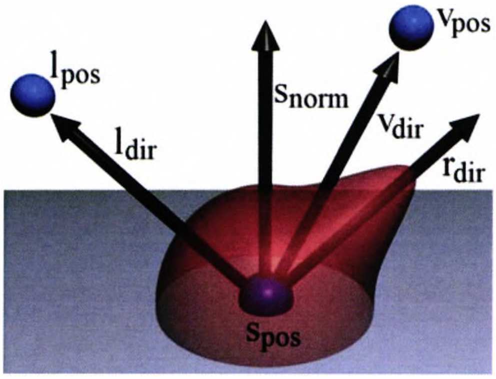
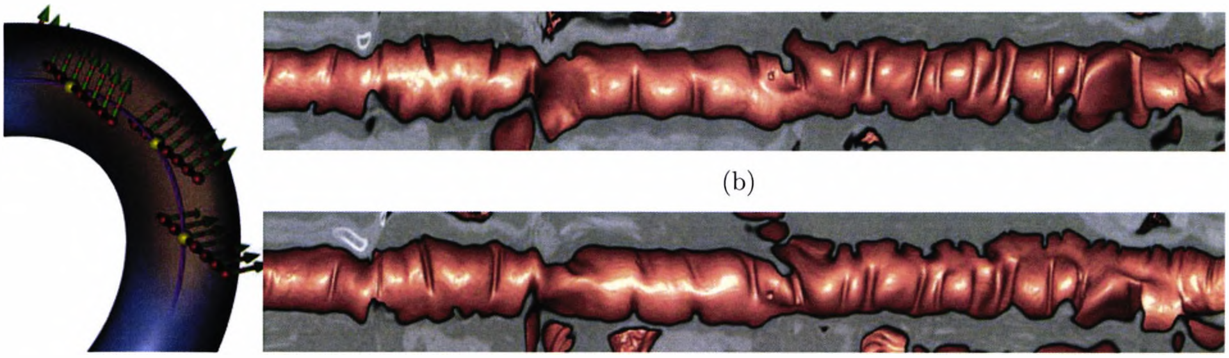


Figure 8.5: The vectors required to compute the Phong lighting for a given point on the surface. s_{norm} *is the surface normal,* l_{dir} *points toward the light source,* v_{dir} *points toward the camera, and* r_{dir} *is the reflection of* l_{dir} *about* s_{norm}.


(b)

(a)

(c)

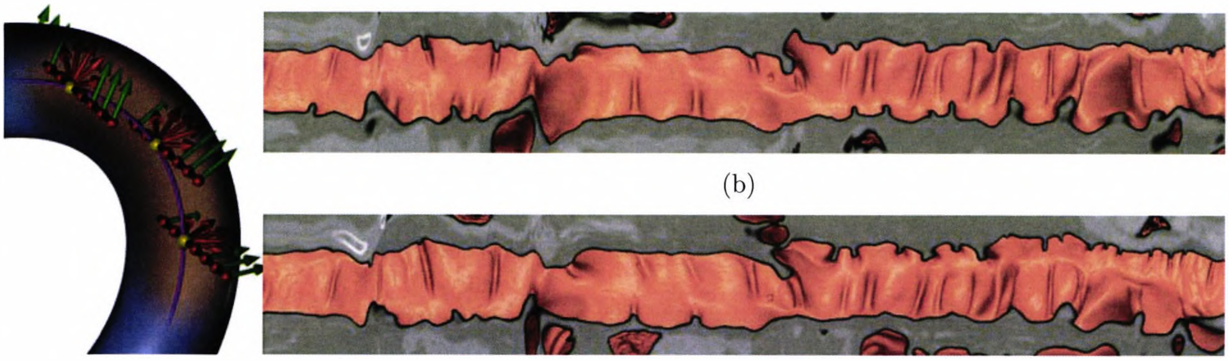
Figure 8.6: When a directional light source is used the light vector (indicated by the red arrows) is the same as the view vector (indicated by the green arrows); hence in Figure 8.6(a) the arrows are shown with bands of both colours (contrast to Figure 8.7(a)). The resulting images are shown in 8.6(b) and 8.6(c) with slightly different rotation. Notice how the shading (and hence the appearance) of a point on the surface varies as it is rotated.

rameters and *global* models in which the parameters are constant for all centreline points (they are taken from a global light source).

8.4.1 Directional Local Lighting

The most basic approach to lighting uses a directional light rather than a positional one. The direction which the light comes from changes between scanlines such that *the light direction is always the same as the view direction*; that is $l_{dir} = v_{dir}$. This is illustrated in Figure 8.6(a). Once the tubular structure is flattened to the image plane the light appears to come from 'above' as shown in Figures 8.6(b) and 8.6(c).

In an image rendered with this lighting approach the lighting does not change as the image is scrolled left–to–right, but does change if the image is rotated around the centreline. As discussed earlier, this works well when several non–rotating Volumetric



(a) (b) (c)

Figure 8.7: When positional light is used the light vectors are computed between the current centreline point and the point on the surface; hence they vary between samples (contrast to Figure 8.6(a)). The resulting images are shown in 8.7(b) and 8.7(c) with slightly different rotation. Notice how the appearance of a point on the surface now stays relatively constant despite rotation.

CPRs are positioned next to each other because the entire image can be precomputed and the appropriate part copied to the framebuffer as it is required; this is much faster than rendering sections of the image on demand.

This approach also allows an optimization to the computation of the specular component. Because the lighting and view direction are the same there is no need to explicitly calculate a reflection vector; the specular component can instead be specified as:

$$spec = l_{spec} \times s_{spec} \times max(s_{norm} \cdot v_{dir}, 0)^{exp} \tag{8.3}$$

8.4.2 Positional Local Lighting

One of the drawbacks of the directional lighting approach is that, because the lighting is dependent on the angle of rotation for the Volumetric CPR, the same point on the surface looks different when viewed from different angles. This is to be expected, but when the user is scrolling along multiple Volumetric CPR images at a time it may be useful for points in adjacent images to look as similar as possible in order to allow them to be mentally registered.

This can be addressed by using a point light source rather than a directional one. When a point light source is used the lighting vector is computed only from the position of the light and the position of the surface which is being lit; hence it is independent of the angle of rotation of the Volumetric CPR. The concept is illustrated in Figure 8.7(a) while Figures 8.7(b) and 8.7(c) shows the resulting images for two slightly different rotations. Compare to Figures 8.6(b) and 8.6(c) to see how the lighting is now invariant to rotation.

This is still a 'local' lighting model because, for each scanline we render, the light position is taken to be the nearest centreline point. This means we maintain the useful property that scrolling the image does not modify the lighting, hence we are still able to use precomputation of the images to enable fast rendering. We also gain the advantage that the same point looks similar in adjacent images with different rotations; the lighting is now consistent although the user is seeing the point from a different direction.

8.4.3 Global Lighting

The simplicity of the local lighting models stems from the fact that each scanline is lit independently of the others. Each scanline has its 'own' light source. In the case of directional lighting each light source is effectively lined up with the camera for that particular scanline, and in the case of positional lighting the light source is placed on the centreline near to the rendered surface. A light for one scanline does not affect its neighbors. The model works well because the lighting is set up properly for each scanline, and within any given scanline distortion does not occur. The distortion seen when straightening a structure only occurs when several scanlines are considered together.

We consider now a different scenario in which there is actually a light placed at a specific location on the centreline, and all scanlines must be lit with this light rather than using their own 'local' light. This is useful because we can then allow the user to move this light along the centreline to observe how the lighting changes – this provides important visual cues to the true shape of the surface.

Figure 8.8(a) shows a curved tubular structure with a light at one end. Imagine we wish to apply the Phong reflection model to compute the lighting at each point on the surface. Assuming we have the ambient, diffuse, and specular properties of the light and the surface, we simply need to compute the relevant lighting vectors. The naive approach would be very similar to computing the vectors for the local point light discussed earlier, with the exception that the light position is now the same for all scanlines. That is, the light vector l_{naive} would be found by subtracting the light's position from the point being lit, and normalizing.

This clearly does not make sense when the structure is being distorted because it can result in the surface being lit from behind (as shown in Figure 8.8(a)). This would result in the surface receiving only ambient light at this location. The correct path for the light to follow is given by $l_{correct}$; this follows the shape of the structure and causes the light to come from the correct direction when the structure is straightened.

Rather than actually modelling a curving light ray, we instead transform the position of the light (on a per–scanline basis) to a location from which we can compute the light vector as the difference between the light position and the point we wish to light. This is shown in Figure 8.8(b).

To perform this transformation it is necessary to know the distance d, measured along the centreline, between the light and the centreline point currently being rendered. This

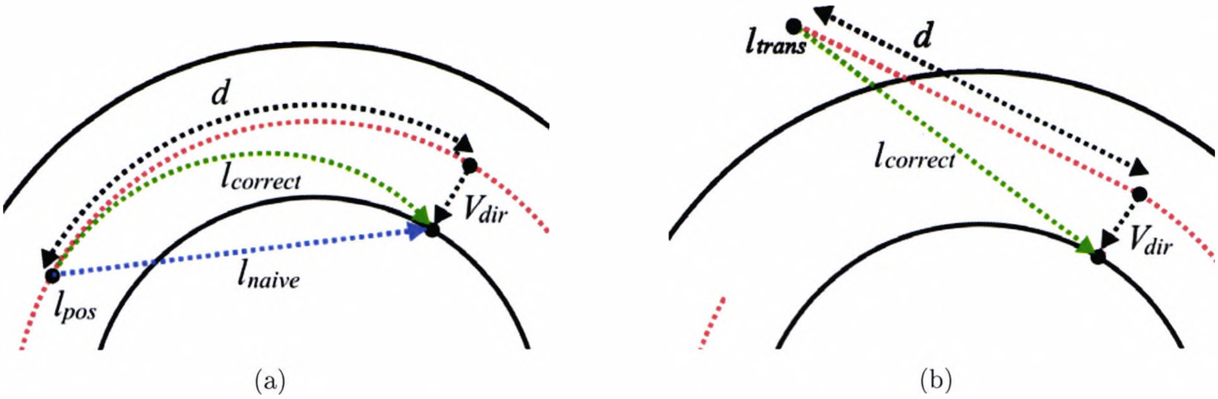


Figure 8.8: Using the naive approach, the light vector l_{naive} is computed straight from the light to the surface; once straightened this results in the light coming from the wrong direction (a). The correct path for the light $l_{correct}$ curves with the structure. Rather than actually curving the light rays, we mimic the effect by transforming the light source as shown in (b).

distance is signed depending on which side of the centreline point the light is. If we have a list c of centreline points, then we can denote the position of the light and the current centreline point as c_l and c_p respectively. If the centreline points have a fixed spacing s then d can be trivially computed as:

$$d = s(p - l) \tag{8.4}$$

If the centreline points have a variable spacing then the distance is computed as the sum of all intermediate distances:

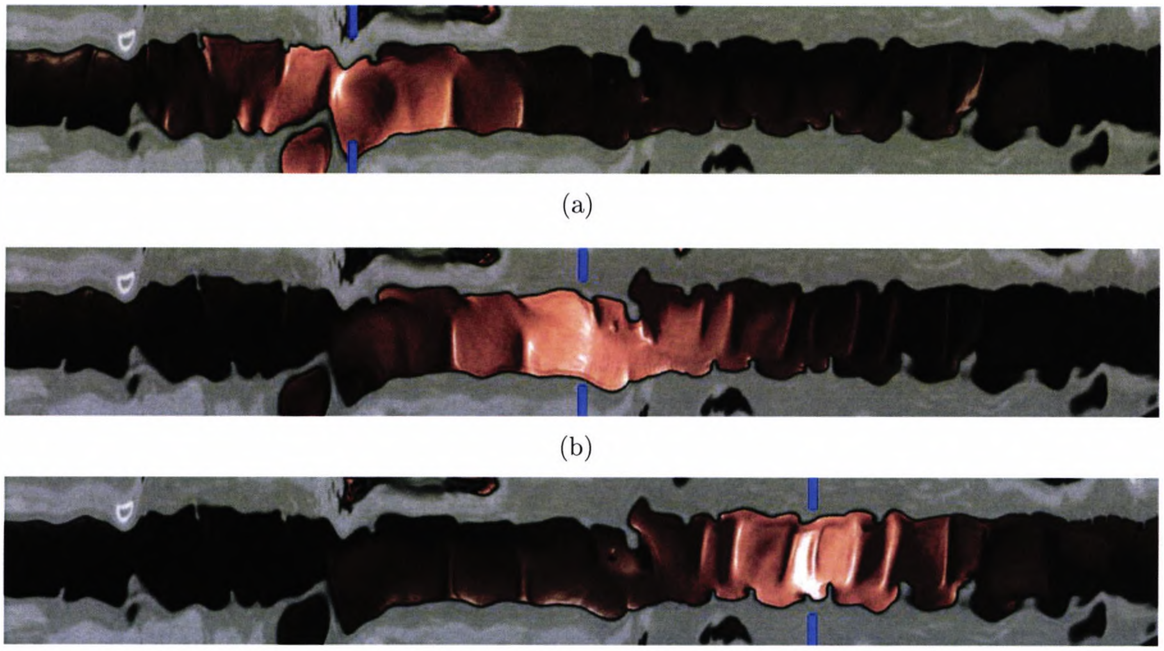
$$d = \sum_{i=l}^{p-1} ||(c_{i+1} - c_i)|| \tag{8.5}$$

As an optimization, it is possible to precalculate the distance from each centreline point to each other centreline point and store the results in a two dimensional look-up table which is indexed by the centreline point indices. The transformed light position l_{trans} can now be computed from the current centreline point c_p, the normalized tangent t to the centreline at that point, and the distance d:

$$l_{trans} = c_p + d \times t \tag{8.6}$$

Computation of the light vector is now done as before, i.e. by subtracting the position of the point on the surface from the position of the light. Figure 8.9 shows the same stretch of colon, with the same viewing direction, but different lighting positions. Notice how this causes different shading on the surface.

Figure 8.9 also shows light attenuation occurring as points on the surface become more distant from the light source. Again, it is not sufficient to use the linear object–



(a)

(b)

(c)

Figure 8.9: The same section of colon with the light placed in different locations; in each image the light is placed level with the blue markers and is centred vertically. The shading on the colon surface can clearly be seen to be different in each case.

space distance between the light source and the surface because the straightening process can change this. Instead the distance d along the centreline and the length of the view vector v_{dir} can be used to compute the correct distance between the light source and the surface. Assuming a quadratic drop–off for the light, the final intensity is computed as:

$$intensity = amb + \frac{dif + spec}{\sqrt{d^2 + ||v_{dir}||^2}} \tag{8.7}$$

Using a single global light which can be moved is useful as it helps the viewer to better understand the shape of the surface. Under static lighting it can be hard to determine whether a particular feature is projecting into or out of the tubular structure. This is because the rays within any scanline are cast in the same direction and so an orthographic projection results[1]. However, when a global light is moved over the feature the specular highlight moves in a way which clarifies the direction in which a feature is pointing.

When using a global light it is no longer possible to completely precompute the image, even when interaction is restricted to scrolling. However, if additional speed is required and memory permits, it is possible to prerender 'position' and 'normal' images. In this case the sampling and raycasting are performed in an initialization stage and instead of writing a colour (which cannot be computed at this time because the light position is

---

[1] Referring to the projection as 'orthographic' may seem a little counter–intuitive because the rays are not all parallel *between* scanlines. However, because of the straightening process the rays do end up being parallel in image space, and so as far as the user is concerned it is an orthographic image.

not known) the position of the surface and its normal are written to the 'position' and 'normal' images.

At runtime, it is no longer necessary to perform raycasting because the position and normal for a given point have already been found. The algorithm simply takes the light position, transforms it into the local space of the current centreline point, and uses the precomputed position and normal to apply the lighting. If rotation is required as well then all raycasting and lighting calculations must be done on the fly.

8.5 Shape Driven Lighting

When working with many volumetric datasets (such as those obtained by CT or MRI) the value at a given voxel is one–dimensional and represents a property such as density. The value at a given voxel does not convey any colour information. As explained in Section 4.4.3 artificial colouring is added during the rendering stage, by a user definable *transfer function* which maps the one–dimensional value into a three or four dimensional $\{R, G, B\}$ or $\{R, G, B, A\}$ colour space. When using the Volumetric CPR all rays terminate at the same threshold and hence the entire surface is usually the same colour.

One of the drawbacks of the Volumetric CPR is that there is no way to know the shape of the original structure, because whatever shape the original structure was it gets mapped to a single long image. Typically the Volumetric CPR would therefore be used alongside an overview of the structure of interest, this would contain a marker indicating the part of the structure the user is currently viewing.

In this section we extend the lighting approaches presented earlier such that the colour of the light is determined by some shape–based feature of the original structure. This shape–based feature is computed on a per–centreline–point basis and as a result changes over the length of this image. This helps recover some of the shape information which was previously being lost.

The final pixel colour is a product of both the light and the surface colour. For simplicity we consider the surface to be white and hence its final colour is completely dictated by the lighting, but generally speaking this does not have to be the case. We also render the images in this section using the local positional lighting model although a global model could be used instead.

8.5.1 Direction Driven Lighting

One shape feature which can be used to control the colour l_{rgb} of the light for a given centreline point is the direction in which the curve is travelling at that point. This tangent t to the curve is computed for any point as the difference between that point and its successor; it is also normalized because we are concerned with the direction of this vector rather than its magnitude.

We wish to map t to the light's colour such that the x component of t controls the intensity of the red component of the light, y controls the green component, and z controls the blue component. Because the components of the vector are signed and in the range $[-1, 1]$ while the colour components are unsigned and in the range $[0, 1]$, one of the following expressions is used to perform the mapping:

$$l_{rgb} = |t| \tag{8.8}$$

or

$$l_{rgb} = \frac{t}{2} + 0.5 \tag{8.9}$$

Figures 8.10 and 8.11 show these colour mappings applied to both a model of the colon, and to a Volumetric CPR of the entire length of the colon. In the case of examining a colon, this approach is powerful because most colons have a roughly similar shape and so, after exposure to several images, the viewer is able to quickly determine where they are without the use of an external overview.

8.5.2 Curvature Driven Lighting

In Chapter 6 we discussed the trade offs between the different projection methods which can be used in the Volumetric CPR, and highlighted the fact that the straightened projection can cause distortion in areas of high curvature. This is because the rays in adjacent scanlines are not necessarily sent in parallel. When part of the centreline is curved, the rays being cast can diverge (potentially missing features on the surface) or converge (potentially causing features to appear twice). It is difficult to spot this distortion by eye, and therefore it is useful to have a method of drawing the users attention to those areas where it is likely to occur.

Because curvature and distortion are closely related it is useful to use the curvature at a particular centreline point to control the colour of the light at that point. Within our system the curvature at a given centreline point is defined as the magnitude of the difference between the normalized tangent at a point, and the normalized tangent at the following point. That is, for a centreline point c_i with tangent t_i, the curvature k_i is:

$$k_i = \frac{||\hat{t_i} - \hat{t}_{i-1}||}{2} \tag{8.10}$$

Given this curvature we can generate the image in Figure 8.12 by changing the light colour from green, through yellow, to red as curvature (and hence distortion) increases.

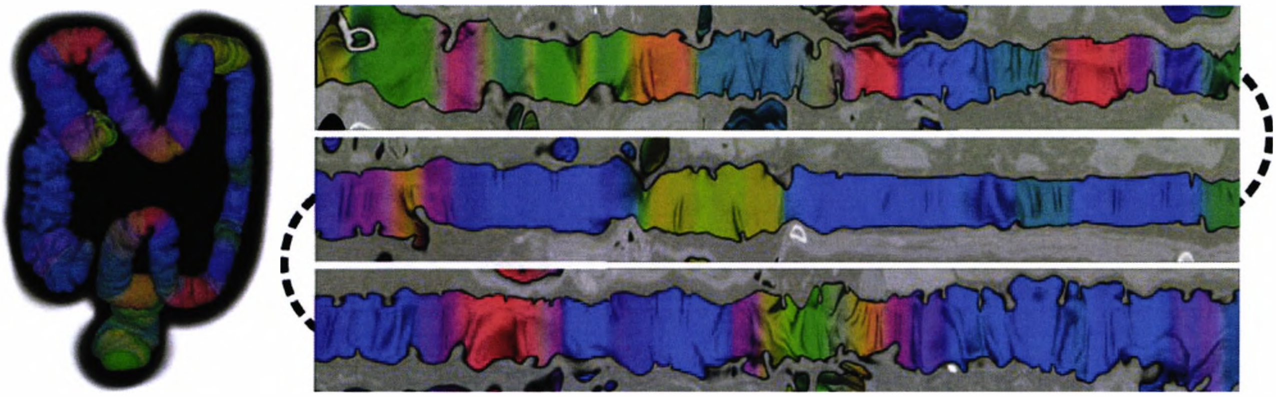

(a) (b)

Figure 8.10: The result of mapping the direction of the tangent at a given point to the colour of the light. In this case, Equation 8.8 was used to perform the mapping.

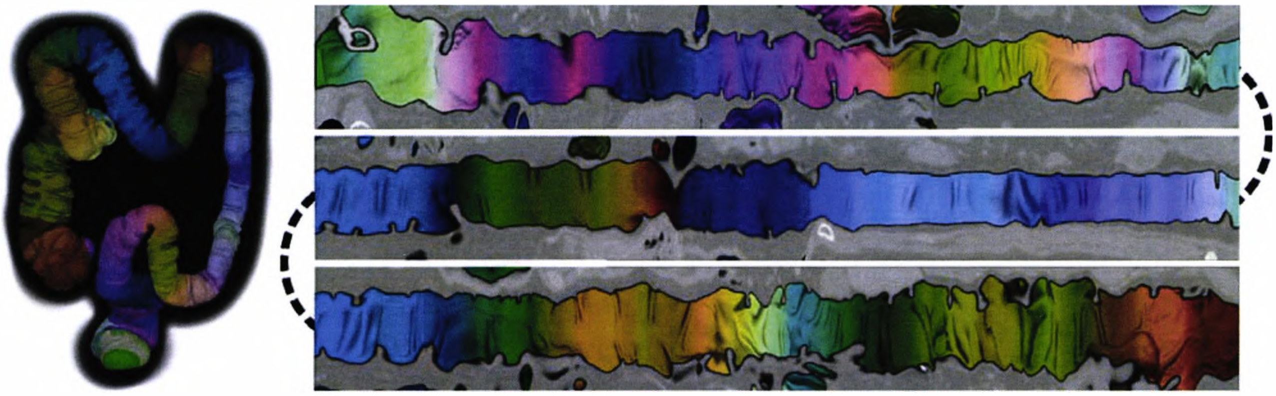

(a) (b)

Figure 8.11: The result of mapping the direction of the tangent at a given point to the colour of the light. In this case, Equation 8.9 was used to perform the mapping.

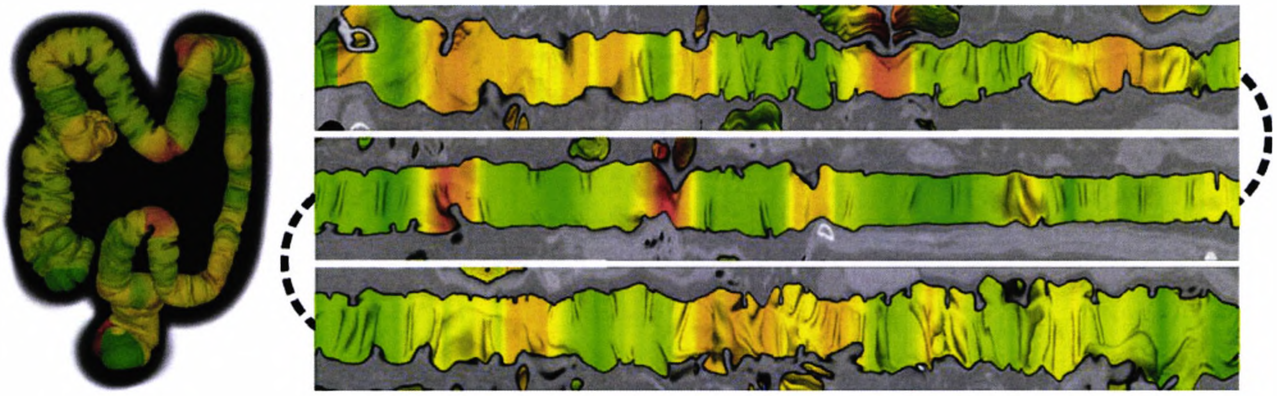

(a) (b)

Figure 8.12: The result of mapping the direction of the tangent at a given point to the colour of the light. In this case, Equation 8.10 was used to perform the mapping.

8.6 Summary

This chapter has presented some techniques to expand upon the Volumetric CPR as it was presented in Chapter 6. We introduced the concept of overlays as a way of combining additional information with the Volumetric CPR image and demonstrated two uses of this idea. Firstly we were able to monitor surface coverage in the flythrough to make sure regions were not missed, and secondly we were able to use a translucency window to identify contrast enhanced fluid in the colon. This work was presented at the 21st International Conference on Computer Aided Radiology and Surgery (CARS 2007) and the version of the paper which appeared in the proceedings is included as Appendix B.

We have also developed some advanced lighting models which help to better convey the structure of the surface. Global lighting allows a single point light to illuminate the whole surface, and so moving this light causes important changes to the shading. Shape driven lighting is used to modify the colour of the light in accordance with shape related features such as curvature – this can be used to indicate regions of the colon where distortion can potentially occur. This work on lighting is currently being integrated into a paper but has not yet been published.

Part III

Evaluation and Conclusion

Chapter 9

Evaluation

9.1 Introduction

Having developed our new visualisation technique it is important to evaluate the extent to which it meets the original aims, and to determine how it compares to existing solutions. It is also important to understand the impact the system will have on those who will make use of it. This chapter is dedicated to assessing the system in a quantitative sense, and also to presenting the feedback we have received from clinicians who have made use of it.

9.2 Methodology

Our evaluation consists of three main components, the details of which are presented in corresponding sections of this chapter. As an overview, we provide:

- A quantitative analysis of the Volumetric CPR as compared to other approaches such as a virtual flythrough, colon unfolding, and MPR. In comparing these approaches we measure and discuss quantities such as surface coverage, image generation time, and resource usage; each of these can be determined with a high degree of accuracy across a number of data sets. We also present details of the methods we used to measure each of the criteria and the ways in which they can be further improved or traded against each other. Results relating to surface coverage have already been presented in our paper at the 21st International Conference on Computer Aided Radiology and Surgery.
- An explanation of how our ongoing formative evaluation affected the design and implementation of the system, including integration into existing work flows as well as issues such as the layout and operation of the user interface. We identify points in the development of the system where there were several options for how to proceed, and will explain how and why medical input led to a particular solution. It is

important to understand the factors which influenced these decisions because, if they change in the future (for example, moving to a new problem domain), it might be beneficial to review the decisions which were made.

- Final clinical feedback obtained during the summative evaluation of the system. This includes feedback from a radiologist who had not previously seen the system, as well as from an expert in IBD who gave his opinions on how the system could be adapted for the projects longer term goal of IBD detection and visualisation.

The analysis and feedback have been very positive and has also given some directions for future work in IBD visualisation (this future work is outlined in Section 11.2). The work on surface coverage analysis was also published [76] in the International Journal of Computer Aided Radiology and Surgery (the paper is included as Appendix B).

9.3 Quantitative Evaluation

In this section we compare four different virtual colonoscopy visualisations (including the Volumetric CPR) in order to determine the strengths and weaknesses of each and to determine the ways in which Volumetric CPR can benefit the field. The approaches we will be comparing are:

Slice by slice analysis: Within this category we will consider the use of both *axis aligned* slices and slices which are aligned to the centreline. In most cases the advantages and disadvantages are similar but where this is not true it will be pointed out.

Virtual Flythrough: The implementation which forms part of the Biotronics3D system will be used as the basis for comparison.

Colon Unfolding: We do not have access to an implementation of the colon unfolding technique and so our comparison is a theoretical one based on the properties described in [52].

Volumetric CPR: Our new technique, as described in Part II of this thesis.

In some ways these different approaches are not directly comparable (for example it does not make sense to talk about the surface coverage of the slice by slice approach. In other cases there are different implementation choices which may trade, for example, speed against memory usage. In the following discussion we try to identify these issues when they occur.

9.3.1 Image generation time

Image generation time is an important factor in determining the usability of a system because it relates directly to frame rate and hence to the degree of interactivity which a user experiences. If a particular visualisation approach suffers from high image generation time then the frame rate is reduced and the system appears sluggish to the user - interaction commands take a noticeable amount of time to take effect.

There is no fixed value determining what is considered to be an acceptable frame rate. Standard television sets typically produce 25 frames per second (FPS) in Europe and most people are unable to see any problems – although it should be noted that *interlacing* is used to increase the perceived frame rate. Interactive applications such as video games typically require a higher frame rate of about 30 FPS to be perceived as smooth. Most people are unable to perceive flickering above about 60 FPS but the exact value depends on the person and on other conditions such as lighting. For our purposes we just want the image generation time to be as low as possible.

Among our visualisation approaches there are two key factors which can affect the image generation time; these are resolution and quality. Resolution refers to the number of pixels in the resulting image and, in all the approaches which we are considering, this is linearly related to the image generation time. That is, doubling the image area means it will take twice as long to generate the image.

Although traditional scientific methods would dictate that resolution should be kept constant between approaches in order to make a fair comparison, this approach does not really make sense in our scenario. This is because within the final system all visualisations are *not* the same size and so it does not make sense to evaluate on this basis. In order to provide a compromise we have measured both the *image generation time* and the *pixel generation time* for each approach. The pixel generation time is given by the image generation time divided by the number of pixels (i.e. the area) and can be useful to give an indication of how the techniques would perform if they *were* all the same size. It should also be noted that most visualisation approaches can support progressive rendering, in which a low resolution image is initially displayed. If the user is not interacting with the system then a high resolution version can then be generated to replace it. Because the low resolution version is faster to create it allows a higher frame rate and much smoother interaction with the system.

The second factor – quality – covers many other aspects of the image which we aim to keep constant. For example, the same trilinear interpolation method is used both for the slice views and for the greyscale part of the Volumetric CPR. Similarly, when performing raycasting, we always use the same Phong lighting model and use a fixed transfer function to ensure that the same number of samples are taken into the colon wall. We also use a constant step size for the ray casting algorithm.

We now outline the specific considerations which affect the different visualisations when analyzing image generation time.

9.3.1.1 MPR

MPR images are very fast to generate because there is no raycasting involved - the algorithm simply has to determine the density at a large number of locations and apply window–levelling. Actually, our implementation differentiates between axis aligned MPR and arbitrary orientation MPR, which means images for a slice by slice analysis are generated slightly faster than those for a view aligned with the centreline. This is because when the slices are axis aligned it is possible to replace the full trilinear interpolation with a faster *bilinear* interpolation from only four neighbouring voxels. However, for the purposes of the comparison in this work, we consider the more general arbitrary orientation version.

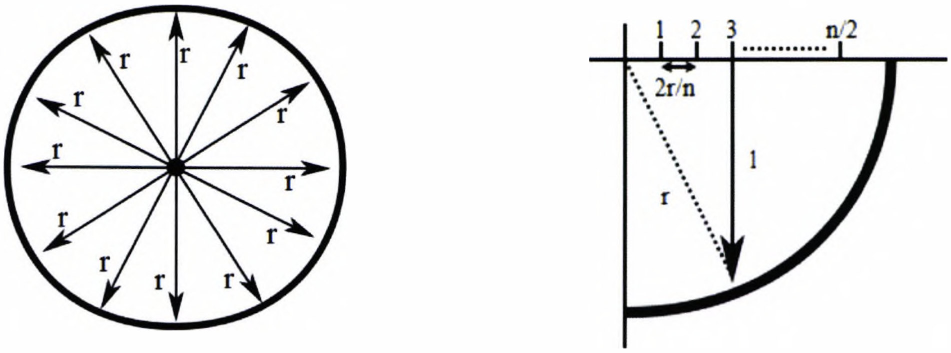
9.3.1.2 Flythrough

One of the problems with providing image generation times for the flythrough is that they are highly variable. When the camera is facing down the length of the colon the rays must traverse a significant distance and this causes the frame generation time to be longer than when the camera is placed close to a wall and is facing straight at it. Because the full range of scenarios occurs during a typical endoscopy it would be artificial to choose one over another. Hence we have computed the frame generation time by averaging over an entire flythrough; we measure the total time for the flythrough and divide by the number of frames rendered.

9.3.1.3 Volumetric CPR

The volumetric CPR is unique among our visualisation approaches in that in contains both greyscale regions which can be rendered very fast using just interpolation and window-levelling, and full colour regions which require raycasting. Hence the time to generate each pixel varies considerably based on whether raycasting is performed, and so the total image generation time is highly dependent on the ratio of colour pixels to greyscale ones. Our consistent approach to this is to adjust the size of the volumetric CPR so that at the colons widest point it just about fits inside the image.

Also, because our Volumetric CPR is precomputed, there are actually two distinct values we could measure. The first is the time taken to generate the precomputed image and the second is the time taken to copy the relevant part to the screen. Although the time to copy is what actually affects the user experience it does not really make a fair comparison because there is very little processor work involved. Furthermore, the approach wouldn't work if we wish to allow the user to perform rotation.



(a) (b)

Figure 9.1: Computing the ray lengths for the colon unfolding approach (a) and the Volumetric CPR (b)

For these reasons it is the time to generate the precomputed image which is measured. This is then scaled to give the time required to generate an image the size of the viewport.

9.3.1.4 Unrolling

As has been mentioned previously, we do not actually have an implementation of the colon unrolling technique and so we are unable to measure the true image generation time. However, we can make some observations given the similarities to our other techniques and based on the information originally given by Vilanova [52].

The entire colon unrolling technique is performed using raycasting, and we know from our experience with the flythrough and the Volumetric CPR that the time to generate each pixel can be split in to two categories. Firstly there is the time taken to traverse the ray from its start point to the surface, and secondly there is the time spent performing sampling and compositing in the wall. Of these, the time spent traversing is by far the most significant. Hence we can estimate the performance of the unrolling by computing the total distance rays must travel as compared to the Volumetric CPR.

If we consider the colon to be a tube with radius r then the cross section at any point is a circle also with radius r. We consider rendering an image which is n pixels wide and hence requires n rays to be cast for each scanline. In the case of colon unfolding the total distance travelled by the rays is simple to compute; each ray travels the same distance (r) and there are n of them (see Figure 9.1(a)). Hence the total distance d is given by:

$$d = n \times r \tag{9.1}$$

The total ray distance for the volumetric CPR is a little more complex (see Figure 9.1(b)). The length l of the i^{th} ray can be computed using Pythagoras theorem as follows:

$$\begin{aligned} r^2 &= l^2 + (\frac{2ir}{n})^2 && (9.2) \\ \Rightarrow l^2 &= r^2 - (\frac{2ir}{n})^2 && (9.3) \\ \Rightarrow l &= \sqrt{r^2 - (\frac{2ir}{n})^2} && (9.4) \end{aligned}$$

The total distance d is then given by the sum of all these ray distances, and then doubled because Figure 9.1(b) only shows half the Volumetric CPR image:

$$d = 2 \times \sum_{i=0}^{n/2} \sqrt{r^2 - (\frac{2ir}{n})^2} \qquad (9.5)$$

The practical effect of this is that, for significant values of n, the distance travelled by the rays in the unrolling approach is roughly 1.29 times the distances travelled by the rays in the Volumetric CPR; and hence takes 1.29 times longer to generate. Of course, it should be noted that *two* Volumetric CPR images are required to visualise the whole surface (whereas only one unrolled image is required), but, on the other hand, the Volumetric CPR samples the surface at a higher frequency.

9.3.1.5 Comparison

We are now in a position to present the results of the image generation time based on the properties we have just described. Timings were measured on an AMD Athlon 64 Processor 3200+ with 2GB of memory and running the 64-bit edition of Microsoft Windows Vista. Many of our algorithms are capable of running across multiple threads of execution; for the purposes of these tests we limited the algorithms to a single thread. The results are shown in Table 9.1.

It comes as no surprise that the MPR is the fastest technique because it displays a simple 2D slice through the volume and does not require any raycasting. Colon unfolding

Algorithm	Image Generation Time (s)	Image Resolution (Pixels)	Pixel Generation Time ($s \times 10^{-6}$)
MPR	0.025	512 × 512	0.095
Flythrough	0.16	260 × 270	2.279
Volumetric CPR	0.32	1024 × 293	1.054
Unrolling (Theoretical)	N/A	N/A	1.370

Table 9.1: Image generation times for the various visualisations. Because different visualisations are rendered at different resolutions we provide a normalised pixel generation time for easier direct comparison. For the unrolling we provide only a theoretical pixel generation time because we do not actually have an implementation.

and Volumetric CPR take roughly the same amount of time to generate; both are about twice as fast as the virtual flythrough because all rays are being cast sideways and so do not have to travel long distances down the colon. Generating a complete Volumetric CPR image during the precomputation stage takes about 2 seconds, but this task is only performed once.

9.3.2 Surface coverage

One of the most important criteria for an endoscopic visualisation tool is whether it allows the user to see the whole of the surface, or whether there are parts of the surface which are missed during the examination. Low surface coverage is something for which the classical flythrough approach has been widely criticized (Figure 9.2(a) shows how the surface behind folds can be missed) and there have been numerous attempts to address this problem. Figure 9.2(b) shows that the Volumetric CPR misses a different part of the surface. To compute surface coverage statistics we need to know the true area of the colon, and the amount of it which is seen by each of the visualisations.

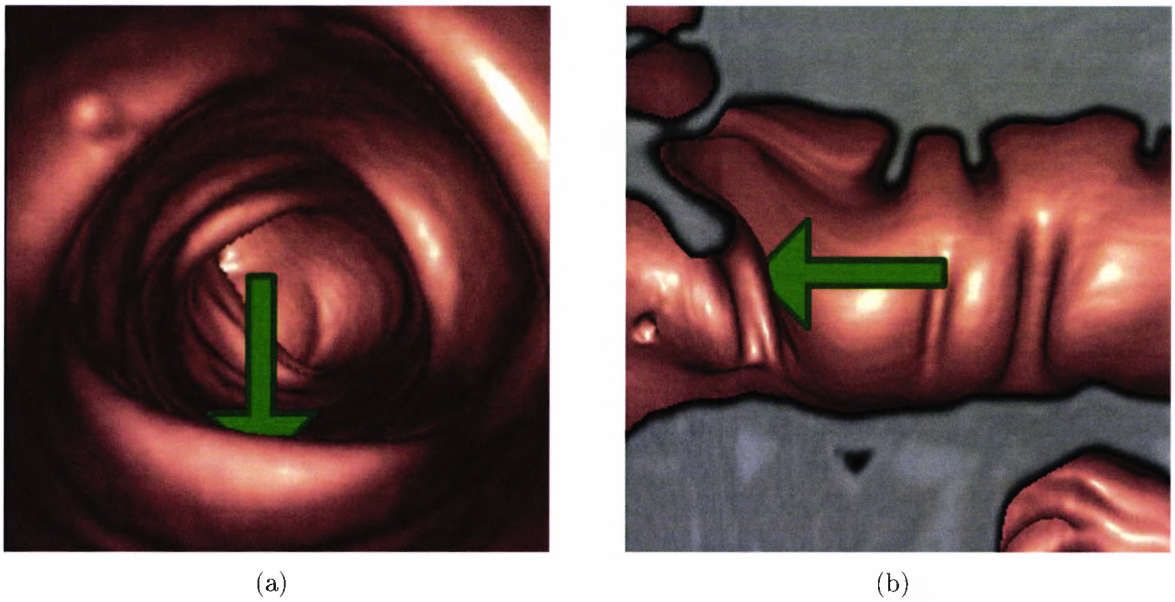

(a) (b)

Figure 9.2: Both the virtual flythrough and Volumetric CPR have regions which can be missed, as indicated by the green arrows in the above picture. Generally the regions missed by each are different, and so combining the two helps to improve overall coverage.

9.3.2.1 Determining the total area

One approach to determining the total area of the colon is to make use of the segmentation information. Segmentation is already performed in our system because we we require a segmented colon in order to compute the centreline. We can consider the total surface area to be given by the number of voxels lying on the boundary of this segmented component;

a voxel is on the boundary if some of its neighbours are in the component and some are not, as defined by a 26-connected neighbourhood.

There are a couple of drawbacks to this approach. Firstly, our segmentation is conservative in that the region marked as colon is slightly smaller than the real size of the colon; this is because we use the segmentation information to skip the rays through empty space and we wish to avoid accidentally skipping them into the wall. The result of this is that all visualisation approaches would seem better than they really are, to the extent that it would actually be possible to get greater than 100% coverage. The second drawback is that if the centreline does not cover some part of the colon then this unfairly penalizes the visualisations (which are not really at fault). This is important in the real system, but for the purposes of our comparison we wish to avoid such external influences and focus on the visualisation themselves.

Instead our solution is to build the total surface as the aggregate of all the different 'seen' surfaces. That is, if a voxel is seen by at least one visualisation then it is considered to be part of the surface. This is slightly unfair because there may be voxels which are not seen by *any* technique because they are hidden by folds or other structures. To compensate for this we perform two morphological dilation operations which expand the surface component but only include voxels which lie on the surface (so as to avoid dilating into the wall). We experimented with using more than two dilations but found no significant increase in area, and so concluded that two was an appropriate number to use.

9.3.2.2 Determining the seen area

The method of measuring the surface area depends on how the visualisation is rendered. In the case of MPR it is difficult to measure the surface area because it is not specifically a surface visualisation technique – it is a general purpose technique which shows *all* the data in the volume. If we step though each slice then at some point we have seen every single voxel; in some ways this could be interpreted as 100% coverage. This is misleading, however, because the actual surface has not been separated from the rest of the data. Therefore, within the context of surface coverage, we do not consider MPR views any further.

For the raycast views (which encompass the flythrough, Volumetric CPR, and colon unrolling) the procedure is relatively straight forward. We create a 'seen' volume, this has the same dimensions as the data volume but only requires 1 bit per voxel. This volume is initially cleared to all 0's to indicate that none of the voxels have been seen.

As images are generated, rays are cast through the volume. Once these rays hit the surface (i.e. at the point at which lighting calculations are performed) the corresponding voxels in the seen volume are set to 1. At the end of the rendering process the seen area is given by the number of voxels which are set to 1.

	Dataset 1	Dataset 2	Dataset 3	Average
Flythrough (Antegrade)	62.2%	69.6%	65.1%	65.6%
Flythrough (Retrograde)	61.9%	66.1%	64.5%	64.2%
Flythrough (Both Directions)	84.8%	89.8%	85.9%	86.8%
Volumetric CPR (Up + Down)	76.1%	82.8%	74.1%	77.6%
Volumetric CPR (Left + Right)	76.3%	84.3%	74.8%	78.5%
Volumetric CPR (All Directions)	97.9%	98.8%	96.5%	97.7%
Flythrough (Both) + Vol. CPR (All)	99.3%	99.6%	98.8%	99.2%

Table 9.2: Surface coverage for different view configurations was measured over three datasets. Adding the Volumetric CPRs to both flythroughs increased surface coverage from 86.8% to 99.2%.

In order to achieve accurate results it is important to ensure the rays do not sample *into* the colon wall. During normal rendering we use the transfer function presented in 6.5 which has a smooth transition from transparent to opaque. This allows a single ray to composite several samples before terminating and so possibly touch on more than one voxel. This is to improve image quality but it has the secondary effect that it artificially increases the number of seen voxels. Hence, for these surface coverage measurements, the transfer function was modified to have a vertical cut off.

9.3.2.3 Comparison

The results of performing these measurements with various combinations of the flythrough and the Volumetric CPR are shown in Table 9.2. We tested the flythrough in both directions because areas obscured behind folds might only be visible in one of these cases. We also performed the measurements with both 2 and 4 Volumetric CPRs. For reasons already mentioned we have not included the MPR, and we have not included the colon unrolling because we do not have access to an implementation.

It can clearly be seen that the Volumetric CPR gives a significant improvement - increasing surface coverage from 86.8% for a bidirectional flythrough to 99.2% when the Volumetric CPRs are added in the average case.

9.3.3 Memory requirements

Given the huge size of modern CT datasets and the limited resources available in commodity PCs, memory consumption can be an important factor in determining whether a particular algorithm is actually practical - regardless of how useful it may be. In this section we analyse the memory consumption of the Volumetric CPR and compare it to alternative approaches. Before we do this, however, we provide a brief review of the way memory is managed in modern PCs.

The amount of memory which a PC is able to access is limited by the width of its *address bus*, measured in bits. Most PCs in use today have a 32 bit address bus, although it is becoming increasingly difficult to actually buy 32 bit machines as sales are being phased out in favour of the newer 64 bit machines. These 64 bit machines have a much greater memory capacity and will therefore solve many of the issues we are about to discuss, but for the time being 32 bit machines are prevalent and are widely used in hospitals.

For a machine with an address bus width w, the maximum amount of memory which can be addressed is 2^w bytes. For a 32 bit machine this gives 4Gb, this is split into 2Gb for the Windows operating system and 2Gb for the users programs. However, even given this upper limit it is not common for hospital computers to actually have this much memory – the Biotronics3D system aims to work with only 1Gb. A CT data set will typically use up several hundred megabytes which means that remaining space can be very limited.

There are two main places where memory is used within a visualisation. Firstly, many visualisations make use of supporting data structures - in the case of raycasting in virtual endoscopy these data structures are often designed to accelerate the ray traversal process. For example, when traversing a volume a ray will usually check the eight corners of a cell before deciding to perform trilinear interpolation to compute a sample. If all eight of the corners are below the surface isovalue then the cell is considered empty and can be skipped. At one point in the development of our system we were making use of an additional volume which stored, for each voxel, a single value indicating whether the corresponding cell was occupied. For a $512 \times 512 \times 512$ volume this required 128Mb of additional memory (although is could be reduced to 16Mb at the expense of access time) and it was decided the memory overhead was not worth paying for the speed boost.

The second place where memory is used is when images are precomputed. This is especially relevant for the Volumetric CPR as we use precomputation to avoid having to generate images during the examination. It is therefore useful to determine how much memory is required to precompute Volumetric CPR images. For each pixel we precompute and store the following values:

Colour: The colour obtained from raycasting or the greyscale value resulting from window-levelling. Stored in RGB form using one byte per channel and giving a total of three bytes per pixel.

Position: The 3D position in the volume corresponding to the pixel. This is used for picking, so that the user can click part of the image and jump to that location. Stored as a 3D vector where each component is two bytes - giving six bytes in total.

Normal: The surface normal at the 3D position corresponding to the pixel. It is primarily used for lighting calculations and it is sufficient to store it using one byte per component, giving a total of three bytes per pixel.

Translucent Colour (Optional): The colour for the pixel in the translucency image, if the translucency image is being used. As with the colour of the main image it is an RGB value using one byte per channel and so a total of three bytes per pixel.

Seen Region Tool (Optional): Each pixel is either seen or unseen, meaning one byte per pixel is sufficient for representation.

9.3.3.1 Comparison

In total, the storage requirements per pixel vary between 12 bytes for the basic Volumetric CPR to 16 bytes when translucency and seen region information is included. A typical Volumetric CPR image is $200 \times 5000 = 1000000$ pixels, and so each precomputed image generally requires between 12Mb and 16Mb. There may be up to four (in the case that we have images for up, down, left, and right) and so the total memory usage is around 64Mb.

Generally this is a price which is worth paying for the high speed which the precomputation allows. However, if the memory is not available the system has to fall back on generating images on demand. In this case the memory overhead is only what is used for any ray acceleration structures and so can be considered equal to that used by the virtual flythrough and colon unfolding approaches.

9.4 Formative Evaluation

The previous section has shown that the Volumetric CPR is a viable technique for the examination of curved tubular structures in medical datasets and exhibits many desirable properties when compared to other visualisations. But while the techniques which have been described allow us to determine quantitative measures of how well the Volumetric CPR functions it is also important to understand the requirements of the user, and how the Volumetric CPR fits into their expected work flow.

The Volumetric CPR implementation has been developed in partnership with Biotronics3D and now forms part of their virtual colonoscopy package. Hence Biotronics3D have been closely involved with this ongoing formative evaluation process and with them we have demonstrated the system and received feedback at a number of locations including:

RNSA 2006: The Radiological Society of North America's annual conference is one of primary events for radiologists, hospital IT staff, and medical imaging companies, with the 2006 conference attracting over 60,000 visitors. The Biotronics3D system was demonstrated to a number of radiologists and health professionals and received very positive feedback.

Hospitals: For the duration of the project we have been liaising closely with several hospitals including University College London Hospitals and Oxford Radcliffe Hospitals. This included an initial requirements gathering meeting with Professor Alastair Forbes who is head of the Centre for Gastroenterology and Nutrition at UCLH, as well as ongoing demonstrations and discussions handled through Biotronics3D.

In this section we describe how the feedback and comments of radiologists made during this formative evaluation process influenced key parts of the systems development.

9.4.1 System Layout

The layout of the various views was decided after RSNA 2006 where extensive consultation with radiologists was carried out. At the time of RSNA 2006 we had the configuration of views shown in Figure 7.1(a). Within this configuration the flythrough could be toggled between looking forwards and looking backwards so that the user could see the back of any folds they passed. We did not consider this feature particularly important as it was also possible to simply pause the camera and turn it around manually, or to perform two flythroughs (one in each direction) as is done on other systems.

In a similar manner, the Volumetric CPR part of the system consisted of two Volumetric CPRs with one looking 'up' and the other looking 'down'. An option was provided to instead make the Volumetric CPRs look 'left' and 'right'. We were aware that there could be surface coverage problems using just two Volumetric CPRs (as explained in Section 6.7) and so we introduced a version using three Volumetric CPRs, having determined that this should cover the whole surface at least once, and half the surface would actually be covered twice. User feedback was as follows:

- Having realised that we could optionally provide a reverse view, several radiologists requested that the reverse view be present at all times. That is, they wanted to be able to use the reverse view without interrupting the work flow of the system.

- Not all users understood the need for both up–down and left–right Volumetric CPRs; believing that just one pair was necessary. Of those who did understand the problem, the majority wanted the system to be modified to optionally support the display of all four Volumetric CPRs simultaneously.

- Generally users did not like the three–way Volumetric CPR because of the lack of continuity between the greyscale parts of adjacent images.

In light of these comments we added a new layout mode as shown in Figure 7.1(b).

9.4.2 Non-Rotating Volumetric CPRs

When we originally created the Volumetric CPR we did so by extending the existing CPR implementation within the system. The original CPR allowed rotation by the user and this feature was naturally carried over into the volumetric version. However, it did not make it into the final version of the system and was removed for two main reasons.

The first reason was technical - as has already been discussed there were difficulties getting the system to run at a reasonable frame rate and so it made sense to precompute the images. This would not have been a problem if the Volumetric CPR had been running in isolation, but instead it was running along side other CPU intensive visualisations.

The second reason was that user feedback revealed that radiologists were not comfortable with rotating the Volumetric CPR. Partly they found it disorientating (something we had most likely not found due to understanding how the images were generated) and partly they were concerned that areas could be missed by not doing the rotation correctly. Providing images which only scroll simplified user interaction with the system as they were able to simply sit back and watch until they encountered something of interest.

9.5 Summative Evaluation

The consultation mentioned above was part of an ongoing process of requirements gathering, analysis, and implementation. After the PhD work was completed we also performed some summative evaluation by showing the system to two doctors who had not seen it before. Dr Sherif Sadek looked at our system as a radiologist screening for polyps, while Dr Nick Croft provided information on how it could be extended for our longer term goal of IBD detection.

9.5.1 Radiological Perspective

Dr Sherif Sadek is a consultant radiologist from Whipps Cross hospital. The feedback session took place at City University and consisted of a demonstration of the system, with questions asked at various points during the demonstration. Dr Sadek was also given a chance to use the system directly in order to get a feel for how it compared to other solutions. In this section we discuss the key points which were raised.

Dr Sadek had general experience in radiology and had worked with CT colonoscopy datasets in the past. These included both prone and supine datasets in which contrast agent had been used to enhance the visibility of material in the colon. He had used mostly a slice by slice approach to analysing the datasets, primarily because this was the only option available through the software he used at the time.

He also had exposure to 3D visualisation for cranial vascular imaging while searching for cranial aneurysms. This is not directly related to our colonoscopy work, but it is worth noting that we are considering vascular screening as another potential application of the Volumetric CPR technology (see Section 11.4) and so he had some understanding of the issues which needed to be addressed for this field. He had also seen the virtual flythrough approach to colonoscopy in the past but had not used the systems himself.

The system was demonstrated using the same dataset which has been used for many of the images in this thesis, such as Figure 8.6. The system was started up in a layout which did not include the Volumetric CPR; Dr Sadek noted that the general feel of the system at this point was quite similar to many other systems he had used. We then switched to the layout which included two Volumetric CPRs and gave a short demonstration of how they were used.

Dr Sadek said he felt comfortable using the system; the overview made it clear where he was in the dataset and, although it was difficult to judge orientation, he was able to mentally register the different views. Hence he did not feel disoriented or see any immediate problems with using the system for a prolonged period of time. He also felt the Volumetric CPR was a useful tool for reducing disorientation because it was more static than the flythrough.

The translucency window was considered to be a useful tool for reducing false positives, but he was particularly pleased with the surface coverage tool as he could immediately see that it reduces the chance of polyps being missed. The combination of both tools at the same time was found to be confusing as it simply presented too much data at a time.

Probably the biggest point (which was raised several times) was that Dr Sadek always wanted to be able to refer back to the base data (using MPR) rather than relying exclusively on 3D reconstructions. This may be partly as a result of having been trained to use these 2D views and not having training in 3D, but it also matches more general feedback which has been obtained via Biotronics3D. Clinicians do trust the system, but still feel safer if the original 2D data is also present. Of course, this is not a problem because it is perfectly possible to display 2D and 3D views in the same layout.

Overall he was happy with the Volumetric CPR and felt it would improve examinations. He claimed the Volumetric CPR could help reduce uncertainty, but that the user must be aware of the limitations of the system or it could be falsely reassuring.

9.5.2 IBD Perspective

Dr Nick Croft was shown the Volumetric CPR technique in order to better understand how it can be adapted for the purpose of IBD visualisation. Dr Croft's expertise is in the field of detecting IBD using more traditional optical colonoscopies and so he was extremely interested to see how an automated system could improve the process. Having been shown the approach he stated that it could already be useful for IBD, and had a lot of potential. In terms of improvements he made three main points:

- For Crohn's disease it is useful to measure the thickness of the bowel wall in inflamed regions. We have not implemented a measurement tool in our Volumetric CPR but it would be possible to do so, and is relatively easy if measurements are restricted to the vertical direction where distortion does not occur. For Colitis this is not so useful as the inflammation occurs in different tissue layers.
- It is important to screen the small bowel as well as the large bowel. The Volumetric CPR has no limitations regarding the size of the structures it can examine (in Section 11.4 we apply it to blood vessels) so this really becomes a problem of performing the segmentation.
- Clinicians want to know how far into the colon the disease extends. This information is useful for determining followup treatment, and could be easily obtained from our system as we provide similar information for recording the location of polyps.

Section 11.2 provides more information about our ideas for extending the Volumetric CPR in the direction of IBD visualisation.

9.6 Summary

Evaluation of a system is important in order to verify that it performs the task for which is was designed in an efficient and effective manner. In health care it is often difficult to judge what effect a new system will have on existing work flows and practices, but in this chapter we have identified the key variables which we believe differentiates our approach from the others. We have also presented both the summative and formative aspects of the evaluation, including feedback from clinicians and conferences as well as a direct one–on–one demonstration of the system to a radiologist. This has verified that the system is useful for both the screening for polyps, and potentially as a method of identifying IBD.

The next stage, to be considered future work, would be to carry out a more complete clinical trial using some of the approaches described by Friedman *et al* [77]. The aim would be to directly compare an optical colonoscopy with our virtual approach while measuring parameters such as the polyp detection rate, the number of false positives, and the examination time. Once the remaining components of the IBD system are in place, a similar study on the effectiveness of the IBD detection can be carried out.

Chapter 10

Conclusion

Virtual colonoscopy has been an active research area for many years, and is widely used in clinical practice, but there are still many questions which must be answered before it can be expected to fully replace the optical approach. In this thesis we have addressed the problem of visualisation; that is how to display the huge amount of data to the user in such a way that they can interpret it easily and completely. The work forms part of a larger project on the automatic detection of IBD, but because we do not yet have all the required components for this application we have instead been developing primarily in the context of screening for colon cancer.

We have presented the *Volumetric CPR* as a possible solution to this visualisation problem, and as an improvement over existing approaches such as the virtual flythrough. The Volumetric CPR extends the concept of conventional CPR [46] with volume rendering to make it more useful for examining the surface of tubular structures within the body. The greyscale CPR provides context information about what lies beyond the immediate surface, while the new volume-rendered component displays the surface of the structure in vivid 3D so that polyps can be easily identified.

We began by describing the image generation process for Volumetric CPR, and specifically the way in which each of the existing CPR projection techniques (projected, stretched, and straightened) can be extended. We described the effect which each projection mode has on the distortion in the final image, on the presence of occlusion, and on the ability to perform measurements. We also presented some approaches to reducing the area which was raycast and concluded that performing region growing in image space was the most effective method. This work formed our first journal publication in the IEEE Transactions on Visualization and Computer Graphics [75].

Having established how images were generated we then looked at how they could be effectively used for the purpose of colon screening. We first observed that even the basic Volumetric CPR image has some significant advantages over the existing virtual flythrough approach. These advantages correspond to the aims of the research which were presented in Sections 1.3 and 5.5 and include:

- An ability to see further ahead and behind.
- An ability to see behind folds and into regions which are typically obscured during a flythrough.
- Seeing more of the surface at any given time; hence each part of the surface is visible for longer.
- Making it easy to recognize when contrast enhanced fluid is present and allowing the user to view behind it.

Secondly we introduced the concept of overlays as a method of providing additional information to the user. We used this concept to implement *translucency rendering*; a technique pioneered by the Viatronix system [58, 62]. Integrating translucency rendering within the Volumetric CPR allowed for a smoother work flow because there was no need to keep toggling it on and off as is done in a flythrough. It also ensured that each part of the surface passed through the translucency window at some point.

Overlays also provided an effective means of visualising surface coverage information; this is important because insufficient surface coverage has been one of the main arguments used against virtual colonoscopy. By modifying the flythrough to record all parts of the surface which it sees we were able superimpose this information onto the Volumetric CPR via an overlay. This overlay is updated in real time so that the user is immediately aware that they have missed a region. This work on the usage of the Volumetric CPR for colon screening formed the majority of the work for a paper [76] published at the conference for Computer Aided Radiology and Surgery (CARS 2007).

We next moved on to examine lighting within the context of the Volumetric CPR. We established two basic models; a 'local' model in which the centreline was treated as a single long strip-light, and a global model in which a point light source could be moved along the centreline. Within the 'local' model we provided two approaches to computing the ray direction (positional and directional) which resulted in noticeably different images. The advantage of the 'global' method was that it provided addition cues to the structure of the surface because the user could witness the shading changing as the light was moved around.

We also used lighting as a method of conveying shape information about the structure which was being examined; we called this technique *shape driven lighting*. The idea behind this was to allow properties of the shape (such as direction or curvature) to influence the lighting parameters. We chose to influence the colour of the light though it would also be possible to influence other parameters such as the specular reflectance. The technique is useful because it can improve the user's spatial awareness and also alert then to the presence of distortion in certain parts of the colon. This lighting related work formed our third paper [78] which has not yet been published.

Lastly we have presented a preliminary evaluation of our work. This included a technical section in which we showed that the use of the Volumetric CPR significantly improves the surface coverage of a virtual colonoscopy system (from 86.8% to 99.2%) as well as demonstrating favourable image generation times. It also included a clinical section in which we presented the results of our discussions with radiologists and users in which we determined that the system has real benefits for the virtual colonoscopy process in terms of improving confidence and reducing examination time.

Overall this research has been a success and has resulted in:

- 2 Journals Papers
- 1 Conference Paper
- 1 Patent Application
- 3 Posters (including 1 'best poster' award)
- 1 Invited talk in Vienna

Furthermore, we can see many opportunities for the research to be continued in new and interesting directions. Several of these areas of future work are discussed in the following chapter.

Chapter 11

Future Work

11.1 Introduction

Within this thesis we have shown the Volumetric CPR applied to the relatively small problem domain of virtual colonoscopy visualisation, but it is really a much more general purpose technique than that. Actually it can be applied to any kind of curved tubular structure in any kind of volumetric data set meaning it has applications outside the medical domain. However, in the short term it is most likely that we will continue to apply it to medical problems due to existing expertise in this area. This chapter presents some of our future ideas for applying and improving the Volumetric CPR.

In this chapter we present two addition medical visualisation uses for the Volumetric CPR. The first is the visualisation of IBD data - as has already been mentioned this was one of our aims for the PhD project but it has not been tested because we do not yet have real IBD data available. Instead we demonstrate how it might look using artificial data. The second is the visualisation of the aorta - it is a common problem that radiologists need to screen the aorta for the presence of calcifications or for the purpose of preoperative planning.

We also describe a method for doing computer aided polyp detection on the CPR image via the Hough transform, and we have some preliminary images to show how this might work. If it proves to be feasible then it may be more efficient than doing polyp detection on the volume because it would reduce a three dimensional problem to a two dimensional one.

We consider other types of information which could be displayed on the Volumetric CPR, and give computation fluid dynamics data as an example of something which could be useful (again in the context of the aorta). We mention this example because a project to obtain this data is currently in the early stages.

Lastly we look at ways in which the image generation process could be improved. This includes the possibility of using dedicated hardware for the rendering, or of generating separate left-eye and right-eye images in order to use them on a stereoscopic display.

11.2 IBD visualisation

In Chapter 1 of this thesis we explained that this PhD work forms part of a larger project to ease the diagnosis of Inflammatory Bowel Disease (IBD) from volumetric data such as a CT or MRI scan. However, because the PhD work has focused on the visualisation component of the system, and because visualisation is the last phase of the pipeline as defined in Figure 1.1, we have been unable to really apply the technology to its intended purpose. In this section we look forward to the point at which the rest of the pipeline has been implemented and analyse how the Volumetric CPR can be used for the visualisation of its output.

There are really two aspects to this. Firstly we look at the concept of *abnormality based transfer functions* as a method of conveying the health of a region of tissue to the user. This extends and modifies the concept of transfer functions which was introduced in Chapter 4. Secondly we look at ways in which the greyscale context information can aid IBD diagnosis by indicating the presence of fistulae and wall-thickening.

11.2.1 Abnormality Based Transfer Functions

As they have been used so far, transfer function provide a way of mapping greyscale density values onto $\{R, G, B\}$ triplets so as to obtain an artificial colour image. The idea behind an *abnormality based* transfer functions is to map a function of the degree of abnormality to an $\{R, G, B\}$ triplet instead of the density value. This degree of abnormality would be the result of the previous pipeline stages. For example, if we assume that the classifier stage can assign to each voxel a value in the range 0.0 to 1.0 then we can map this value to a colour ranging from green (for healthy tissue) through yellow and onto red (for unhealthy tissue). This is similar to the approach we used in Section 8.5.2 when mapping curvature to surface colour.

An example of the kind of image we might expect is shown in Figure 11.2.1. Note that this image is entirely artificial and was not generated from real IBD data – it is merely intended to show how the result might look.

It is also expected that the alpha component and compositing strategy may be different for an abnormality based transfer function. Because IBD affects the tissue just behind the surface it may not be sufficient to use a standard isosurface raycaster as has been used for the purpose of polyp detection. Instead it will be necessary to ensure that the ray traverses and samples a short distance (3-5mm) into the wall and the alpha component of the transfer function will most likely be independent of degree of abnormality. Further more, rather than using a compositing approach like that described in Section 4.4.3 it

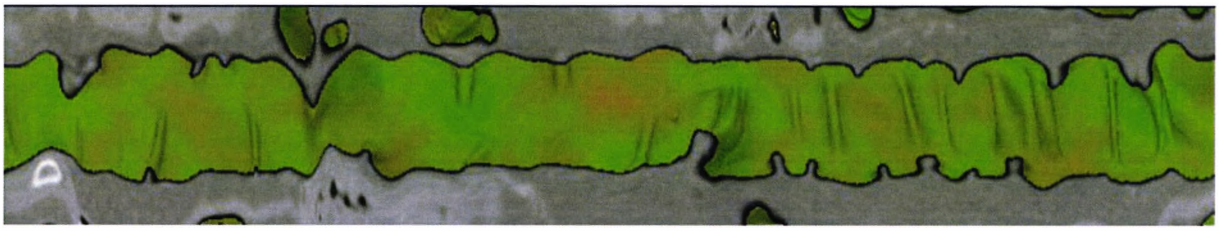


Figure 11.1: A simple proof-of-concept image for how the Volumetric CPR might look once IBD data is placed on it. Green represents healthy tissue while the patches of red represent infected tissue.

is likely that the samples would be averaged. Alternatively, the maximum sample value could be found (similar to MIP; Section 4.4.1) as this would ensure unhealthy tissue is still visible even if it is obscured by healthy parts.

11.2.2 Using the context information

Within the Volumetric CPR the context information refers to the greyscale regions at the top and bottom of the image. These show structures beyond the surface of the colon and have not been used extensively for the purpose of polyp detection (although in Section 8.3 we highlighted their use for screening behind fluid or in blocked regions). When working with IBD data, however, this context information has the potential to be extremely useful.

The presence of IBD causes tissue inflammation and, in turn, a thickening of the colon wall. This wall-thickening is an important indication of the presence of IBD; we are informed by radiologists that it can be difficult to spot on CT scans due to the limited resolution and inability to differentiate tissue types, but it is much easier to identify in MRI data. However, visualising this data in a flythrough is not trivial as it would require the outer wall of the colon to be segmented. This would then allow a thickness calculation to be performed which could be used as the input to a transfer function. The accuracy of the presented information would depend on the accuracy of the segmentation – something which may be hard to perfect.

The advantage of the Volumetric CPR in this scenario is that the thickness of the colon wall can be determined by looking at the 'raw' greyscale data which comprises the CPR component of the image. This could be a standalone method or it could be combined with a thickness based transfer function, perhaps so that the transfer function could identify regions of concern which can then be verified by rotating the view and examining them in the context information.

A second use for the context information is for the visualisation of the *fistula* which are often associated with IBD. Fistula are small tunnels or passageways which form in the body and connect together organs which should not normally be connected (alternatively they may connect an organ to the skin). Because they exist outside the colon they are again difficult to spot using virtual flythrough techniques, but the surrounding context information in the Volumetric CPR could be a powerful way to identify them.

11.3 Polyp Recognition in the Volumetric CPR

Computer Aided Detection (CAD) of polyps in the colon is an active research area which aims to aid clinicians by automatically identifying suspicious structures. Accurate identification can shorten screening times and improve diagnosis by immediately directing users to areas of concern. There has been significant academic research [59, 79, 80] on the topic and it has also been included in commercial virtual colonoscopy packages [81].

Approaches to the problem typically involve analysing the raw CT data. They usually begin with a surface extraction stage (to avoid wasting processing power analysing parts of the volume which do not lie on the colon wall). Then some form of 3D shape analysis is performed, for example to identify spherical structures or to determine the curvature at all points of the surface. Lastly a classifier must be applied to determine which characteristics are indicative of polyps.

If successful, the results of such CAD could easily by superimposed on the Volumetric CPR via overlays (as presented in Chapter 8). However, we are more interested in whether the polyp detection can be done in *image space* rather than in the volume, and whether there are any benefits from doing so (as image space operations might be expected to be faster). We have written a simple proof–of–concept system to test some of our ideas (See Figure 11.2) but to properly research and test the techniques will take many months.

Our system begins by detecting edges in the input image using one of the various image filters provided by the ITK toolkit [82]. The detected edges often fall in the context information part of the image – polyps do not form here because it is beyond the wall of the colon. A mask is therefore used to suppress these edges before a thresholding operation is applied.

Circle detection is now performed on the image using the Hough transform as provided by ITK. This gives the final image in Figure 11.2 where it can be seen that the polyp has been marked with a high intensity white dot. However, it should be noted that this image was only achieved after a lot of experimenting with parameters at different stages of the system, and after manually specifying the radius of the circle the system was looking for. The input image was also fairly clear; other polyps might be located on the side of folds or in other difficult to identify locations. Further research will reveal whether image space recognition of polyps in the Volumetric CPR is really a viable alternative to current solutions.

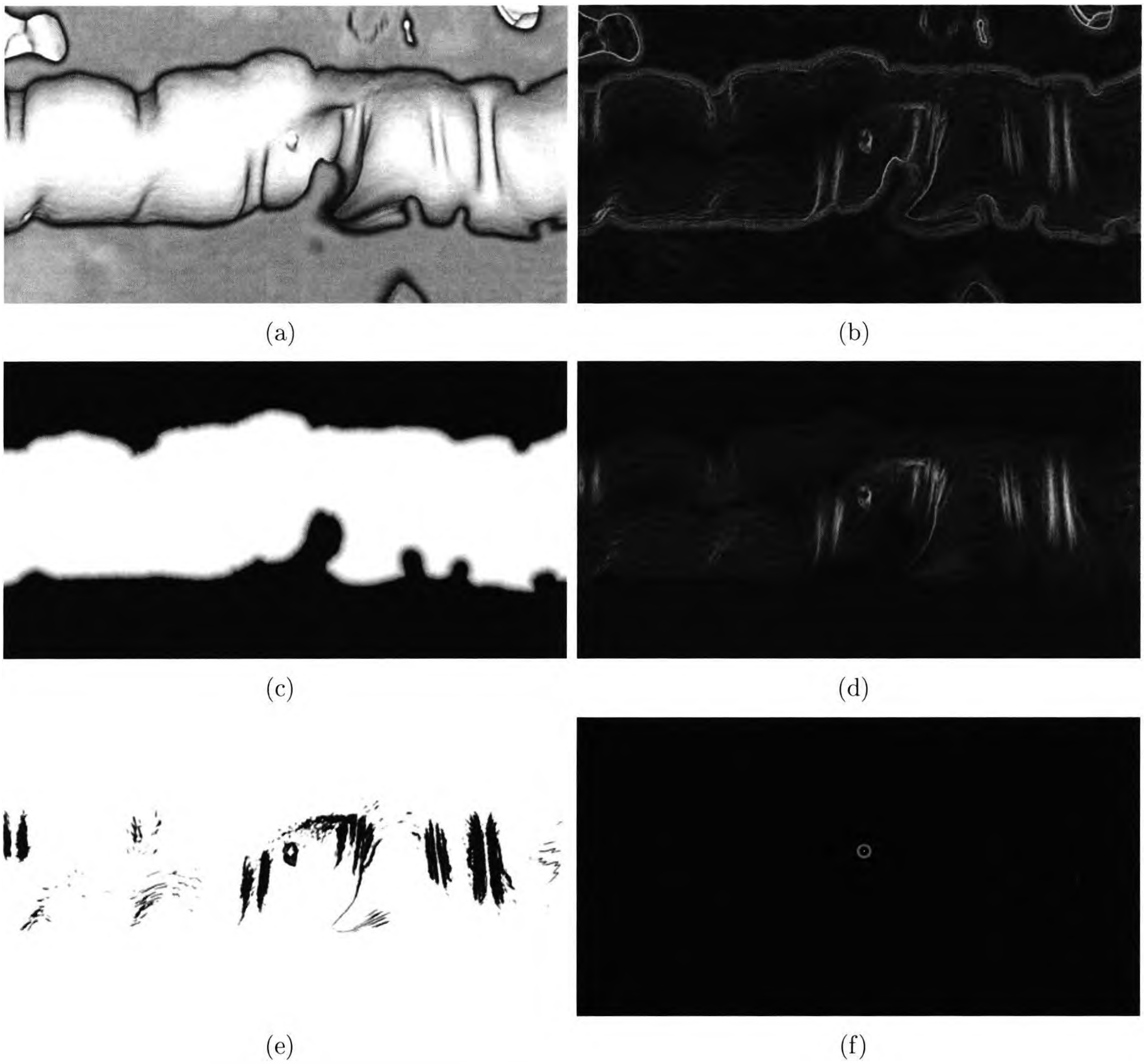

(a) (b) (c) (d) (e) (f)

Figure 11.2: Our prototype implementation took the input image (a) and applied a Sobel edge detection image filter (b). To avoid finding polyps outside the colon a mask (c) was applied to give the image in (d). Thresholding was applied (e) and the result was processed using the Hough transform for circle detection. In the final image (f) a bright spot can be seen at the location of the polyp (ringed in red).

11.4 Volumetric CPR for CT Angiography

Although we have presented our new technique within the context of colon screening (whether it is for colon cancer or IBD) the Volumetric CPR is actually far more general purpose than that. We can use it for any tubular structure in the body, and another good example of its flexibility is found by applying it to the problem of CT Angiography.

CT Angiography involves the examination of the vascular system; typically for the purpose of identifying calcifications or for doing preoperative planning for the placement of a *stent*. The process usually begins with a contrast agent being administered to the patient in order to cause the vascular system to show up bright white on the upcoming CT scan (a scan without contrast agent may also be performed for segmentation and registration purposes). Next the patient undergoes a CT scan which in many cases will cover the full length of the body. The resulting data is commonly visualised using convention CPR or a MIP volume rendering, but because the vascular system is a tubular structure it is also possible to apply the Volumetric CPR.

The vascular system differs from the colon in that it is a 'tree' structure rather than being a single long tube. In its current state the Volumetric CPR will not deal with this directly, but it is possible to compute a single path through the tree and apply the algorithm to that. Future versions of the algorithm could incorporate the technique presented by Kanitsar *et al* [83] which allows an entire vessel tree to be flattened while also ensuring that occlusion does not occur when the visualisation is rotated.

There is also a difference in the range of density values which is encountered in CT angiography. In the colon everything below a certain threshold was considered to be empty space while everything above the threshold was considered to be tissue. Hence the simple transfer function shown in Figure 6.5 was sufficient to produce the desired images. The inside of a vessel, however, is actually *higher* than the surrounding tissue (due to the contrast agent). Furthermore it is desirable to identify *three* different density ranges (rather than the two with exist in the colon) because we wish to visualise tissue, contrast agent, and calcifications.

We modify the transfer function to that shown in Figure 11.4. The middle density range is the contrast fluid marking the interior of the vessel and so we set the opacity to zero so that the rays will skip over this region. If they encounter a lower density surface it is tissue and is shown in red, whereas a higher density surface is a calcification and is shown in white. There is a fairly sharp cutoff as we wish to visualise an isosurface.

We have implemented a proof–of–concept version of the approach described above, and the results can be seen in Figure 11.4. Compared to the CPR, our new version has the advantage that calcifications can be identified with less need for rotation, and also that the user is able to get a real sense of its 3D shape.

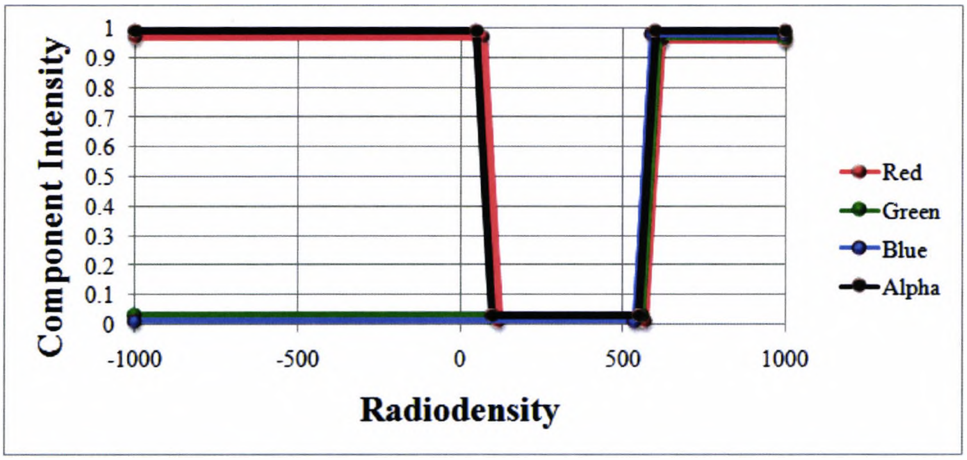


Figure 11.3: A transfer function which can be used to generate Volumetric CPR images of the aorta, as shown in Figure 11.4. The dip in the middle of the transfer function corresponds to the contrast agent which we do not want to see in the final image.

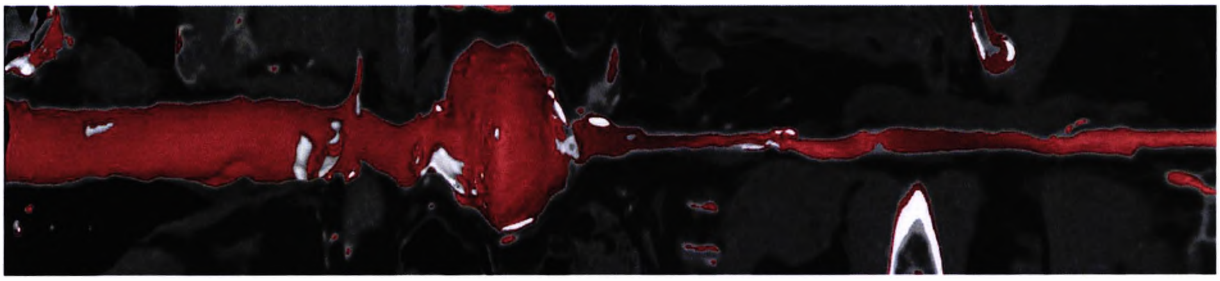


Figure 11.4: A Volumetric CPR of the aorta; the largest artery in the body. Calcifications can clearly be seen in white while the widening of the vessel in the centre of the image is an aneurysm.

11.4.1 Flow Visualisation

In Chapter 8 we presented two methods of conveying additional information via the Volumetric CPR. The first of these made use of overlays to blend information with the underlying Volumetric CPR image, while the second approach modified the colour of the surface directly by changing the shading algorithm to incorporate additional information. In both cases the final result was to allow us to modify the colour of the Volumetric CPR image to convey information such as surface coverage, surface density, or curvature.

Of course, these techniques can be used to display any information which can be computed for points on a structure's surface. One type of information we are particularly interested in visualising is the results of Computation Fluid Dynamics (CFD) simulations carried out in the vascular system. These calculations can be used to compute blood pressure and flow rates; crucial information when undertaking preoperative planning for vascular surgery.

Biotronics3D is in the early stages of initiating a collaboration which will see an academic partner become involved in such CFD computations on the vascular system. It is anticipated that the academic partner will perform complex CFD calculations on a number of vascular datasets, with each dataset taking potentially several weeks to compute. This is of course impractical for real applications, but it is expected that from these precomputed datasets it will be possible to interpolate the values of a new dataset to within a certain degree of accuracy. This interpolation process should take just a few minutes, allowing easy application to real world problems.

11.5 Hardware Accelerated Image Generation

Over the last 10 years, the appearance and evolution of dedicated *graphics processing units* (GPUs) has revolutionized the field of computer graphics research. These GPUs are designed to excel at the kind of highly parallelizable operations which are common place in computer graphics - operations such as matrix transformations, lighting calculations, and scanline rasterisation. Their rapid evolution has largely been driven by the games industry and its continuous demand for ever more realistic graphics. In this section we begin by describing the basics of how such graphics hardware operates, before describing how it can be used for volume rendering and, more specifically, the Volumetric CPR.

It should be noted from the start that most graphics hardware is designed for rendering *polygon* graphics rather than *volume* graphics. There are some exceptions to this rule - notably the VolumePro board from TeraRecon, Inc [84] - but these tend to be highly specialized and do not have the flexibility to be adapted to Volumetric CPR. However, although most graphics hardware is designed for polygon rendering there has been a significant amount of research into adapting them to volume rendering. This has been

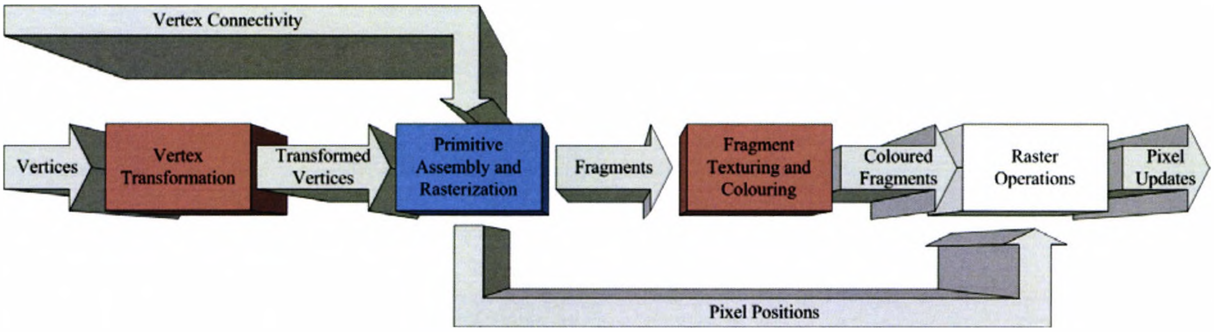


Figure 11.5: The graphics pipeline as implemented on most GPUs. The stages in red are programmable on most GPUs whereas the blue stage is only programmable on the latest NVidia GeForce 8 series. The white stage is generally not programmable.

getting easier with each generation of GPU as they become more flexible and general purpose.

The graphics pipeline as implemented in most graphics hardware is shown in Figure 11.5. It has been an increasingly popular trend to allow stages of the pipeline to be programmable via custom *shaders*; these are pieces of code which can be loaded onto the GPU in order to achieve certain special effects. It is the availability of these shaders (particularly fragment shaders) which has now allowed raycasting to be implemented on the GPU.

The process would involve loading the CT data into the graphics card memory as a 3D texture. The centreline positions, normals, bi-normals, and tangents would be computed by the CPU, stored in a texture, and loaded onto the graphics card for later access by the shaders. This would only have to be done once because the values do not change between frames. However, the users position and angle of rotation do change between frames and so these would have to be passed in each frame as parameters to the shader. The majority of the logic would then be implemented in a fragment shader; this would check whether the current position was above the threshold in order to determine whether to sample the volume or cast a ray. The ray casting can then be implemented using the new looping capability of Shader Model 3 which is present on recent graphics cards.

Typically GPU implementations are several times faster than those done on the CPU. If this also applies to the Volumetric CPR then it will allow smoother interaction and also free the CPU to perform other tasks.

11.6 Stereoscopic Image Generation

Throughout this thesis we have referred to the process of generating '3D images' of the colon or other body parts. In this section we note that this is actually an inaccurate term (although a widely used one) because the resulting images are actually two dimensional and are designed to be displayed on two dimensional media such as a computer monitor or printed on paper. What we are *actually* generating is an image which is intended to *look* three dimensional via the use of techniques such as perspective, occlusion, and shading. These are known as *monocular depth cues* because they are available from just one eye.

In practice the human visual system makes use of *binocular depth cues* and, in particular, *retinal disparity*. This is a depth cue which arises from having two different images of the same object, taken from slightly different angles due to the different position of the two eyes. The human visual system is then able to use triangulation to determine the depth of the object. Most displays do not allow retinal disparity to be used as a depth due because both eyes are seeing the same image.

However, the development of stereoscopic displays is currently a very active research area. These include basic approaches making use of red–green or polarized glasses, lenticular displays which split the images and projects them in different directions, and advanced approaches which make use of head tracking technology to project a image at a given eye position. Regardless of the underlying technique which is used, the aim is always to provide the two eyes with different images so that retinal disparity can be exploited.

Before discussing the generation of these images it is important to establish why stereoscopy can be beneficial to the Volumetric CPR. As was explained in Section 8.4.3 it can be difficult to determine whether a particular structure on the surface is sticking 'in' or 'out' due the orthographic projection which is used when generating the Volumetric CPR. This was part of the motivation behind introducing the global lighting model, as the way that light changes across the surface can help determine its structure, but the use of stereoscopic displays can also reduce this problem.

The remaining question (and the likely focus of future research) is then how do we generate the two images for the left and right eye? In many scenarios (such as when rendering a flythrough) it is sufficient to simply move the camera left and right. However, this approach does not work when using our effectively–orthographic projection because the two images are identical but offset from each other. This produces a meaningless image with no additional depth information. Instead we believe it will be necessary to adjust the ray direction between the two images to give the impression that they are 'focusing in' on each other. This is illustrated in Figure 11.6.

We have already implemented stereoscopic image generation for the virtual flythrough component of the Biotronics3D system (see Figure 11.7), as well having a partial implementation of the Direct Volume Rendering component. This means much of the infrastructure is already in place to support stereoscopic rendering - it is just a case of actually generating the two Volumetric CPR images.

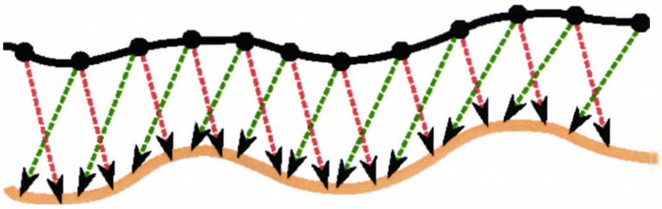
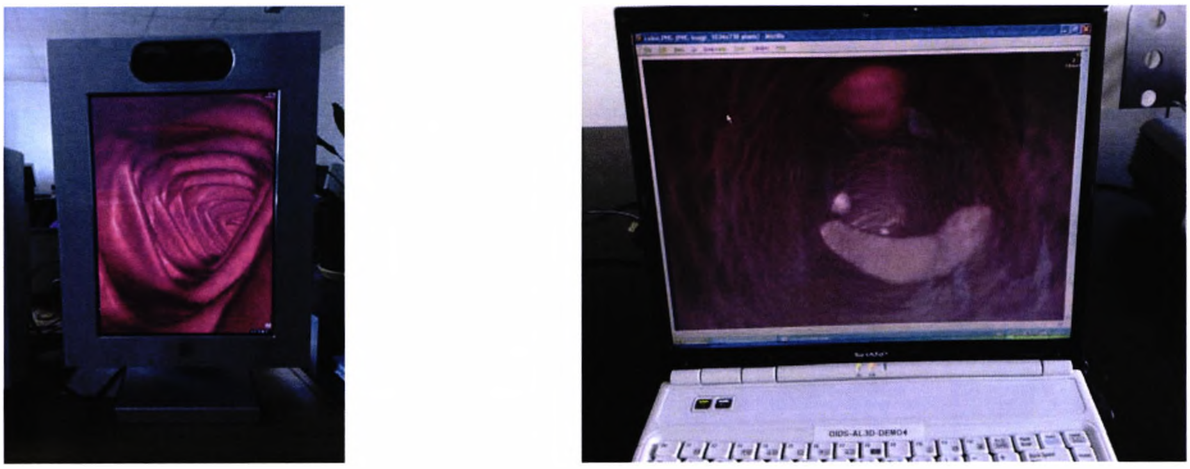


Figure 11.6: The stereoscopic Volumetric CPR will be generated by adjusting the ray directions between each eye (shown in red and green).



(a)

(b)

Figure 11.7: A stereoscopic display (a) showing a 3D rendering of the flythrough. This particular model includes a head tracker to ensure that the 3D image is maintained as the user moves their head. Similar stereoscopic displays are also available in laptops (b) though in this case without the head tracker. It is anticipated that we could create stereoscopic Volumetric CPRs for these displays.

11.7 Summary

This chapter has served to illustrate that there is a significant amount of future work which can be carried out on the Volumetric CPR. This includes both modifications to the underlying algorithm and application of the technique in new and interesting ways. Depending on the work undertaken it could be suitable for MSc, PhD, or post doctorate levels of study.

Bibliography

[1] B. Crohn, L. Ginzburg, and G. Oppenheimer, "Regional ileitis; a pathologic and clinical entity," *Journal of the American Medical Association*, vol. 99, pp. 1323–1329, 1932.

[2] NHS Direct, "Ulcerative colitis," http://www.nhsdirect.nhs.uk/articles/article.aspx?articleId=381.

[3] P. Orsoni, S. Berdah, C. Verrier, A. Caamano, B. Sastre, R. Boutboul, J. Grimaud, and R. Picaud, "Colonic perforation due to colonoscopy: a retrospective study of 48 cases," *Endoscopy*, vol. 29, pp. 160–164, 1997.

[4] D. Nicholson, "Virtual colonoscopy versus fibre optic colonoscopy," Department for Health Research Finding Register, 2002.

[5] S. Jackson and R. Thompson, *Cross-Sectional Imaging Made Easy*. Churchill Livingstone, August 2004.

[6] L. Hong, S. Muraki, A. Kaufman, D. Bartz, and T. He, "Virtual Voyage: Interactive Navigation in the Human Colon," in *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques*, 1997, pp. 27–34.

[7] S. Lakare and A. Kaufman, "Light weight space leaping using ray coherence," in *Proceedings of IEEE Visualization*. Washington, DC, USA: IEEE Computer Society, 2004, pp. 19–26.

[8] M. Wan, Q. Tang, A. Kaufman, Z. Liang, and M. Wax, "Volume Rendering Based Interactive Navigation within the Human Colon." in *Proceedings of IEEE Visualization*, 1999, pp. 397–400.

[9] M. Wan, W. Li, K. Kreeger, I. Bitter, A. Kaufman, K. Liang, D. Chen, and M. Wax, "3d virtual colonoscopy with real-time volume rendering," *SPIE Medical Imaging*, vol. 3978, pp. 165–171, 2000.

[10] M. Wan, A. Sadiq, and A. Kaufman, "Fast and reliable space leaping for interactive volume rendering," in *Proceedings of IEEE Visualization*, 2002, pp. 195–202.

[11] S. You, L. Hong, M. Wan, K. Junyaprasert, A. Kaufman, S. Muraki, Y. Zhou, M. Wax, and Z. Liang, "Interactive Volume Rendering for Virtual Colonoscopy," 1997, pp. 433–446.

[12] National Electrical Manufacturers Association, "Digital Imaging and Communications in Medicine (DICOM)," 2007. [Online]. Available: http://medical.nema.org/dicom/2007/

[13] V. Cerf, Y. Dalal, and C. Sunshine, "Specification of internet transmission control program," United States, 1974.

[14] S. Grimm, S. Bruckner, A. Kanitsar, and E. Gröller, "Memory efficient acceleration structures and techniques for cpu-based volume raycasting of large data," in *Proceedings of IEEE/SIGGRAPH Symposium on Volume Visualization and Graphics*, 2004, pp. 1–8.

[15] S. Grimm, S. Bruckner, A. Kanittsar, and E. Gröller, "A refined data addressing and processing scheme to accelerate volume raycasting," *Computers & Graphics*, vol. 28, no. 5, pp. 719–729, 2004.

[16] M. Sato, S. Lakare, M. Wan, A. Kaufman, Z. Liang, and M. Wax, "An automatic colon segmentation for 3d virtual colonoscopy," *IEICE Transactions on Information and Systems*, vol. E84-D, no. 1, pp. 201–208, 2001.

[17] Y. Ge, D. Stelts, J. Wang, and D. Vining, "Computing the centerline of the colon: a robust and efficient method based on 3d skeletons," *Journal of Computer Assisted Tomography*, vol. 23, pp. 786–794, 1999.

[18] S. Lakare, D. Chen, L. Li, A. Kaufman, and Z. Liang, "Electronic colon cleansing using segmentation rays for virtual colonoscopy," *Proceedings of SPIE Symposium on Medical Imaging*, vol. 4683, pp. 412–418, 2002.

[19] L. Li, D. Chen, S. Lakare, K. Kreeger, I. Bitter, A. Kaufman, M. Wax, P. Djuric, and Z. Liang, "Image segmentation approach to extract colon lumen through colonic material tagging and hidden markov random field model for virtual colonoscopy," in *Proceedings of SPIE Symposium on Medical Imaging*, 2002.

[20] C. Wyatt, Y. Ge, and D. Vining, "Automatic segmentation of the colon for virtual colonoscopy," *Computerized Medical Imaging and Graphics*, vol. 1, no. 24, pp. 1–9, 2000.

[21] L. Vincent and P. Soille, "Watersheds in digital spaces: An efficient algorithm based on immersion simulations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, pp. 583–597, 1991.

[22] R. Haralick and L. Shapiro, *Computer and Robot Vision.* Addison–Wesley, 2005.

[23] R. Haralick, "Statistical and structural approaches to texture," *Proceedings of the IEEE*, vol. 67, no. 5, pp. 786–804, 1979.

[24] R. Gonzalez and R. Woods, *Digital Image Processing*, 3rd ed. Prentice Hall, 2007.

[25] M. Tuceryan and A. Jain, "Texture analysis," in *The Handbook of Pattern Recognition and Computer Vision*, C. Chen, L. Pau, and P. Wang, Eds. World Scientific Publishing Co, 1998, ch. 2.1.

[26] B. Mandelbrot, *The Fractal Geometry of Nature.* W. H. Freeman, 1983.

[27] R. Duda and P. Hart, *Pattern Classification and Scene Analysis.* John Wiley & Sons, 1973.

[28] D. Pelleg and A. Moore, "Accelerating exact K-means algorithms with geometric reasoning," in *Proceedings of the Fifth International Conference on Knowledge Discovery in Databases*, 1999, pp. 277–281.

[29] D. Pelleg and A. Moor, "X-means: Extending K-means with efficient estimation of the number of clusters," in *Proceedings of the Seventeenth International Conference on Machine Learning.* Morgan Kaufmann, 2000, pp. 727–734.

[30] R. Weil, D. Modney, G. Diana, and S. Hiss, "Real-time window/leveling on a radiographic workstation," United States Patent and Trademark Office, September 1995.

[31] E. Caoili and E. Paulson, "CT of small-bowel obstruction: another perspective using multiplanar reformations," *American Journal Of Roentgenology*, vol. 174, no. 4, pp. 993–998, 2000.

[32] K. Ibukuro, C. Charnsangavej, M. Chasen, and A. Cinqualbre, "Helical ct angiography with multiplanar reformation: Techniques and clinical applications," *Radiographics*, vol. 15, no. 3, p. 671, 1995.

[33] T. Nakanishi, K. Ito, M. Imazu, and M. Yamakido, "Evaluation of coronary artery stenoses using electron-beam CT and multiplanar reformation," *Journal of Computer Assisted Tomography*, vol. 21, no. 1, pp. 121–127, 1997.

[34] T. Möller, K. Mueller, Y. Kurzion, R. Machiraju, and R. Yagel, "Design of accurate and smooth filters for function and derivative reconstruction," in *Proceedings of IEEE Symposium on Volume Visualization*, 1998, pp. 143–151.

[35] R. Avila, L. Sobierajski, and A. Kaufman, "Towards a comprehensive volume visualization system," in *Proceedings of IEEE Visualization.* Los Alamitos, CA, USA: IEEE Computer Society Press, 1992, pp. 13–20.

[36] B. Mora and D. Ebert, "Low-complexity maximum intensity projection," *ACM Transactions on Graphics*, vol. 24, no. 4, pp. 1392–1416, 2005.

[37] J. Wallis, T. Miller, C. Lerner, and E. Kleerup, "Three-dimensional display in nuclear medicine," *IEEE Transactions on Medical Imaging*, vol. 8, no. 4, pp. 297–230, 1989.

[38] B. Phong, "Illumination for computer generated pictures," *Communications of the ACM*, vol. 18, no. 6, pp. 311–317, 1975.

[39] T. Möller, R. Machiraju, K. Mueller, and R. Yagel, "A comparison of normal estimation schemes," in *Proceedings of IEEE Visualization*, 1997, pp. 19–26.

[40] M. Levoy, "Efficent Ray Tracing of Volume Data," *ACM Transactions on Graphics*, vol. 9, no. 3, pp. 245–261, 1990.

[41] M. Häfner, "Conventional colonoscopy : Technique, indications, limits," *European journal of radiology*, vol. 61, no. 3, pp. 409–414, 2007.

[42] G. Akerkar, J. Yee, R. Hung, and K. McQuaid, "Patient experience and preferences toward colon cancer screening: a comparison of virtual colonoscopy and conventional colonoscopy," *Gastrointestinal Endoscopy*, vol. 54, no. 3, pp. 310–315, 2001.

[43] C. Kay, D. Kulling, R. Hawes, J. Young, and P. Cotton, "Virtual endoscopy: Comparison with colonoscopy in the detection of space-occupying lesions of the colon," *Endoscopy*, vol. 32, no. 3, pp. 226–232, 2000.

[44] A. Laghi, R. Iannaccone, I. Carbone, C. Catalano, V. Panebianco, E. D. Giulio, A. Schillaci, and R. Passariello, "Computed tomographic colonography (virtual colonoscopy): Blinded prospective comparison with conventional colonoscopy for the detection of colorectal neoplasia," *Endoscopy*, vol. 36, no. 6, pp. 441–446, 2002.

[45] I. Bitter, A. Kaufman, and M. Sato, "Penalized-distance volumetric skeleton algorithm," *IEEE Transactions on Visualization and Computer Graphics*, vol. 7, no. 3, pp. 195–206, 2001.

[46] A. Kanitsar, D. Fleischmann, R. Wegenkittl, P. Felkel, and E. Gröller, "CPR: Curved Planar Reformation," in *Proceedings of IEEE Visualization*. Washington, DC, USA: IEEE Computer Society, 2002, pp. 37–44.

[47] F. Klok, "Two Moving Coordinate Frames for Sweeping along a 3D Trajectory," *Computer Aided Geometric Design*, vol. 3, no. 3, pp. 217–229, 1986.

[48] T. Vrtovec, B. Likar, and F. Pernus, "Curved Planar Reformation of CT Spine Data," in *Proceedings of SPIE; Medical Imaging : Image Processing*, vol. 5747, 2005, pp. 1446–1456.

[49] A. Kanitsar, D. Fleischmann, R. Wegenkittl, D. Sandner, P. Felkel, and E. Gröller, "Computed tomography angiography: a case study of peripheral vessel investigation," in *Proceedings of the conference on Visualization*. Washington, DC, USA: IEEE Computer Society, 2001, pp. 477–480.

[50] S. Haker, S. Angenent, A. Tannenbaum, and R. Kikinis, "Non-distorting flattening for virtual colonoscopy," in *Proceedings of the Third International Conference on Medical Image Computing and Computer-Assisted Intervention*. London, UK: Springer-Verlag, 2000, pp. 358–366.

[51] W. Hong, X. Gu, F. Qiu, M. Jin, and A. Kaufman, "Conformal virtual colon flattening," in *Proceedings of the ACM symposium on solid and physical modeling*, 2006, pp. 85–93.

[52] A. Vilanova i Bartrolí, "Visualization techniques for virtual endoscopy," Ph.D. dissertation, Institute of Computer Graphics and Algorithms, Vienna University of Technology, Favoritenstrasse 9-11/186, A-1040 Vienna, Austria, 2001.

[53] A. Vilanova i Bartrolí, R. Wegenkittl, A. König, and E.Gröller, "Nonlinear Virtual Colon Unfolding," in *Proceedings of IEEE Visualization*, 2001, pp. 411–420.

[54] A. Vilanova i Bartrolí, R. Wegenkittl, A. König, E. Gröller, and E. Sorantin, "Virtual Colon Flattening," in *Proceedings of VisSym '01 Joint Eurographics - IEEE TCVG Symposium on Visualization*, 2001, pp. 127–136.

[55] R. Yagel and Z. Shi, "Accelerating volume animation by space-leaping," in *Proceeding of IEEE Visualization*, 1993, pp. 62–69.

[56] L. Sobierajski and R. Avila, "A hardware acceleration method for volumetric ray tracing," in *Proceedings of IEEE Visualization*, 1995.

[57] U. Tiede, N. von Sternberg-Gospos, P. Steiner, and K. H. Höhne, "Virtual Endoscopy using Spherical QuickTime-VR Panorama Views," *Studies in Health Technology and Informatics*, vol. 85, pp. 523–528, 2002.

[58] "Viatronix V3D Colon, http://www.viatronix.com."

[59] D. Chen, A. Farag, M. Hassouna, and R.Falk, "Principal curvature-based polyp detection," *International Journal of Computer Assisted Radiology and Surgery*, vol. 2, no. Supplement 1, pp. S6–S8, 2007.

[60] M. Hassouna, A. Farag, and R. Falk, "Virtual fly-over: A new visualization technique for virtual colonoscopy," in *Proc. of International Conference on Medical Image Computing and Computer-Assisted Intervention*, 2006, pp. 381–388.

[61] M. Wan, F. Dachille, K. Kreeger, S. Lakare, M. Sato, A. Kaufman, M. Wax, and J. Liang, "Interactive electronic biopsy for 3D virtual colonoscopy," *SPIE Medical Imaging*, vol. 4321, pp. 483–488, 2001. [Online]. Available: citeseer.ist.psu.edu/wan01interactive.html

[62] Z. Liang, "Virtual colonoscopy: An alternative approach to the examination of the entire colon," *INNERVISION*, vol. 16, no. 10, pp. 40–44, 2001.

[63] M. Mlejnek, P. Ermes, A. Vilanova, R. van der Rijt, H. van den Bosch, F. Gerritsen, and E. Gröller, "Profile flags: a novel metaphor for probing of t2 maps," in *Proceedings of IEEE Visualization*, 2005, pp. 599–606.

[64] D. Bartz, "Vivendi: A virtual endoscopy system supporting minimally invasive interventions," in *Proceedings of EuroGraphics*, vol. 22, no. 3, 2003.

[65] E. Coto, S. Grimm, and D. Williams, "O-Buffer based IFT watershed from markers for large medical datasets," *Computer and Graphics (accepted)*, 2007.

[66] P. Felkel, R. Wegenkittl, and M. Bruckschwaiger, "Implementation and complexity of the watershed-from-markers algorithm computed as a minimal cost forrest," in *Proceedings of EuroGraphics*, 2001, pp. 26–35.

[67] E. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, pp. 269–271, 1959.

[68] M. Levoy, "Display of Surfaces from Volume Data," *IEEE Computer Graphics and Applications*, vol. 8, no. 3, pp. 29–37, 1988.

[69] G. Marmitt, A. Kleer, H. Friedrich, I. Wald, and P. Slusallek, "Fast and Accurate Ray-Voxel Intersection Techniques for Iso-Surface Ray Tracing," in *Vision, Modeling, and Visualization*. Stanford, USA: Akademische Verlagsgesellschaft Aka, 2004, pp. 429–435.

[70] S. Parker, P. Shirley, Y. Livnat, C. Hansen, and P. Sloan, "Interactive Ray Tracing for Isosurface Rendering," in *Proceedings of IEEE Visualization*, 1998, pp. 233–238.

[71] E. Gröller, "Nonlinear raytracing - visualizing strange worlds," *Visual Computer*, vol. 11, no. 5, pp. 263–274, 1995.

[72] G. Wang, S. Dave, B. Brown, Z. Zhang, E. McFarland, J. Haller, and M. Vannier, "Colon Unraveling Based on Electrical Field: Recent Progress and Further Work," *Proceedings of SPIE*, vol. 3660, no. 1, pp. 125–132, 1999.

[73] R. Adams and L. Bischof, "Seeded Region Growing," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, pp. 641–647, 1994.

[74] A. Mehnert and P. Jackway, "An Improved Seeded Region Growing Algorithm," *Pattern Recognition Letters*, vol. 18, pp. 1065–1071, 1997.

[75] D. Williams, S. Grimm, E. Coto, A. Roudsari, and H. Hatzakis, "Volumetric curved planar reformation for virtual endoscopy," *IEEE Transactions on Visualization and Computer Graphics (to appear)*, 2008.

[76] D. Williams, S. Grimm, E. Coto, A. Roudsarri, and H. Hatzakis, "Volumetric cpr as an enhancement to virtual colonoscopy systems," *International Journal of Computer Aided Radiology and Surgery*, vol. 2, no. Suppl 1, pp. 4–6, June 2007.

[77] C. Friedman and J. Wyatt, *Evaluation Methods in Biomedical Informatics*. Springer, 2005.

[78] D. Williams, S. Grimm, E. Coto, A. Roudsari, and H. Hatzakis, "A Light At The End Of The Tunnel: Enhancing The Volumetric CPR," Unpublished.

[79] S. Gokturk, C. Tomasi, B. Acar, D. Paik, C. Beaulieu, and S. Napel, "A new 3-D volume processing method for polyp detection," *Engineering in Medicine and Biology Society*, vol. 3, pp. 2522–2525, 2001.

[80] R. Summers, C. Beaulieu, L. Pusanik, J. Malley, J. Brooke, D. Glazer, and S. Napel, "Automated polyp detector for CT colonography : Feasibility study," *Radiology*, vol. 216, no. 1, pp. 284–290, 2000.

[81] "Medicsight CAD, http://www.medicsight.com."

[82] L. Ibanez and W. Schroeder, *The ITK Software Guide*. Kitware, Inc., 2005.

[83] A. Kanitsar, R. Wegenkittl, D. Fleischmann, and E. Groller, "Advanced curved planar reformation: Flattening of vascular structures," in *Proceedings of IEEE Visualization*. Washington, DC, USA: IEEE Computer Society, 2003, p. 7.

[84] "Terarecon VolumePro Board, http://www.terarecon.com."

Appendix A

Volumetric Curved Planar Reformation for Virtual Endoscopy

This paper was accepted for publication in IEEE Transactions on Visualization and Computer Graphics and is expected to be published in early 2008. It was our first paper, and describes the technical implementation of the Volumetric CPR. This includes some motivation for the technique, a description of the different project methods and how they affect the resulting image, and a description of our image space approach for restricting the number of raycast pixels. The paper is already available in the IEEE Digital Library.

IEEE TRANSACTIONS ON VISUALIZATION AND COMPUTER GRAPHICS, VOL. 14, NO. 1, JANUARY/FEBRUARY 2008

Volumetric Curved Planar Reformation for Virtual Endoscopy

David Williams, Sören Grimm, Ernesto Coto, Abdul Roudsari, and Haralambos Hatzakis

Abstract—Curved Planar Reformation (CPR) has proven to be a practical and widely used tool for the visualization of curved tubular structures within the human body. It has been useful in medical procedures involving the examination of blood vessels and the spine. However, it is more difficult to use it for large tubular structures such as the trachea and the colon because abnormalities may be smaller relative to the size of the structure and may not have such distinct density and shape characteristics. Our new approach improves on this situation by using volume rendering for hollow regions and standard CPR for the surrounding tissue. This effectively combines gray-scale contextual information with detailed color information from the area of interest. The approach is successfully used with each of the standard CPR types, and the resulting images are promising as an alternative to virtual endoscopy. Because CPR and volume rendering are tightly coupled, the projection method used has a significant effect on the properties of the volume renderer, such as distortion and isometry. We describe and compare the different CPR projection methods and how they affect the volume rendering process. A version of the algorithm is also presented which makes use of importance-driven techniques; this ensures the users' attention is always focused on the area of interest and also improves the speed of the algorithm.

Index Terms—Curved planar reformation, virtual endoscopy, colon screening.

◆

1 INTRODUCTION

VIRTUAL endoscopy has greatly evolved since its earlier days when it suffered from difficulties in generating high-quality images at the real-time frame rates required for interactivity, and it has now overcome many of these technical obstacles to become widely accepted and applied. Virtual endoscopy is used in many hospitals, and its benefits to patients and clinical staff are widely acknowledged. In this paper, we will be addressing certain problems that arise during virtual endoscopy such as maximizing the surface coverage and (in the case of virtual *colonoscopy*) dealing with contrast-enhanced fluid and insufficient inflation.

The most common solution in virtual endoscopy is to perform a virtual flythrough; this focuses on reconstructing the image that would normally be produced by the endoscope if an optical endoscopy were to be performed. This makes sense because clinicians are used to seeing such images and know how to interpret them. However, such an approach suffers from problems with surface coverage (especially in the colon) as parts of the surface may be occluded by folds. Also, a relatively small area is displayed at once, which increases the examination time.

We present here a new approach based on the concept of *Curved Planar Reformation* (CPR) [1]. By combining a CPR rendering and volume rendering, we are able to generate images that clearly show the areas of interest in 3D and provide context and spatial information via the 2D surroundings. This technique, which we call *Volumetric* CPR, allows a larger percentage of the tissue wall to be visualized at once and therefore improves the diagnostic value compared to a virtual flythrough.

The remainder of this paper is organized as follows: A brief review of approaches related to virtual endoscopy is given in Section 2. Next, we discuss the two core visualization approaches from which our work is derived: volume rendering from a cut plane (in Section 3) and CPR (in Section 4). This provides the basis for our new approach, which is described in Sections 5 and 6. An importance-driven extension is then introduced in Section 7 before methods of ensuring maximum surface coverage are considered in Section 8. Finally, Section 9 gives conclusions and a discussion.

2 RELATED WORK

There has been a large amount of previous research on using flythroughs as an approach to virtual endoscopy. This previous work has been concerned with rendering real-time images [2], [3], [4], [5], developing an intuitive camera and navigation model [6], and performing a computer-aided diagnosis such as polyp detection [7].

The technique of *virtual colon flattening* [8], [9], [10] provides an alternative approach for the colon that aims to increase the surface area that is seen. It also displays a much larger area of the colon wall at any given moment, allowing the user more time to identify polyps. The flattening is performed via a rendering technique in which rays are cast from the centerline and essentially project the colon surface

---

- *D. Williams and A. Roudsari are with the Centre for Health Informatics, School of Informatics, City University, London, UK, EC1V OHB. E-mail: {d.p.williams, A.V.Roudsari}@city.ac.uk.*
- *S. Grimm and H. Hatzakis are with Biotronics3D Ltd., The Gatehouse, Trinity Buoy Wharf, 64 Orchard Place, Docklands, London, UK, E14 0JW. E-mail: {sgrimm, hhatzakis}@biotronics3d.com.*
- *E. Coto is with the Laboratorio de Computación Gráfica, Escuela de Computación, Universidad Central de Venezuela, Apartado Postal 47002, Los Chaguaramos, 1041-A, Caracas, DC, Venezuela. E-mail: ecoto@ciens.ucv.ve.*

Manuscript received 25 Aug. 2006; revised 31 Jan. 2007; accepted 3 Apr. 2007; published online 24 May 2007.
Recommended for acceptance by C. Silva.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org, and reference IEEECS Log Number TVCG-0138-0806.
Digital Object Identifier no. 10.1109/TVCG.2007.1061.

onto a cylinder that is then unrolled. Vilanova i Bartrolí et al. [8], [9] also describe a technique to reduce the distortion that can result from this projection.

Although not used for virtual endoscopy, CPR [1], [11] is another visualization approach that is widely used for procedures such as the screening of blood vessels for calcium deposits. CPR allows the user to define a curve in 3D space and map it to the screen, commonly in one of three ways that trade off occlusion, isometry, and spatial perception.

We present a new visualization that is based on the concept of CPR. By extending CPR with direct volume rendering techniques based on ray casting [12], we are able to generate vivid color images of the inside of the tubular structure while also providing the CPR as context, similar to other "focus+context" visualization techniques [13].

3 COMBINING MULTIPLANAR REFORMATION (MPR) AND RAY CASTING

Almost every volume visualization application supports MPR, which allows arbitrary planes to be defined in the volume and for the voxels that are intersected by these planes to be mapped to the screen as pixels, usually modified by some window-level function. It is relatively straightforward to extend this concept by performing ray casting from pixels in the dark areas of the image (corresponding to empty areas of the volume) and terminating the rays once they hit a surface.

The problem with this approach is that very few structures in the body lie on the flat 3D plane that MPR allows to be defined. The colon (encompassing both the red and blue sections in Fig. 1a) is a notable example of this due to its high curvature and knotted shape. Any flat plane will intersect only a small portion of the structure of interest, which significantly limits its use, as shown by Figs. 1b, 1d, 1e, and 1f. It is also difficult to interact with the visualization in an intuitive way because there are too many degrees of freedom; visualizing the next section of colon requires the plane to be moved and rotated in all three dimensions.

We propose that a solution to these limitations lies in the use of *curved* planes that follow the shape of the structure being examined, as shown in Fig. 1c. This significantly complicates the process of mapping the plane to the screen (as will be discussed in Sections 4, 5, and 6) but allows more of the structure to be visualized at once (Fig. 1g). The interaction with the system is also easier as the user simply moves forward and backward along the curve and rotates around its centerline to reveal parts of the surface that were previously behind the viewer.

4 CONVENTIONAL CPR

CPR is a technique which removes one of the limitations of MPR for medical volume exploration by allowing the user to specify *curves* rather than planes and mapping these onto the screen. This curve may also be generated automatically as the result of a previous segmentation step.

The curve is defined by a centerline and a *resampling vector* that specifies the direction in which the line is extended. This centerline is in turn defined by a series of points at subvoxel resolution. The image generation algorithm works by taking successive centerline points along the centerline (oversampled if necessary) and resampling in the volume along the resampling vector. The way in which this resampling is performed and the way in which the samples are mapped to the image space are dependent on the projection method used. In [1], Kanitsar et al. identify three commonly used projection methods that preserve spatial perception and isometry to varying degrees:

1. **Projected CPR.** Each centerline point maps to a scan line based upon its mathematical projection onto the image plane. There may be many points on the centerline that project to the same scan line and, so, compositing can optionally be used to combine them. The resampling direction is the same for all centerline points. This method has high spatial perception but does not preserve isometry.
2. **Stretched CPR.** Each centerline point maps to a scan line based upon its distance from the start point. This distance is measured *along the curve* (rather than linearly) and this effectively stretches the curve. The process removes any occlusion, but at the expense of reducing spatial perception. As with the projected CPR, the resampling direction is the same for all centerline points, but isometry is preserved.
3. **Straightened CPR.** Each centerline point maps to a scan line based upon its distance from the start point. It always maps to the center of the scan line that has the effect of straightening the centerline in the image space. The resampling direction varies between centerline points and is always given by the normal to the centerline. To avoid artifacts due to excessive rotation, it is possible to use the techniques described by Klok in [14].

CPR is useful in many areas, such as visualization of computerized tomography (CT) spine data [15] and CT angiography [16]. In the latter case, the centerline is placed through the aorta, and a straightened CPR is generated. This yields an image that can be rotated around the centerline to screen the whole of the vessel wall for calcium deposits.

The same approach can be used on other tubular structures such as the colon, though polyps can only be seen when they are intersected by the curved plane. Our new technique addresses this issue and makes CPR much more useful for such purposes.

5 VOLUMETRIC CPR

Our proposed new visualization is a hybrid 2D/3D approach that incorporates volume rendering into CPR. We essentially perform CPR based on a centerline through the colon and then augment the image by performing volume rendering for the dark parts of the image—we call this new technique *Volumetric* CPR. As shown in Fig. 2, this makes a considerable difference to the information contained in the images.

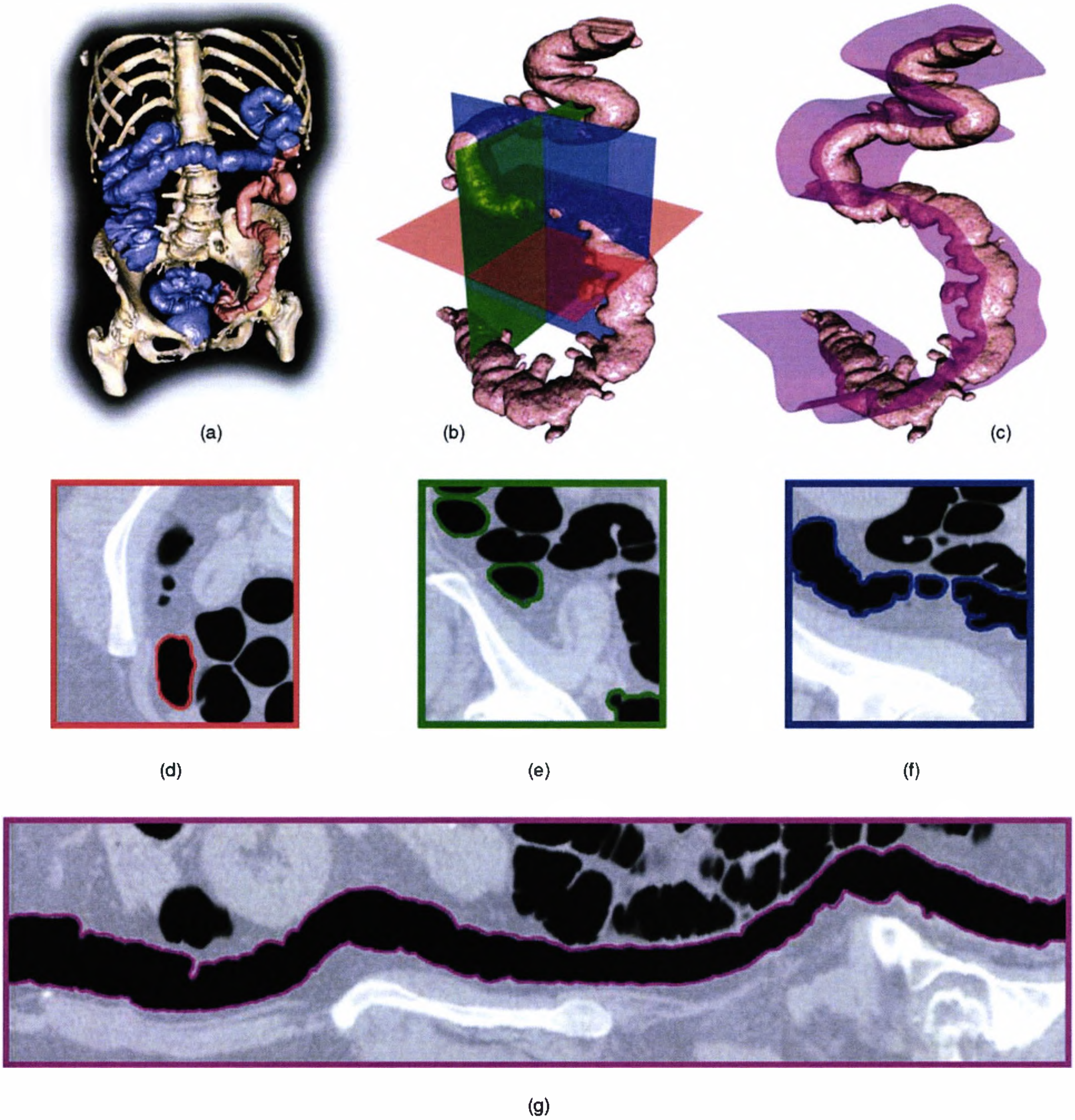


Fig. 1. The colon is an excellent example of an anatomical structure that cannot be effectively visualized with flat cut planes. An overview of the colon is shown in (a) with a 57-cm section marked in red. The arrangements of some flat cut planes and a curved cut plane are shown in (b) and (c), and the images that result from projecting them to the screen are shown in (d), (e), (f), and (g).

Although it is relatively straightforward to extend *MPR* with volume rendering (as described in Section 3), it is significantly more difficult to do so for the *CPR* case. As is already established, there are three different projection methods available for mapping the plane to the image, and it is very important to choose a suitable method for the data being examined. We describe the properties of the various projection methods in Section 6.

Regardless of the projection method chosen, our prototype implementation uses ray casting [12], [17] for the volume rendering, mainly due to its greater flexibility compared to other volume rendering algorithms and the fact that the projection is not necessarily orthographic or perspective for all CPR types. Specifically, it is an isosurface ray caster and, so, it is possible to take advantage of some of the existing techniques for space leaping [3], [18] and accurately finding the isosurface [19], [20].

The algorithm begins as a conventional CPR by identifying the first centerline point on the centerline and then resampling along the resampling vector. It maps each sample to a pixel and determines, according to some criterion, whether the pixel should get its color from the sample that was just taken or through ray casting of the volume. If it decides the former, then the sampled value is modified by the window-leveling function and drawn to the image. If it decides on the latter, then a ray is set up in

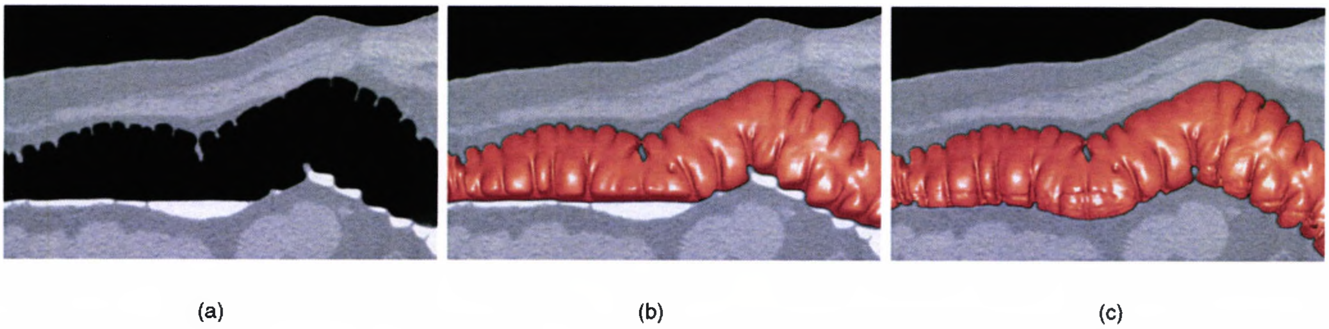

(a) (b) (c)

Fig. 2. (a) A CPR of a section of the colon without volume rendering. (b) Adding volume rendering produces a much more useful image that reveals a previously invisible polyp. (c) Electronic cleansing can be performed to remove the contrast-enhanced fluid.

the appropriate direction, and ray casting is performed before drawing the result. Hence, there are three main questions that must be answered during the rendering process:

1. Which pixels should have their color determined by the window-leveling function and which by ray casting?
2. In which direction are rays cast?
3. What are the criteria for terminating a ray?

A simple criterion for determining whether or not a pixel should be ray cast is its resampled intensity value, as used previously when enhancing MPR in a similar way. If the intensity value is high, then the window-leveling function is applied to give the final color, and if it is low, then ray casting is used instead. This is a simple yet effective approach that is fast to check and results in images like that shown in Fig. 2b. An even better criterion that makes use of segmentation information is presented later in Section 7.

The direction of the rays is dependent upon the projection method being used. For the projected Volumetric CPR, the ray direction is computed from the relative positions of the object and view plane, whereas, for the stretched and straightened CPR, it is computed from the shape of the curve. This is discussed in more detail in Section 6.

The rays are terminated once the resampled intensity value goes above a certain threshold, indicating that it is leaving the air-filled region. This usually means that the ray has hit the surrounding tissue, but in the case of virtual colonoscopy data sets, it is common for a contrast enhancer such as barium to also be present—this marks the stool, so it can be easily recognized during the examination. Such contrast-enhanced fluid typically shows up as bright white in standard CPR rendering (Fig. 2a), and because it has a high density, it will cause ray termination and show up as flat surfaces in the volume rendered part of Volumetric CPR (Fig. 2b) or during virtual flythroughs.

Because of the problems presented by such fluid covering polyps or blocking the colon, it should normally be removed before the data is visualized, as shown in Fig. 2c. However, this cleansing process cannot be guaranteed to be perfect and must be conservative to avoid removing polyps that may be concealed within the fluid. This means it is quite possible for a thin layer of fluid to be left covering parts of the colon wall. During a flythrough or even a colon flattening, this remaining fluid is just displayed as part of the colon wall, and so, the user is unaware that there are potentially some polyps still hidden. An advantage of the Volumetric CPR (and an example of where context information can be useful) is that this remaining fluid can still be seen as a thin white region outside the ray-cast area. It is then possible to rotate the Volumetric CPR to check for polyps.

A further option that is made available by the Volumetric CPR is to not perform electronic cleansing at all and, hence, avoid the risk of polyps being accidentally removed. This was not previously an option with a virtual flythrough because the fluid could occlude polyps or even completely block the colon. With Volumetric CPR, the contrast-enhanced fluid can be easily identified, and even blocked regions can be scanned for polyps by rotating. This approach is also applicable in areas where the colon is not sufficiently inflated to allow a flythrough.

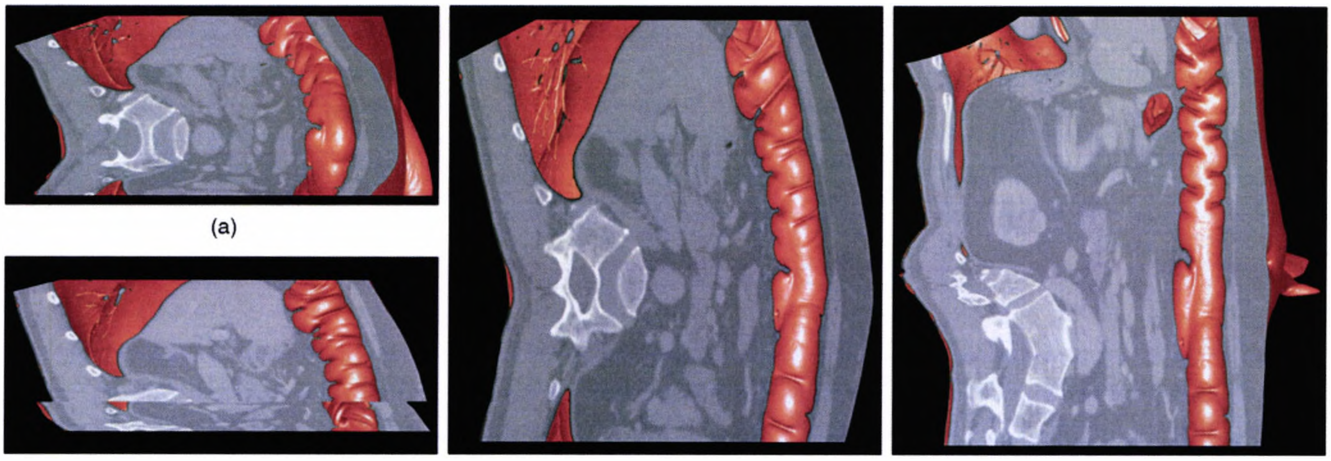
6 COMPARISON OF PROJECTION METHODS

As with a conventional CPR, the projection method used makes a significant difference to the resulting image [1]. In fact, the difference is even greater with the ray-casting enhancements because the projection method has an effect on the direction in which the rays are cast; see Fig. 3 for a comparison of the resulting images. In this section, we look at the implications of using each of the projection methods.

6.1 Projected Mode

A projected Volumetric CPR takes each point along the centerline and projects it into the image space. The values along the projected scan line are then computed by resampling in the volume along the resampling vector. If the curve folds back on itself, then more than one point on the centerline will map to the same scan line. When this occurs, a policy is required to determine what should be displayed; within our implementation, the old value is simply overwritten, which can cause occlusion to occur in the image. An alternative is to use some form of compositing, but this usually results in confusing images that are hard to understand.

When operating in projected mode, our Volumetric CPR casts rays from points on the curve corresponding to empty voxels in the direction of the normal to the viewport. That is, all rays that are cast are parallel; this results in an orthographic 3D rendering. This is illustrated for the



(a)

(b)

(c)

(d)

Fig. 3. The same piece of colon is shown above using different projection methods. Projection is used for (a) and (b); in the case of (b), the camera has been rotated to demonstrate how occlusion can occur. Occlusion cannot occur in the stretched or straightened cases shown in (c) and (d), respectively.

general case in Fig. 4, which also shows the area the curve is projected onto and the places in which occlusion occurs.

A real example is illustrated in Fig. 5, which shows the process of rendering a curved segment of a tubular structure. In this particular case, the view plane is positioned such that no occlusion occurs. Fig. 3a shows the result of rendering a colon segment with this setup.

In the middle of Fig. 3a, the rendering is undistorted, and it is easily possible to identify the folds of the colon. However, toward the top and bottom of the image, a form of distortion occurs. Referring to Fig. 5 reveals why this is the case; it can be seen that, near the center of the image, rays travel from the centerline to the *nearest* part of the surface, whereas, near the top and bottom of the image, the rays travel a much greater distance to a part of the surface that is further away. This has the effect that, near the top and bottom of the image, the 3D rendering no longer corresponds as closely to the surrounding CPR that provides the context. For this reason, projected Volumetric CPR is most applicable to short lengths of tubular structures or those that have only a low degree of curvature.

A further issue occurs when we rotate the viewport or the curve to examine a different part, as shown in Fig. 6. Note that, in both Figs. 5 and 6, there is a point marked A from which rays are cast into the surface. This point is the same in both figures, but the rays that are cast reach a different point on the surface. As we rotate the curve, the 3D-rendered position corresponding to a CPR position can change. Although this can look strange, it is not disorientating for the user because there is only a small change between each frame.

Fig. 6 also illustrates how occlusion can occur, which, as previously stated, is as a result of multiple points on the centerline mapping to the same scan line. The easiest approach is simply to overwrite the previous contents of the scan line with the new values; this results in images like that shown in Fig. 3b. It could be argued that this is actually the correct behavior because a part of the curve near the view plane is occluding a part of the curve that is further away,

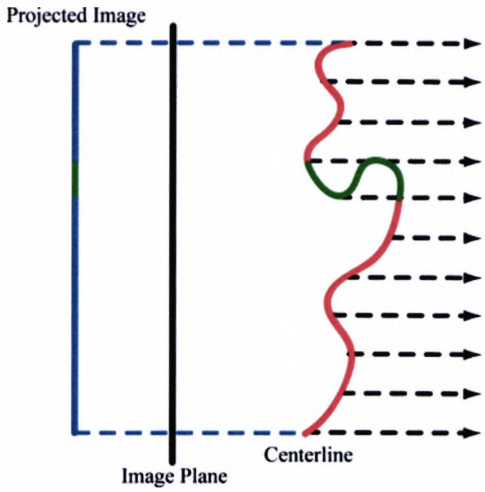


Fig. 4. Ray pattern for the Volumetric CPR. The area shown in green is where occlusion occurs.

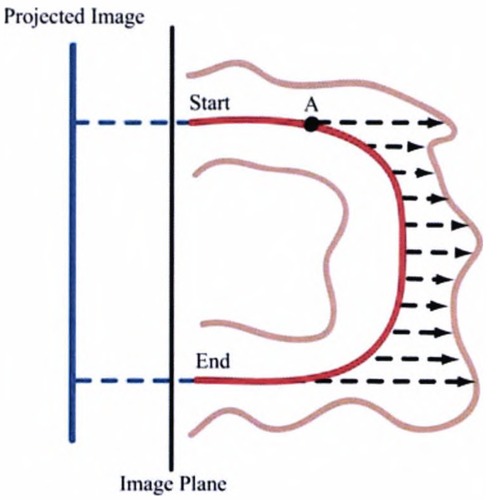


Fig. 5. Rendering a projected Volumetric CPR of a tubular structure. This ray pattern gives an image like Fig. 3a when applied to the colon.

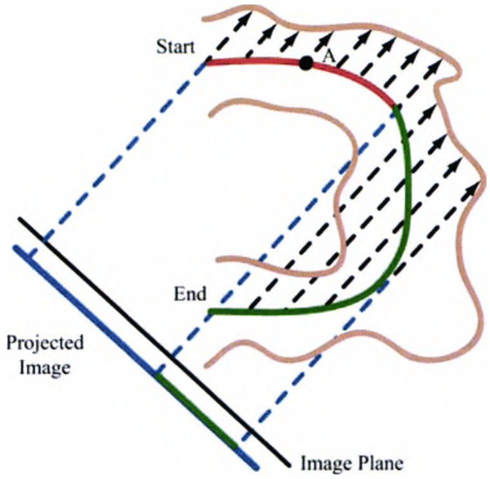


Fig. 6. The same arrangement as in Fig. 5 but with rotation. This causes occlusion to occur on the green part of the image plane as the whole of the green part of the curve maps to this region.

but, unfortunately, this behavior is not guaranteed. If we had performed the Volumetric CPR generation by starting at the other end of the centerline (marked "End" on Fig. 6 rather than "Start"), then the far part of the curve would have been rendered over the near part of the curve. For some curve configurations, there is no start/end pair that gives the correct ordering, though a depth buffer can help resolve this.

6.2 Stretched Mode

The stretched mode builds and improves on the projected mode by eliminating the occlusion and some of the distortion that occurs. The occlusion is addressed by ensuring that no two points on the centerline map to the same scan line in the image space. As with the projected Volumetric CPR, the algorithm takes successive centerline points along the centerline and maps them to the image space before resampling into the volume along the resampling vector to give values along the scan line. However, rather than simply projecting the centerline point, its distance from the start point along the centerline is used as an offset from the bottom of the resulting image. No two points on the centerline have the same distance from the start and, hence, no occlusion can occur.

The stretched Volumetric CPR algorithm is further modified from the projected version such that the rays are no longer always sent parallel to the view direction. Instead, the direction is modified according to the curvature at the centerline point. This is illustrated in Fig. 7. Note that the rays are still parallel *within any given scan line* but are not necessarily so between scan lines.

Modifying the ray direction in this manner leads to some reduction in distortion compared to the parallel ray approach. The issue that was illustrated earlier in Figs. 5 and 6 (namely, that the point on the wall that corresponds to a point on the curve varies as the view plane is moved) does not occur because the ray direction is no longer dependent on the view plane or curve position.

However, a new artifact can present itself, as shown in Fig. 8. The centerline near point A is exhibiting a high degree of curvature, which causes the rays to diverge; this in turn causes the spacing of the sample points along the wall to increase. Near point B, the opposite occurs. In both cases, the perceived effect is a form of distortion: near point A, parts of the wall get compressed, whereas near point B, parts of the wall are stretched (or may even appear twice).

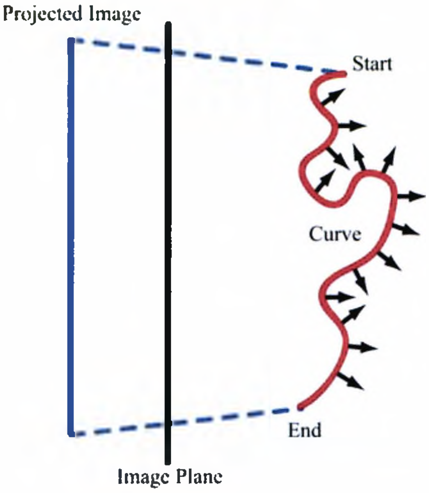


Fig. 7. In a stretched Volumetric CPR, rays are no longer sent in a parallel formation.

We are certainly not the first to have experienced these difficulties when mapping a curve to a plane and, fortunately, there are some known approaches that can help reduce the problem or at least trade off different forms of distortion. During colon flattening, Vilanova i Bartrolí et al. [8], [9] made use of an adaptive sampling approach in order to ensure that samples are taken at equidistant intervals along the colon wall. A similar approach is applicable here, but it should be noted that varying the frequency of samples along the centerline will mean that a given distance in image space will no longer correspond to a fixed distance in the CPR; whether this is a beneficial trade-off will depend on the application.

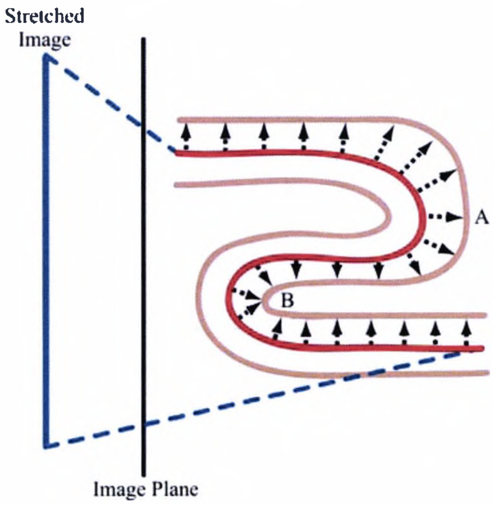


Fig. 8. Ray spacing along the walls varies according to curvature.

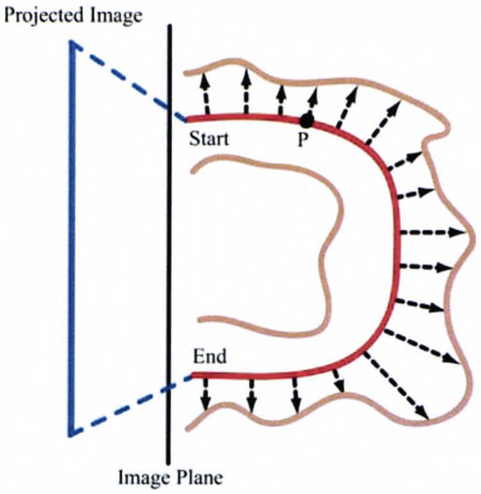


Fig. 9. Rendering a stretched Volumetric CPR of a tubular structure. This ray pattern gives an image like Fig. 3c when applied to the colon.

A second approach is to curve the rays as they traverse space to perform a kind of nonlinear ray casting [8], [21]. This can be used to prevent the rays from intersecting, but it requires some extra computational expense and a precomputation stage [22]. A real example of the stretched Volumetric CPR in action is given in Fig. 3c, and the ray pattern is shown in Fig. 9.

There are some similarities between the ray-cast images generated by our stretched Volumetric CPR and those generated by colon flattening without the ray-curving modification. Both techniques make use of the centerline as an origin for the rays, but there are differences in the way they are cast.

During colon flattening, rays emanate from the centerline in all directions in a way that either maintains a constant angle between samples or maintains a constant sampling distance along the wall of the colon. This is shown in Fig. 10a.

By contrast, the Volumetric CPR takes the centerline and generates the curve that corresponds to it. Rays are then cast from this curve, as shown in Fig. 10b; this means that rays are parallel within a scan line but are not so between scan lines.

6.3 Straightened Mode

The stretched mode described previously allows an arbitrary centerline to be defined in 3D space. However, the curve is then formed from the centerline by extending it along the resampling vector; this vector is the same for all points and, so, the curve is always extended in the same direction (see Fig. 11a).

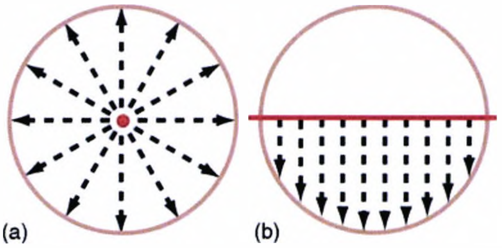


Fig. 10. (a) During colon flattening, the rays radiate from the centerline. (b) During stretched Volumetric CPR, they are all normal to the curve.

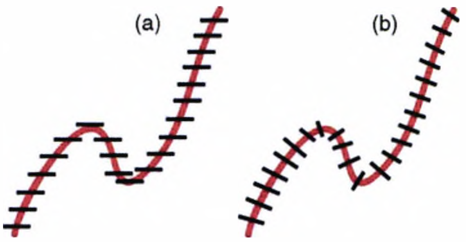


Fig. 11. (a) For a stretched Volumetric CPR, all resampling vectors are in the same direction. (b) For the straightened Volumetric CPR, they follow the normal to the centerline.

For the straightened CPR, we drop the restriction that the resampling vector is the same for all points on the centerline. Instead, the resampling vector is now specified by the normal to the centerline at any given point, and this results in a curve that twists and turns in the full three dimensions, as shown in Fig. 11b.

For each point on the curve, there are actually an infinite number of normals that could be used, and the choice of normal has a significant effect on the resulting image. A simple approach is to compute the normal using the Serret-Frenet equations, but this has the disadvantage that twists can appear when the curve changes direction. Instead, we choose to compute the normal $\mathbf{n}$ from the tangent $\mathbf{t}$ to the curve and an external reference vector $\mathbf{r}$ such that

$$\mathbf{n} = \mathbf{t} \times \mathbf{r}.$$

Because the successive resampling vectors are directly mapped to scan lines in the resulting image, varying them along the centerline has the advantage of straightening the centerline in the image space.

Because centerline points are mapped to the center of each scan line, the centerline is straightened in the image space. As with the conventional straightened CPR, this has the benefit of making the images significantly easier to understand and to interact with as rotation in the projected and stretched mode can be nonintuitive.

A straightened Volumetric CPR of the same section of colon that was shown in Figs. 3a, 3b, and 3c is shown in Fig. 3d. A conventional straightened CPR suffers from some distortion at points that are not near to the centerline, and this continues to be the case for the straightened Volumetric CPR. This is caused by the resampling vectors intersecting as they move away from the centerline.

7 RESTRICTING THE RAY-CAST AREA

The algorithm described in the previous sections gives some pleasing results, but further refinements are proposed. Looking at Fig. 12a, it can be seen that some of the areas in which ray casting has been used do not actually correspond to areas of interest, and there are two reasons why it is not desirable to perform ray casting for these areas. First, they can be a distraction; the clinicians' attention should be focused on the area lying along the section of the

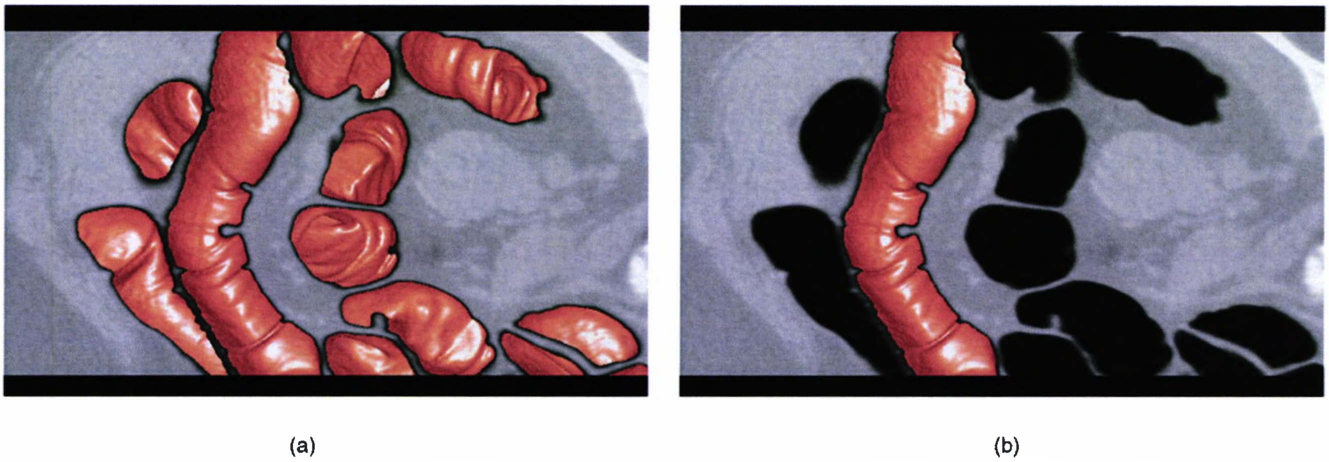


Fig. 12. Region growing can be used to restrict the number of pixels for which direct volume rendering is performed. (a) Without region growing, areas not of immediate interest are still augmented with direct volume rendering. (b) Using region growing rectifies this.

centerline section currently being visualized, rather than a section of the centerline that they may have already been seen. Second, the ray casting of those areas can have a significant performance impact. This is especially true for areas that are outside the body as rays may travel a long distance before hitting a surface or leaving the volume.

What is needed is an importance-driven criterion for deciding which pixels should be ray cast. There are several candidates; we describe two simple approaches and a more complex one that has proven very successful. During the Volumetric CPR generation, the volume is resampled along the resampling vector starting from the current centerline point. It is trivial to limit the distance to which this resampling occurs, and this results in an image that is clipped a short distance from the centerline. This clipping distance needs to be chosen carefully to ensure that it is large enough to always cover the structure of interest and small enough to minimize other air-filled parts of the volume, but, in practice, this approach works fairly well for the straightened CPR. For other CPR types, it is less effective due to the curved nature of the centerline in the image space.

A second simple solution is a scan-line-based connectivity approach. For each scan line, a flag is kept, indicating whether ray casting can currently be performed. This flag is initially true when the rendering of the scan line is started and is set to false when the samples go above a certain threshold. This occurs when the scan line leaves the structure of interest and, then, no further ray casting is performed for that scan line. Again, this technique can be reasonable when applied to a straightened CPR with a centerline that is vertical in the image space but fails with other types (see Fig. 13).

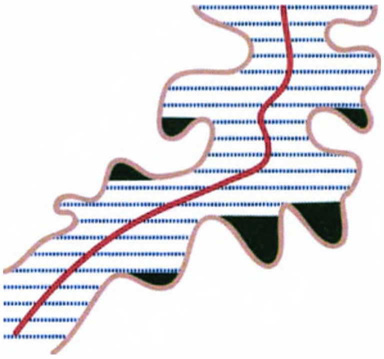


Fig. 13. The scan-line approach to clipping fails under many circumstances. Areas in green should be ray cast but are not.

Our solution is instead to only perform ray casting for those pixels that are connected to the centerline in the image space. This is determined by a region-growing approach using the pixel intensities as the criterion and the centerline's image-space position as a set of seed points. This prevents the ray casting of pixels outside the body and in other air-filled structures and also avoids the issue illustrated in Fig. 13.

We have implemented the rendering using a two-pass approach. During the first pass, the CPR is rendered as normal using any of the available projection methods, but each centerline also gets projected to the image space for use as a seed in the second pass. During the second pass, region growing [23], [24] is performed from the seed points to determine the list of pixels that are connected to the centerline in the image space. Ray casting is then used to determine the color for each of the pixels in the list, and the result is written over the previous value.

Breaking the rendering into two passes introduces some additional complications. A CPR image is usually an irregular shape, and yet, it is drawn to a rectangular viewport. Some viewport pixels therefore fall outside the data volume, and it is conventional to draw these black; this causes problems during the second pass as the region-growing algorithm cannot tell which black pixels are air-filled structures and which simply fell outside the image. To avoid this situation, the first pass sets a flag for each pixel of the viewport, indicating whether it is inside the image. This can then be checked during the second pass. Fig. 14 shows the different categories of pixel that result from this process.

The complete algorithm generates images like those shown in Fig. 12b as compared to Fig. 12a, which does not use the new region-growing approach.

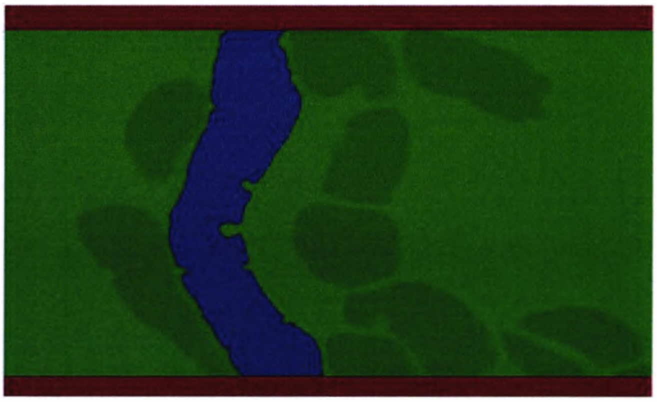


Fig. 14. Pixels are tinted *blue* if they were originally black, are inside the image, and are connected to the centerline; these pixels should be ray cast. Pixels are tinted *red* if they were originally black but fell outside the image. The remainder of the pixels are tinted *green* and either were not originally black or were originally black but are not connected to the centerline.

8 MAXIMIZING SURFACE COVERAGE

An important aim for virtual endoscopic techniques is to maximize the surface area that is seen during the visualization process. This increases the chance of the user identifying any features that are characteristic of the condition being diagnosed.

Virtual flythroughs, the most commonly used approach, do not guarantee that the whole surface is seen, as it is possible for areas to be occluded, for example, behind folds of the colon wall, as shown in Fig. 15. Solutions to this problem include providing reverse or multiple views (which require the user to watch two displays at once) or marking seen areas during the flythrough and allowing the user to go back to any areas that were missed.

The Volumetric CPR can also have problems with surface coverage; the most obvious of which is that only half of the surface is seen at a time. This is because the curve essentially cuts the tubular structure in half and casts rays in only one direction (see Fig. 16a). A simple solution would be to cast rays in both directions and show the results as two separate images (Fig. 16b). This is a good approach but can fail in some situations, such as when the centerline is not perfectly "central" or when the tube has a complex cross-sectional shape. These situations are shown in Figs. 16c and 16d, respectively.

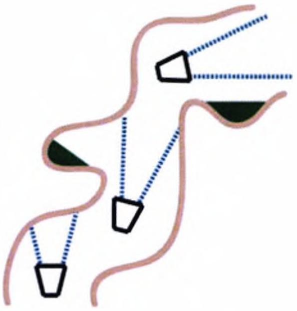


Fig. 15. The virtual-flythrough approach can miss the areas marked in green.

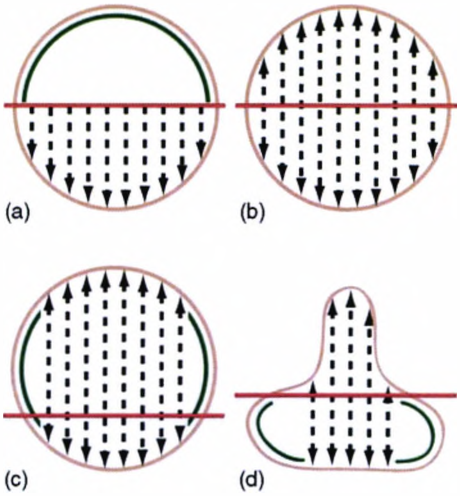


Fig. 16. (a) In Volumetric CPR, only half of the surface of the tubular structure is covered as the areas marked in green are missed. (b) This can be rectified by rendering two images with the rays going in opposite directions. However, this solution can fail if (c) the centerline is not accurate or if (d) the cross section is of a more complex shape.

Our current proposed approach is to simply rotate the curve around the centerline, as with CT angiography, either presenting this to the user as a video or allowing the user to control it interactively. It can be seen that, in the case of Fig. 16d, this will allow all the surface area to be seen. This is true for all cross-sectional shapes that commonly occur during endoscopy.

9 CONCLUSION

We have presented an extension to CPR that enhances its capabilities as a tool for examining the inside of tubular structures. We have illustrated the different projection methods for mapping a curve to the screen and discussed the effect this has on volume rendering.

The projected Volumetric CPR is useful when the tubular structure being visualized does not exhibit such a high degree of curvature that occlusion occurs. When this is the case, it has the advantage that spatial perception is maintained, and so, the user is less likely to get disoriented. It also does not suffer from distortion for parts of the curve that are facing the viewer; these are generally the areas the user is currently interested in.

In cases where the structure does have a high degree of curvature, such as the colon, it is necessary to view shorter lengths at a time. This can be an effective complement to a virtual flythrough as it allows the surrounding walls to be visualized while the camera is facing down the structure. This helps increase the percentage of the surface that is seen and can improve the chances that lesions are detected.

Stretching the Volumetric CPR makes it significantly easier to see the whole of the structure at once; the lack of occlusion means that it is useful for highly curved structures, and it is also easier to manipulate because there is no need to choose a suitable viewing angle. Although distortion can occur, it is possible to use the adaptive curved-ray approaches mentioned in Section 6.2 to alleviate this.

TABLE 1
Frame Rates in Frames per Second

	CPR	VCPR	VCPR + RG
Projected	20.0	3.2	8.2
Stretched	21.2	2.38	9.1
Straightened	27.7	5.0	8.6

The straightened Volumetric CPR is a very powerful technique in highly curved structures. The straightening in the image space makes it very easy to interpret and also makes it more practical to implement as it can be displayed in a viewport of fixed size.

Our region-growing-based approach to performing direct volume rendering has proven very effective both as a method of keeping the user focused on the area of interest and as a method of speeding up the visualization. In the future, we would like to look at this further and potentially find better ways of determining which areas are interesting.

In order for any of the techniques to be useful, it is necessary for their operation to be interactive so as to allow the user to rotate the curved plane around the centerline, as discussed earlier in Section 8. In Table 1, we provide the frame rates for the CPR, Volumetric CPR, and Volumetric CPR with region growing when using each of the three projection methods. The timings were recorded on a Athlon 3200+ with 1 Gbyte of RAM while rendering images like those shown in Figs. 3a, 3c, and 3d to a 512 × 512 pixel viewport. The renderer ran only on the CPU and did not make use of hardware acceleration.

Conventional CPR is clearly interactive, whereas the addition of ray casting adds significant overhead. It should be noted, however, that we have not implemented any of the many space-skipping techniques that are available [3], [18], and these could add a significant speedup. Restricting the number of ray-cast pixels with our region-growing approach brings the frame rates back to interactive levels.

The technique was demonstrated to the Gastrointestinal Department of the University College London Hospital and has received positive comments and feedback. The prototype system used two Volumetric CPRs (looking up and down) that were synchronized with a flythrough but did not rotate. These scrolled across the screen as the flythrough proceeded and could be used to check missed regions or as a navigational tool. Clicking the Volumetric CPR aligned the flythrough with that point.

This early stage trial has proven that Volumetric CPR is a viable technique and, when combined with other approaches such as a flythrough, could help provide more of the information required when assessing a patient's condition. We are planning to conduct a larger scale clinical trial with the group based on this and similar techniques for the analysis of gastrointestinal diseases.

Further research is likely to focus on more thorough clinical evaluations and applying the Volumetric CPR to new domains. Preliminary work has already been done with CT angiography and examination of the "Circle of Willis" and initial results are promising.

ACKNOWLEDGMENTS

This work was partially funded by the Engineering and Physical Sciences Research Council (EPSRC) Industrial Case Studentship 2004/05 Ref no. 04303095. The "prone16" and "supine16" data sets used for some of the figures are publicly available from http://www.volvis.org, and a diagnosis confirming the presence of polyps is available from http://nova.nlm.nih.gov/WRAMC/VC Training Web Manual/cases/VID_289_VC_213M/index.html. The colon data set used for Figs. 12a, 12b, and 14 was provided by the UK National Health Service. The authors would like to thank Stefan Bruckner from the Institute of Computer Graphics and Algorithms, Vienna University of Technology, for his ideas and feedback and Elnaz Gederi from Biotronics3D Ltd. for her assistance with the CPR implementation. Finally, they would like to thank Dr. Alastair Forbes and his group from the University College London (UCL) Hospital for their clinical input and feedback.

REFERENCES

[1] A. Kanitsar, D. Fleischmann, R. Wegenkittl, P. Felkel, and M.E. Gröller, "CPR—Curved Planar Reformation," *Proc. IEEE Conf. Visualization (VIS '02)*, 2002.

[2] R.S. Avila, L.M. Sobierajski, and A.E. Kaufman, "Towards a Comprehensive Volume Visualization System," *Proc. IEEE Conf. Visualization (VIS '92)*, pp. 13-20, 1992.

[3] S. Lakare and A. Kaufman, "Light Weight Space Leaping Using Ray Coherence," *Proc. IEEE Conf. Visualization (VIS '04)*, pp. 19-26, 2004.

[4] S. You, L. Hong, M. Wan, K. Junyaprasert, A. Kaufman, S. Muraki, Y. Zhou, M. Wax, and Z. Liang, "Interactive Volume Rendering for Virtual Colonoscopy," *Proc. IEEE Conf. Visualization (VIS '97)*, pp. 433-446, 1997.

[5] U. Tiede, N. von Sternberg-Gospos, P. Steiner, and K.H. Höhne, "Virtual Endoscopy Using Spherical QuickTime-VR Panorama Views," *Studies in Health Technology and Informatics*, vol. 85, pp. 523-528, 2002.

[6] L. Hong, S. Muraki, A. Kaufman, D. Bartz, and T. He, "Virtual Voyage: Interactive Navigation in the Human Colon," *Proc. 24th Ann. Conf. Computer Graphics and Interactive Techniques (Siggraph '97)*, pp. 27-34, 1997.

[7] A. Laghi, R. Iannaccone, I. Carbone, C. Catalano, E.D. Giulio, A. Schillaci, and R. Passariello, "Detection of Colorectal Lesions with Virtual Computed Tomographic Colonography," *Am. J. Surgery*, vol. 183, no. 2, pp. 124-131, 2002.

[8] A. Vilanova i Bartrolí, R. Wegenkittl, A. König, and E. Gröller, "Nonlinear Virtual Colon Unfolding," *Proc. IEEE Conf. Visualization (VIS '01)*, pp. 411-420, 2001.

[9] A. Vilanova i Bartrolí, R. Wegenkittl, A. König, E. Gröller, and E. Sorantin, "Virtual Colon Flattening," *Proc. Joint Eurographics/IEEE Trans. Visualization and Computer Graphics Symp. Visualization (VisSym '01)*, pp. 127-136, 2001.

[10] S. Haker, S. Angenent, A. Tannenbaum, and R. Kikinis, "Non-Distorting Flattening for Virtual Colonoscopy," *Proc. Third Int'l Conf. Medical Image Computing and Computer-Assisted Intervention (MICCAI '00)*, 2000.

[11] S. He, R. Dai, B. Lu, C. Cao, H. Bai, and B. Jing, "Medial Axis Reformation: A New Visualization Method for CT Angiography," *Academic Radiology*, vol. 8, pp. 726-733, 2001.

[12] M. Levoy, "Display of Surfaces from Volume Data," *IEEE Computer Graphics and Applications*, vol. 8, no. 3, pp. 29-37, 1988.

[13] M. Straka, M. Cervenansky, A. La Cruz, A. Köchl, M. Šrámek, M.E. Gröller, and D. Fleischmann, "The VesselGlyph: Focus & Context Visualization in CT—Angiography," *Proc. IEEE Conf. Visualization (VIS '04)*, pp. 385-392, 2004.

[14] F. Klok, "Two Moving Coordinate Frames for Sweeping along a 3D Trajectory," *Computer Aided Geometric Design*, vol. 3, no. 3, pp. 217-229, 1986.

[15] T. Vrtovec, B. Likar, and F. Pernus, "Curved Planar Reformation of CT Spine Data," *Proc. SPIE—Medical Imaging 2005: Image Processing*, vol. 5747, pp. 1446-1456, 2005.

[16] A. Kanitsar, R. Wegenkittl, P. Felkel, D. Fleischmann, D. Sandner, and M.E. Gröller, "Computed Tomography Angiography: A Case Study of Peripheral Vessel Investigation," *Proc. IEEE Conf. Visualization (VIS '01)*, pp. 477-480, 2001.

[17] M. Levoy, "Efficient Ray Tracing of Volume Data," *ACM Trans. Graphics*, vol. 9, no. 3, pp. 245-261, 1990.

[18] M. Wan, Q. Tang, A. Kaufman, Z. Liang, and M. Wax, "Volume Rendering Based Interactive Navigation within the Human Colon," *Proc. IEEE Conf. Visualization (VIS '99)*, pp. 397-400, 1999.

[19] G. Marmitt, A. Kleer, I. Wald, H. Friedrich, and P. Slusallek, "Fast and Accurate Ray-Voxel Intersection Techniques for Iso-Surface Ray Tracing," *Vision, Modeling, and Visualization*, B. Girod, M. Magnor, and H.P. Seidel, eds., Akademische Verlagsgesellschaft Aka, 2004.

[20] S. Parker, P. Shirley, Y. Livnat, C. Hansen, and P. Sloan, "Interactive Ray Tracing for Isosurface Rendering," *Proc. IEEE Conf. Visualization (VIS '98)*, pp. 233-238, 1998.

[21] M.E. Gröller, "Nonlinear Raytracing—Visualizing Strange Worlds," *Visual Computer*, vol. 11, no. 5, pp. 263-274, 1995.

[22] G. Wang, S.B. Dave, B.P. Brown, Z. Zhang, E.G. McFarland, J.W. Haller, and M.W. Vannier, "Colon Unraveling Based on Electrical Field: Recent Progress and Further Work," *Proc. SPIE*, vol. 3660, no. 1, pp. 125-132, 1999.

[23] R. Adams and L. Bischof, "Seeded Region Growing," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 16, pp. 641-647, 1994.

[24] A. Mehnert and P. Jackway, "An Improved Seeded Region Growing Algorithm," *Pattern Recognition Letters*, vol. 18, pp. 1065-1071, 1997.

David Williams received the bachelor's and master's degrees in computer science from the University of Warwick, United Kingdom. He is now a member of the Centre for Health Informatics, City University, London, where he is studying computer graphics and medical visualization for the PhD degree. He also works closely with Biotronics3D Ltd. to apply research to real-world applications.

Sören Grimm received the MS degree in computer science from the Eberhard Karls University of Tübingen, Germany, and the PhD degree in computer science from the Vienna University of Technology, Austria. He is the chief technology officer and director of research and development at Biotronics3D Ltd., London. His research interests include computer graphics, volume visualization, and medical visualization.

Ernesto Coto received the BS and MS degrees in computer sciences from the Central University of Venezuela (UCV) and is currently a PhD student. He is a faculty member of the Computer Sciences Department, UCV. He is a key member of the Computer Graphics Group, UCV. He is currently a visiting researcher at Biotronics3D Ltd. His main research interests are medical visualization and segmentation.

Abdul Roudsari is the director of the Centre for Health Informatics (CHI), City University, United Kingdom. He has been a member of the center since 1988, having had particular involvement in a range of research projects concerned with the provision of decision support systems to diabetic patients. Recently, he has led a major European Union (EU)-funded project evaluating telecare in the home setting.

Haralambos Hatzakis is responsible for all aspects of Biotronics3D Ltd. including marketing, sales, customer services, project execution, and product development. He has spent his professional working years building on a foundation grounded in medical technology, progressing from research and development through product management to strategic global marketing of diagnostic imaging solutions.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/publications/dlib.

Appendix B

Volumetric CPR as an Enhancement to Virtual Colonoscopy Systems

This paper was presented at the 21st International Conference on Computer Aided Radiology and Surgery (CARS 2007) in Berlin, Germany where it was well received. It omitted the technical details described in the previous paper and instead focused on the application of the Volumetric CPR to the problem of screening for colon cancer. It described the integration of the Volumetric CPR with an existing virtual colonoscopy system, the use of the surface coverage and translucency tools, and results pertaining to the effectiveness of the tool as compared to a virtual flythrough. The paper is also available online from SpringerLink.

Volumetric CPR as an Enhancement to Virtual Colonoscopy Systems

D. Williams[a], S. Grimm[b], E. Coto[b], H. Hatzakis[b], A. Roudsari[a]

[a]City University, London, UK
[b]Biotronics3D Ltd., London, UK

Abstract. Volumetric CPR is a new visualisation technique for curved tubular structures within the body. It works by splitting a structure lengthwise and presenting it to the user as two or more halves, each containing a detailed volume rendering of the inside of the structure and a greyscale rendering providing context information. In this work we look at the integration of the Volumetric CPR with an existing virtual colonoscopy system and the advantages which it can bring.

We demonstrate that the Volumetric CPR is a useful tool for displaying additional information not typically available during a flythrough, such as real-time surface coverage data or translucency rendering. We also show that, because the Volumetric CPR provides and alternative view on the colon, it increases surface coverage from 86.8% (for a flythrough in each direction) to 99.2%; significantly improving the chances of detecting abnormalities.

Keywords: virtual colonoscopy; visualization

1. Purpose

Volumetric Curved Planar Reformation (Volumetric CPR) [1] is a new approach to the examination of curved tubular structures inside the body. Compared to traditional visualisation approaches [2, 3] it has numerous advantages in terms of image clarity, surface coverage, and examination time. Our previous work described the technical implementation of the technique but did not give details of its application. The purpose of this work, therefore, is to explore the benefits it can bring to one particular domain – namely that of virtual colonoscopy.

While it is widely acknowledged that virtual colonoscopy presents a much more pleasant experience for the patient (compared to optical colonoscopy), concerns remain about the effectiveness of the technique [4, 5]. Many of these concerns relate to the surface coverage of a flythrough [6, 7]; if polyps are missed then this can have potentially serious consequences. Other problems include the contrast enhanced fluid (typically taken to aid the electronic segmentation) obscuring polyps or creating false positives.

We intend to demonstrate that Volumetric CPR can help address these issues when integrated with more traditional approaches.

2. Method

A Volumetric CPR works by splitting the colon lengthwise and presenting it to the user as two halves. This is shown in Figure 1; the Volumetric CPRs at the bottom are synchronised with the other views such that, as the camera moves through the colon, the Volumetric CPR images scroll from right to left.

There are a number of benefits to this setup. Firstly, the Volumetric CPR allows the user to see much further ahead than they are able to by using the flythrough alone. They are also able to see behind them – if they are concerned that something of interest may have been missed then it is possible to glance at the Volumetric CPR to check without needing to interrupt the flythrough. Because each piece of the colon surface is visible to the user for a longer period of time there is also a greater chance that any abnormalities will be noticed. Secondly, the Volumetric CPR can be used as a navigational tool. Clicking any point on it will cause the other views to jump to that point in the colon allowing a more detailed analysis.

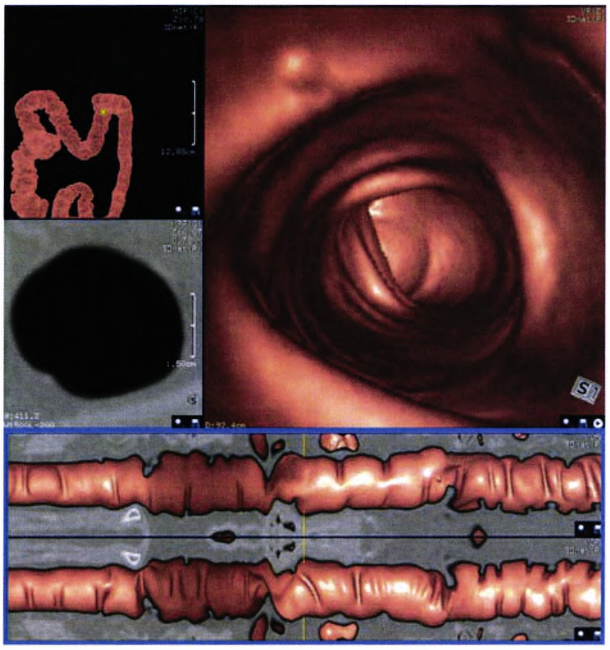


Fig. 1. The layout of the colonoscopy system including an overview (top-left), flythrough (top-right), and MPR (center-left). The two Volumetric CPRs (one looking up, one looking down) are at the bottom marked in the blue rectangle. The current position in the flythrough corresponds to the yellow marking in the centre of the Volumetric CPR. The area marked in purple on the Volumetric CPRs corresponds to what can currently be seen in the flythrough.

Further benefits are provided by the greyscale part of the Volumetric CPR. This shows what is beyond the surface and can provide useful context information. For example, when contrast enhanced fluid is present it is possible for a thin layer of fluid to stick to part of the colon surface – potentially hiding polyps underneath. This is difficult to detect in a flythrough because the fluid just shows up as part of the surface. In the Volumetric CPR, however, it shows up as a bright white boundary where the color rendering meets the greyscale image.

In addition to the basic uses of the Volumetric CPR illustrated so far, it is also possible to add overlays to convey further information. Actually, this capability has already been seen in Figure 1 where it was used to indicate what the flythrough is currently seeing. More generally, it can keep track of all parts of the surface seen so far (see Figure 2). Once the flythrough is complete the user can determine whether any regions have been missed and, if so, simply clicking the region on the Volumetric CPR will cause all views to jump to that location.

As well as monitoring the surface coverage of the flythrough, the Volumetric CPR actually increases coverage by providing an alternative viewing direction. It is often the case that polyps are occluded by folds in the colon but the Volumetric CPR is able to display many regions which would otherwise be missed.

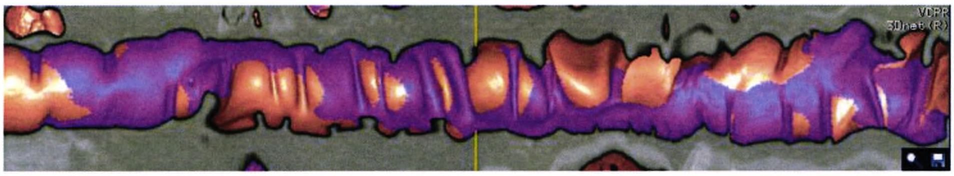
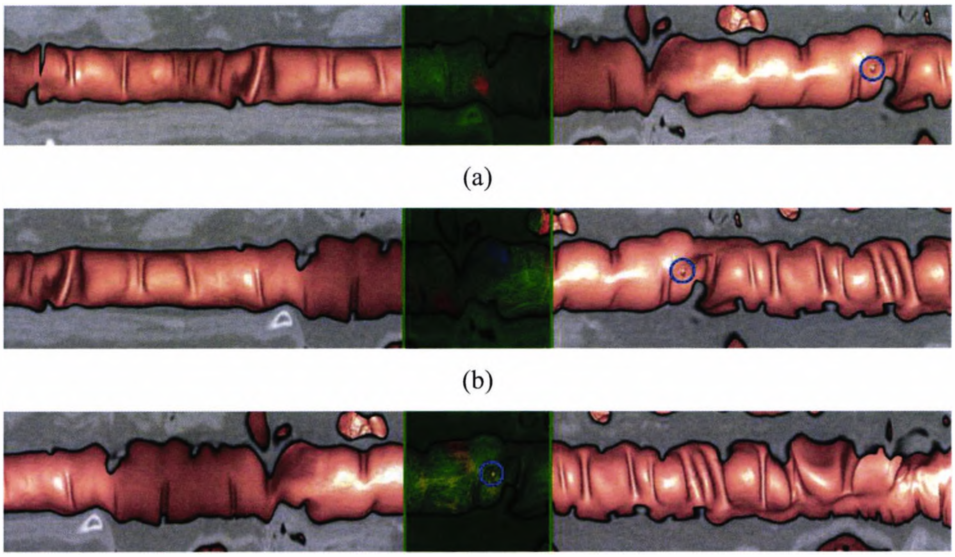


Fig 2. The Volumetric CPR can be used to visualise what regions have been seen by the flythrough. In this case a flythrough has been completed in one direction and it can be seen that large areas have been missed. Repeating the flythrough in the reverse direction will reduce the problem but not solve it completely.

A further problem caused by contrast enhanced fluid is the possibility of false positives due to tagged material in the colon. For example, Figure 3(a) shows what appears to be a polyp on the surface of the colon. In [8] a technique is described for performing 'translucent rendering' which allows the user to determine the density of material just behind the surface; tissue shows up red or green while fluid shows up white. Unfortunately this interrupts the workflow when applied to a flythrough because the user is required to stop, switch on translucent rendering to examine the structure, switch back to normal rendering, and resume the flythrough. It integrates much more elegantly with the Volumetric CPR because we simply provide a 'translucency band' which all of the surface passes through at some point. Applying this to the regions around the 'polyp' reveals that it is actually just a deposit of fluid (Figure 3(c)).


(a)

(b)

(c)

Fig 3. A suspicious structure is identified in the colon (marked by a blue ring in (a)). As the flythrough proceeds, the Volumetric CPR moves from right to left (b). As it passes under the translucency band the inside of the structure can be seen to have a high density (c) and it is therefore a deposit of fluid rather than a polyp.

3. Results

The technique was demonstrated to the gastrointestinal department of University College London Hospital and has received positive comments and feedback. The early stage trial has proven that Volumetric CPR is a viable technique and, when combined with other approaches such as a flythrough, can help provide more of the information required when assessing a patient's condition. We are planning to conduct a larger scale clinical trial with the group based on this and similar techniques for the analysis of gastrointestinal diseases.

We have also obtained quantitative results relating to the surface area which is seen by the flythroughs and the Volumetric CPRs. A flythrough was performed in each direction and sets of 2 (up and down) and 4 (up, down, left and right) Volumetric CPRs were generated for each of 3 colon datasets. The results are shown in Table 1.

Table 1
The percentage of the surface seen by the flythroughs and the Volumetric CPRs in various combinations.

	Dataset 1	Dataset 2	Dataset 3	Average
Flythrough (Antegrade)	62.2%	69.6%	65.1%	65.6%
Flythrough (Retrograde)	61.9%	66.1%	64.5%	64.2%
Flythrough (Both Directions)	84.8%	89.8%	85.9%	86.8%
Volumetric CPR (Up + Down)	76.1%	82.8%	74.1%	77.6%
Volumetric CPR (Left + Right)	76.3%	84.3%	74.8%	78.5%
Volumetric CPR (All Directions)	97.9%	98.8%	96.5%	97.7%
Flythrough (Both) + Volumetric CPR (All)	99.3%	99.6%	98.8%	99.2%

4. Conclusion

Even without overlays, the Volumetric CPR is a valuable extension to the traditional virtual colonoscopy package. It can be used to allow more of the colon to be seen at once, to provide a navigational aid, and to increase surface area which is seen. It the latter case it increases the surface coverage from 86.8% to 99.2% on average. This significantly improves the chance that polyps are detected.

The addition of overlays allows the Volumetric CPR to become a powerful tool for monitoring surface coverage or for analysing the composition of a surface. This adds significant diagnostic value to the technique and allows a more streamlined workflow. Future work will likely focus on the development of further overlays such as curvature information or a display of the results of computer aided polyp detection.

References

[1] Williams D. Grimm S. Coto E. Hatzakis H. Roudsari A. Volumetric Curved Planar Reformation for Virtual Endoscopy. IEEE Trans Vis Comput Graph (accepted pending minor revisions) 2007

[2] You S. Hong L. Wan M. et al. Interactive Volume Rendering for Virtual Colonoscopy. Proceedings of IEEE Visualization 1997

[3] Vilanova i Bartroli A. Wegenkittl R. König A. et al. Virtual Colon Flattening. Proceedings of VisSym '01 Joint Eurographics - IEEE TVCG Symposium on Visualization 2001 p. 127-36

[4] Cotton P. Durkalski V. Pineau B. et al. Computed tomographic colonography (virtual colonoscopy): a multicenter comparison with standard colonoscopy for detection of colorectal neoplasia. JAMA 2004; 291(14) p. 1713-9

[5] Pescatore P. Glücker T. Delarive J. et al. Diagnostic accuracy and interobserver agreement of CT colonography (virtual colonoscopy). Gut 2000; 47 p.126-30

[6] Samara Y. Dachman A. Hoffmann K. Surface coverage in CT colonography studies. Proc SPIE 1999; 3660 p.117-24

[7] He T. Hong L. Chen D. Liang Z. Reliable Path for Virtual Endoscopy: Ensuring Complete Examination of Human Organs. IEEE Trans Vis Comput Graph 2001; 7(4) p.333-342

[8] Pickhardt P. Translucency Rendering in 3D Endoluminal CT Colonography: A Useful Tool for Increasing Polyp Specificity and Decreasing Interpretation Time. AJR Am J Roentgenol 2004 p. 429-36

Appendix C

O-Buffer Based IFT Watershed from Markers for Large Medical Datasets

This paper presents our optimization to the watershed algorithm based on O-Buffers, and has been accepted for publication in ACM Computers & Graphics. It describes how the use of O-Buffers can reduce the memory footprint required for maintaining the cost and label volumes and so allow larger medical images to be segmented on common hardware. Using this algorithm is one way we can segment the colon for use in the virtual flythrough and Volumetric CPR. The paper is not yet available online.

O-Buffer Based IFT Watershed from Markers for Large Medical Datasets

Ernesto Coto[a,*,1], Sören Grimm[b] David Williams[c]

[a] Computer Graphics Lab, School of Computer Sciences, Faculty of Sciences, Central University of Venezuela, 47002, Los Chaguaramos 1041-A. Caracas, Venezuela

[b] Biotronics3D Ltd., The Gatehouse, Trinity Buoy Wharf, 64 Orchard Place, London, E14 0JW, UK

[c] Centre for Health Informatics in Medicine, School of Informatics, City University, Northampton Square, London, EC1V 0HB, UK

Abstract

The watershed transform from markers is a very popular image segmentation operator. The Image Foresting Transform (IFT) watershed is a common method to compute the watershed transform from markers using a priority queue, but which can consume too much memory when applied to three-dimensional medical datasets. This is a considerable limitation on the applicability of the IFT watershed, as the size of medical datasets keeps increasing at a faster pace than physical memory technologies develop. This paper presents the O-IFT watershed, a new type of IFT watershed based on the O-Buffer framework, and introduces an efficient data representation which considerably reduces the memory consumption of the algorithm. In addition, this paper introduces the O-Buckets, a new implementation of the priority queue which further reduces the memory consumption of the algorithm. The new O-IFT watershed with O-Buckets allows the application of the watershed transform from markers to large medical datasets.

Key words: Watershed transform, image segmentation, graphic data structure, image foresting transform.
PACS: 07.05.Pj, 89.20.Ff

---

1. Introduction

Modern medical imaging technologies allow extremely detailed and complex medical studies, for which the use of computers in facilitating their processing and analysis has become necessary. For instance, algorithms which segment anatomical structures and other regions of interest are a key component in assisting and automating medical diagnosis. A very popular technique for such a segmentation task is the watershed transform [1][2].

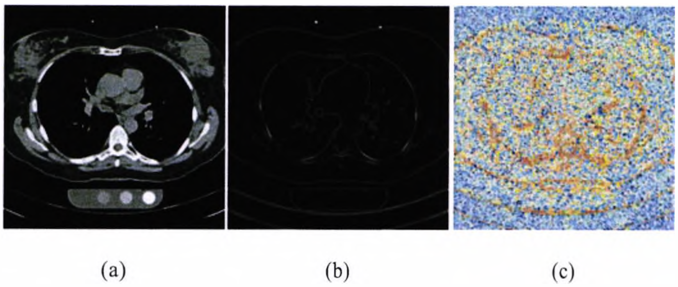
The principle of the watershed transform is to consider a grayscale image as a topographic relief which is then flooded by water coming out of holes pierced in local minima. As the water level rises it forms catchment basins, and dams are built where waters coming from different basins meet. When the water level has reached the highest peak in the landscape, the process stops. As a result, the landscape is partitioned into basins separated by dams known as *watersheds*. Depending on the application, the desired result is either the watersheds or the resulting basins. When applied to the gradient image, the catchment basins produced by the watershed transform are expected to match the homogeneous gray level regions of the image. However, with medical images the transform often produces a *watershed over-segmentation*, where a large number of watersheds are obtained due to noise or local irregularities in the gradient image. This issue is illustrated in Figure 1 using a Computed Tomography (CT) slice. In general, a watershed transform of Figure 1a, is expected to contain catchment basins clearly separated by the strong borders shown in Figure 1b. Nevertheless, the transform results in the over-segmentation shown in Figure 1c, due to the noise typically present in CT scans.

Introducing markers in the original image [3][4] to reduce the number of catchment basins is the most common approach to reduce over-segmentation. Markers can be seen as holes in the image relief where water can enter as the relief is flooded. Each marker is associated with a label. As the relief is uniformly flooded, water with different labels may meet but cannot be mixed. When the flooding process stops, each region of water defines a catchment basin associated with a marker. This concept is illustrated in Figure 2 using the same CT slice employed in the previous example.

---

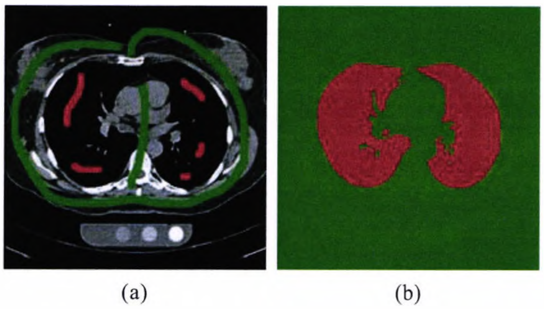
* Corresponding author.
Email addresses: `ecoto@ciens.ucv.ve` (Ernesto Coto), `sgrimm@biotronics3d.com` (Sören Grimm), `D.P.Williams@city.ac.uk` (David Williams).

[1] Phone/Fax:++58(212)6934243.



(a) (b) (c)

Fig. 1. Example of watershed over-segmentation. (a) Original image. (b) Image gradient. (c) Watershed transform.

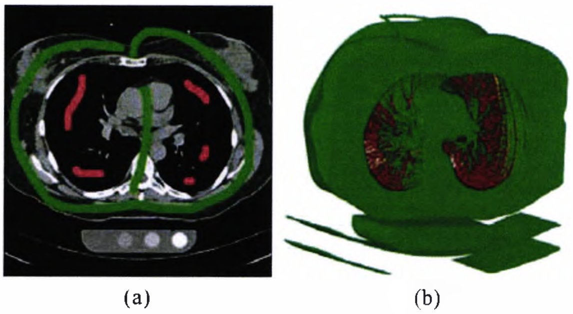


(a) (b)

Fig. 2. Example of watershed from markers. (a) CT slice with markers. (b) Watershed transform.

Figure 2a shows red markers placed with the intention of segmenting the lungs, while the green markers are placed with the intention of producing a second catchment basin which segments the rest of the image. Figure 2b shows the resulting watershed transform from markers. In this paper only the catchment basins are important since they decompose the image in different regions; the watersheds are discarded.

Many algorithms have been developed to compute watershed transforms, see [5] for a survey. They can be divided into parallel algorithms and sequential algorithms. Parallel watershed algorithms, such as [6][7], are out of the scope of this paper. Sequential watershed algorithms can be further divided into two classes, one based on the specification of a recursive algorithm by Vincent and Soille [8], and another based on distance functions by Meyer [9]. This paper focuses on the algorithm by Lotufo and Falcão [10], which simulates a recursive flooding of water in the image using a priority queue of pixels, similar to Vincent and Soille [8]. Lotufo and Falcão's [10] algorithm obtains a watershed transform from markers based on the Image Foresting Transform (IFT) framework, hereafter the *IFT watershed* algorithm. The IFT watershed can be applied to digital images and it has a very high resolution, as it can correctly segment regions separated by only one pixel. Furthermore, the solution of the IFT watershed can be obtained in linear time by using the linear-time priority queue presented in [11]. The algorithm is also extensible to n-dimensions and it has been already applied successfully to two-dimensional [12][13] and three-dimensional images [14][15]. For example, using the markers shown in Figure 3a, the IFT watershed can be applied to a stack of CT slices. Figure 3b shows a three-dimensional reconstruction of the resulting basins.



(a) (b)

Fig. 3. Example of three-dimensional watershed from markers. (a) CT slice with markers. (b) Three-dimensional reconstruction of resulting catchment basins.

However, the linear-time priority queue is a static multidimensional array which can consume a lot of memory. Felkel et al. [15] applied the IFT watershed to Computed Tomography and Magnetic Resonance (MR) images of different body parts with the largest dataset being 512x512x169 voxels in size, for which the linear-time priority queue consumed 433.6 MB of memory. However, current medical imaging technology can produce datasets with a size of 1024^3 voxels [16] or larger. For such a dataset, Felkel's results suggest that their linear-time priority queue would consume approximately 10.26 GB. This high memory consumption is particularly a problem for 32-bit computer systems, considering that their maximum virtual address space is about 3 GB. This is not the case for 64-bit systems since the virtual address space is considerably larger, but several compilers and programming languages still have a 2 GB size limit for static arrays, even in 64-bit machines. This limit can be exceeded with dynamically allocated arrays, but this could cause the system to start paging to disk prematurely, ruining the performance of the algorithm. Furthermore, the increased size of pointers in a 64-bit machine further increases the memory requirements of the linear-time priority queue. Felkel et al. [15] also presented a more memory-efficient version of the queue, which reduces its memory consumption to a range of 19 to 45% of the memory consumed by the linear-time priority queue. However, they also stated that the largest dataset this implementation can handle is a 1024x1024x2048 dataset, since their data structure can only store 2^{31} different positions. Since the size of medical datasets is increasing faster than physical memory as the technology of acquisition devices evolves, alternative memory-efficient data structures are required for segmentation operators such as the IFT watershed.

This paper presents an implementation of the IFT watershed based on O-Buffers [17], introducing an efficient representation of the data structures needed during execution, considerably reducing its memory consumption and allowing its application to medical images with a size beyond Felkel's maximum data size. The paper is organized as follows: Section 2 presents a brief overview of the IFT watershed and the related work. Section 3 presents the new IFT watershed algorithm based on O-Buffers. Section 4 presents a new representation of the priority queue also based on the O-Buffer framework, followed by a study of the complexity of the new IFT watershed algorithm in Section 5. Section 6 describes tests and results. Conclusions and future work are presented in Section 7.

2. Related Work

This section reviews previous work related to the IFT watershed, starting with the concept of the Image Foresting Transform (IFT) and the IFT watershed algorithm. This is followed by a brief discussion of the algorithm and previous work related to it. The notation and definitions introduced in this section are extensively used in the rest of the paper.

Let $G=(V,E)$ be an undirected weighted graph with node set V and arc set E. Let L be a subset of V containing a set of root nodes and $w(p,q)$ be the weight of any arc (p,q) in E. If the path weight for any two nodes is a non-decreasing function of the arc weights in the path, then the *shortest-path forest problem* finds, for each graph node, the path with the lowest weight connecting it to its nearest root node in L. The weights can be called 'costs', and the shortest path can be called a 'minimum-cost path', without loss of generality.

The Image Foresting Transform (IFT) [13] defines a minimum-cost path forest in a graph G, where V corresponds to a discrete image and the arcs in E are defined by an adjacency relationship between nodes. The cost of a path in G is determined by an application-specific path-cost function, which depends on local image properties. The root nodes in L are called *markers*, and each marker is associated with a label. Lotufo and Falcão [10] found that the problem of computing the watershed from markers of a grayscale image can be reduced to the minimum-cost path forest problem in the IFT framework. They presented Algorithm 1 as a solution that uses the IFT to simulate the flooding of water in the image using a priority queue.

Algorithm 1 The IFT watershed algorithm

```
 1: // Initialization stage
 2: for all non-marker nodes p in L do
 3:    C(p) = infinity;
 4: for all marker nodes p in L do
 5:    C(p) = 0;
 6:    Enqueue p with cost 0;
 7: // Propagation stage
 8: while queue not empty do
 9:    q = Dequeue node with minimum cost;
10:    for each p neighbor of q such that C(p) > C(q) do
11:       if max{ C(q),w(q,p) } < C(p) then
12:          if C(p)≠infinity then
13:             Dequeue p;
14:          C(p) = max{ C(q),w(q,p) };
15:          L(p) = L(q);
16:          Enqueue p with cost C(p);
```

In the IFT watershed shown in Algorithm 1, $L(p)$ is the value of the root of p in L, and $C(p)$ is the cost of the path from the nearest root to p. The arc weight is the non-negative dissimilarity function $w(p,q)=|f(p)-f(q)|$, where $f(p)$ corresponds to the value of node p in the input image. When the IFT watershed algorithm finishes, C contains the costs of all minimum-cost-paths connecting a node to its nearest marker and L contains the result of the watershed partitioning. The algorithm can be applied to digital images and it is extensible to n-dimensions.

During the initialization stage of the IFT watershed all nodes are assigned a cost of *infinity*, except for the root nodes which are assigned a cost of zero, see Algorithm 1. During each iteration of the propagation stage, the algorithm dequeues a node with minimum cost from the priority queue and propagates its label to those neighbors with a cost higher than the cost of the dequeued node. This avoids processing neighbors which might have already reached their minimum cost. When a label is propagated over a neighbor, the neighbor becomes eligible for propagation and therefore it must be inserted in the priority queue. However, the cost of the neighbor is compared with *infinity* before this, for nodes with a cost different than *infinity* have been already inserted in the queue. In such a case, the previous entry of the neighbor is removed before inserting the neighbor again with the new cost. Since the path cost is a non-decreasing function, when a node is dequeued in line 9 of the IFT watershed its path cost is the final optimal path cost. The algorithm terminates when the priority queue is empty.

A possible variation of Algorithm 1 is reported in [18], given that the arc weights are computed using the dissimilarity function. The variation consists of removing lines 12 and 13 of the algorithm, thereby allowing a node to be inserted in the queue more than once. In this situation, when a node is dequeued in line 9 of the IFT watershed, a test is required to certify whether the node has already been labeled or not. With this approach, the first instance of the node removed from the queue is the one with the lowest priority and its label is permanently stored in L. Further instances of the same node have no effect, since the node has already been labeled. This non-dequeuing variation of the IFT watershed could be used to save the time spent on searching and removing the node in the priority queue, although it results in a longer priority queue and the uniquity of the nodes in the queue is lost. However, this is not practical if the arc weight function is a gradient function, as originally suggested in [10]. In such a case, Lotufo and Falcão explained that the IFT watershed algorithm does not require re-evaluation of the node values because the nodes are inserted in the queue with their optimum cost. Then, lines 12 and 13 of Algorithm 1 as well as the temporary costs in C are not necessary, since the nodes are dequeued already with their minimum cost, making the algorithm simpler. However, the dissimilarity function achieves higher resolution than the use of morphological gradients [19] and it can be used for color images. In addition, using

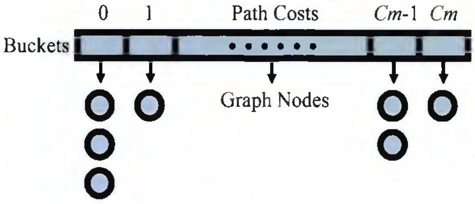


Fig. 4. Priority queue based on buckets. The buckets are arranged linearly and sorted by cost. Each bucket is associated with a list of graph nodes.

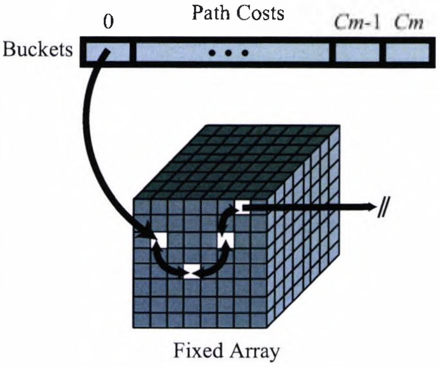


Fig. 5. Three-dimensional linear-time priority queue for the IFT watershed. Four nodes have been enqueued in bucket 0.

the dissimilarity function the path costs are integer numbers within a limited range, allowing the use of *buckets* to implement the priority queue as suggested by Dial [20].

A bucket is a container of all graph nodes with equal path cost, where the nodes are stored in any order. The priority queue can then be represented as an ordered set of buckets, where the nodes in the first non-empty bucket are the nodes with the highest priority. This is illustrated in Figure 4, where the priority queue is represented as a linear arrangement of $Cm+1$ buckets containing the nodes in G, with Cm being the maximal arc weight in E. Each bucket k stores a list of all nodes whose cost is equal to k. Falcão et al. [11] exploited Dial's idea to create the linear-time priority queue. This data structure consists of a static array of $Cm+1$ buckets, where the list of elements in each bucket is implemented as a double-linked list of nodes, stored in a fixed multidimensional array of size $|V|$. Figure 5 illustrates their structure for the three-dimensional case, showing four nodes stored in bucket 0 as an example. Using this structure, dequeuing a node and updating the cost of a node are $O(1)$ operations. The enqueuing operation is also an $O(1)$ operation as long as the node is inserted as the first element in the bucket. Locating the bucket with minimum cost could have a complexity of $O(Cm)$ in the worst case. Using the linear-time priority queue, the time complexity of the IFT watershed is $O(|E|+|V|\times Cm)$ [13]. Note that in this case a non-dequeuing version of the IFT is not necessary, since the queue operations have a complexity of $O(1)$.

Felkel et al. [15] presented five possible implementations of the IFT watershed for three-dimensional medical images. Their first implementation employed a variation of the structure illustrated in Figure 5, where a second array of $Cm+1$ pointers is used. Each entry in the array points to the last node of its corresponding bucket. This array is required to maintain the $O(1)$ complexity of the enqueuing operation, since Felkel's implementation inserts the new node at the end of the double-linked list. The nodes in Felkel's linear-time priority queue stored the label of the node (1 bit), its cost (2 bytes) and the pointers to the next and previous nodes in the list (4 bytes each). Each node also stored a boolean flag (1 bit) to mark the node as either 'temporarily labeled' or 'permanently labeled', as originally suggested in [10]. This is a total of $(10+\frac{1}{4})$ bytes per node. Therefore, the fixed three-dimensional array consumes $|V|\times(10+\frac{1}{4})$ bytes and the two fixed arrays of pointers consumes another $4\times(Cm+1)+4\times(Cm+1)$ bytes. Since Felkel uses $w(p,q)=|f(p)-f(q)|$, then $Cm=2^{12}-1=4095$ is the maximum possible cost for 12-bit medical datasets. In such a case, the two arrays consume a total of $2\times4\times4\ 096=32\ 768$ bytes. Therefore, Felkel's structures for the first implementation consume a total of $(|V|\times(10+\frac{1}{4}))+$ 32 768 bytes. If the input image is stored in a fixed three-dimensional array consuming $|V|\times2$ bytes, the fixed volume of Felkel's first implementation consumes $(10+\frac{1}{4})/2\approx5$ times the memory consumed by the input image, plus the 32 768 bytes required for the two arrays. This is a memory overhead of more than 500% with respect to the input data.

The rest of Felkel's five implementations aimed to reduce the memory consumption of the linear-time priority queue. The fifth implementation was the best of all, which consisted of a non-dequeuing IFT watershed. This implementation required the fixed three-dimensional array, but only to store the flags and the final labels. Each bucket consisted of a dynamic single-linked list containing temporary labels and node positions. The costs were not stored at all, since the cost of a node was intrinsic to the bucket in which it was stored. In order to save the memory used for pointers in the dynamic list, a *queue element grouping mechanism* was used. Up to 256 elements were grouped into one memory chunk, and each group had one pointer to the next group and two local indices for the first and the just-past-the-last elements of the queue section stored in the group. This implementation was the best of Felkel's variants, consuming from 19% of the memory consumed by the first implementation in a 256x256x444 dataset, to 45% in a 512x512x169 dataset. However, since the first implementation shows a memory overhead of more than 500% with respect to the input data, Felkel's fifth implementation could show a memory overhead of more than $500\%\times0.45=225\%$, which could still result in a high memory consumption. Furthermore, Felkel et al. [15] states that their implementation is limited to a dataset with a maximum size of 1024x1024x2048 voxels, as they use one bit for the label and 31 bits are left for the position. Given that datasets of 1024^3 voxels or larger are currently available, and that acquisition devices increase their resolution quite often, an alternative data structure is required for the IFT watershed.

This paper presents new data structures based on the

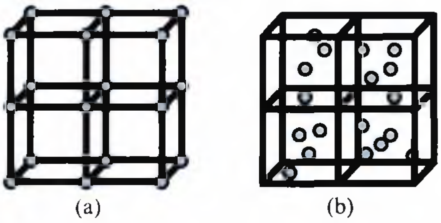

(a) (b)

Fig. 6. Sampling methods for three-dimensional images. (a) Samples on a conventional image. (b) Samples in an O-Buffer.

O-Buffer framework [17], which considerably reduce the memory consumption of the IFT watershed algorithm, allowing its application to datasets with a size beyond Felkel's maximum data size. This paper focuses on the three-dimensional case of the IFT watershed based on dissimilarities presented in Algorithm 1. In addition, the terms 'image', 'three-dimensional image' and 'dataset' are used interchangeably as this paper always refers to a three-dimensional image which is comprised of a stack of two-dimensional images. The new data structures based on the O-Buffer framework are presented in the next sections, and a new IFT watershed algorithm is introduced.

3. The O-Buffered IFT watershed algorithm

The linear-time priority queue described by Felkel et al. [15] for their first implementation could certainly consume a significant amount of memory if the input dataset is large enough. Therefore, the IFT watershed algorithm requires an alternative representation of the data so that it can be applied to the three-dimensional images that current acquisition devices are producing. This section presents a three-dimensional image representation based on the O-Buffer framework [17], which significantly reduces the memory consumption of the IFT watershed algorithm.

Definition 1 An *O-Buffer* is a generic image representation in which image samples are not restricted to equally spaced points in a grid, as opposed to conventional images, see Figure 6. The grid consists of grid-segments, and each grid-segment stores a list holding the position of the samples contained in it.

The O-Buffer employs the grid as a frame of reference to store the position of the samples. Each sample position inside a grid-segment is given by a series of offsets. Figure 7 illustrates this concept, where the front-lower-left corner of the grid-segment is used as a base position, and one offset in each direction is used to record the position of the sample inside the grid-segment. The position of a sample in the O-Buffer is then given by the position of the grid-segment plus the sample's offsets inside the grid-segment.

A three-dimensional medical image can be represented as an O-Buffer by dividing the image data into grid-segments of the same size and containing the same number of nodes, as shown in Figure 8a. Now, let the samples in the O-Buffer be defined by the nodes in set V, corresponding to the input

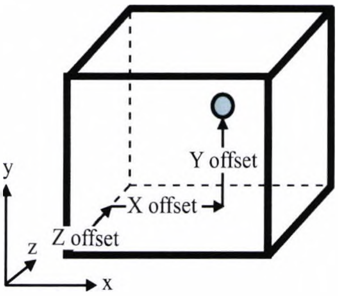


Fig. 7. Illustration of a sample inside a grid-segment. X, Y and Z offsets define its position inside the grid-segment.

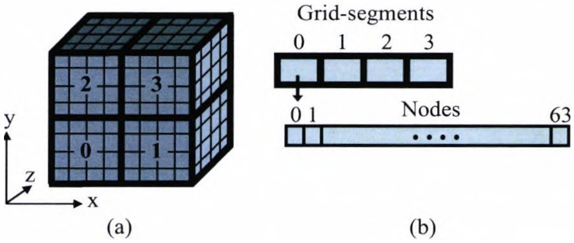


Fig. 8. Example of image subdivision into grid-segments. (a) Original 8x8x4 image divided in 4 grid-segments, in xyz order. (b) The 4 grid-segments are arranged linearly, as are their internal 64 nodes.

image of the IFT watershed. The O-Buffer concept enables the definition of a memory-efficient data structure for the IFT algorithm, hereafter referred to as an *O-Image*, see Definition 2.

Definition 2 An *O-Image* is an O-Buffer represented as a linear arrangement of grid-segments, each containing a linear arrangement of nodes, as shown in Figure 8b. Each grid-segment entry consists of a pointer to its corresponding linear arrangement of nodes and each node entry in this arrangement consists of a node value. All linear arrangements of nodes contain the same number of node values.

Using this data representation, the image is stored using a two-level hierarchical subdivision: a grid-segment level and a node level. Then, in order to access a node in the O-Image, two offsets are required. The first offset accesses the grid-segment containing the node and the second offset accesses the node inside the grid-segment. These positions are referred to as *O-Image positions*, see Definition 3.

Definition 3 An *O-Image position* in an O-Image, is a pair (a,b) in which a corresponds to the offset of the grid-segment where the node is contained in the O-Image, and b corresponds to the offset of the node within the linear arrangement associated with the grid-segment; a is referred to as a *grid-segment-offset*, and b is referred to as a *node-offset.*

When the O-Image is created, all necessary grid-segments are created and so is each linear arrangement of node values. Since all grid-segments always have the same number of node value entries, it is possible to employ a *copy-on-write* approach [21] for memory management. The copy-on-write is an optimization strategy used by many operating systems to optimize the storage of objects in memory. The key idea behind it is that if multiple processes request resources which are initially indistinguishable, they can be provided with the same resource. This sharing can be maintained until a process tries to modify the resource, at which point a private copy is created to prevent the changes from becoming visible to the other processes. If no process ever makes any modifications, no private copy is ever created, therefore expensive resources are saved. The same principle can be applied to the grid-segments in an O-Image. When two grid-segments contain identical nodes, then only one set of node values is actually needed. In such a case, the memory required by the set of nodes in one of the grid-segments could be saved by associating both grid-segments with the same set of node values. The sharing can be maintained until one of the grid-segments changes its node values. Using a copy-on-write strategy, the memory requirements of the O-Image decrease as the number of shared nodes increases. This memory utilization strategy makes the O-Image especially suitable for the storage of sparse images, in which large regions of homogeneous node values are present, since several grid-segments can be shared.

The IFT watershed algorithm as described by Lotufo and Falcão [10] can benefit tremendously from the O-Image. The IFT watershed iteratively divides the complete image into different catchment basins, by changing the node labels and costs on each iteration until the propagation stage ends, see Algorithm 1. During the propagation, numerous neighboring nodes are assigned the same label values, producing large regions of nodes with the same label and making the label set L suitable for an O-Image representation. Similarly, the nodes are also assigned costs. These costs are no longer used once a node has been permanently labeled, producing large regions of neighboring non-used cost values as the propagation stage of the IFT watershed algorithm proceeds. Then, these cost values can be set to 0, consequently producing large regions of nodes with the same cost value and making the cost set C suitable for an O-Image representation. Since more and more grid-segments are shared as the flooding process proceeds, L and C are stored in a very efficient way.

Let L and C be O-Images the same size as V and with the same grid-segment size. In addition, let the priority queue be represented as an ordered set of buckets as suggested in [11], and the arc weight be computed using the dissimilarity function. Algorithm 2 describes the new IFT watershed, which is hereafter known as the *O-IFT watershed.*

In comparison with the IFT watershed algorithm, the initialization of set C in the O-IFT algorithm is different, as stated in line 2 of Algorithm 2. Since all the nodes in C are to be initialized with the same value, the first grid-segment is initialized and then the following grid-segments are associated with the first. This is faster than initializing the nodes individually. Following this, only those nodes corresponding to the input markers are initialized in line 4, with cost 0.

Algorithm 2 The O-IFT watershed algorithm

```
 1: // Initialization stage
 2: Initialize C with cost infinity;
 3: for all marker nodes p in L do
 4:    C(p) = 0;
 5:    Insert p in bucket 0;
 6: // Propagation stage
 7: while buckets are not empty do
 8:    q = Retrieve first node in first non-empty bucket;
 9:    Increase DONE nodes in grid-segment of q;
10:    if all nodes are DONE in grid-segment of q then
11:       if grid-segment of q in L is homogeneous then
12:          Share grid-segment of q in L;
13:       Share grid-segment of q in C;
14:    for each p neighbor of q such that C(p) > C(q) do
15:       if max{ C(q),w(q,p) } < C(p) then
16:          if C(p)≠infinity then
17:             Remove p from bucket C(p);
18:          C(p) = max{ C(q),w(q,p) };
19:          L(p) = L(q);
20:          Insert p in bucket C(p);
```

In comparison with the IFT watershed algorithm, the propagation stage of the O-IFT algorithm requires more operations after the dequeuing of the node with minimum cost, see Algorithm 2. Line 9 keeps track of the number of nodes that have reached their minimum cost in each grid-segment, so that the O-IFT algorithm detects in line 10 when all the nodes in a grid-segment have been permanently labeled. In such a case, all nodes in L which are contained in the current grid-segment should already have been labeled. Then, the grid-segment is checked for homogeneity in line 11. All nodes associated with the grid-segment must have the same value for it to be considered homogeneous. If the grid-segment is homogeneous then it is shared in line 12. An internal homogeneous grid-segment is kept for each label, so this sharing operation is performed instantly. No attempt to share a non-homogenous grid-segment is performed, since such a task would be very time-consuming. A similar approach is applied to C, except that no checking for homogeneity of the grid-segment is necessary since the nodes in it are not needed past this stage. Then, the grid-segment is instantly shared in line 13 with an internal grid-segment with all node values set to 0.

Other IFT-based segmentation operators [13] require, in addition to the cost and the label of each node, also its predecessor in the optimum path. This information is stored so that other possible paths can be computed efficiently. The IFT watershed and the O-IFT watershed do not require this information because once an optimum path to a node has been found no other path is evaluated. In the case of the O-IFT algorithm, it would be impractical to store the predecessor map in an O-Image because it is very unlikely that numerous homogenous regions of values could be found in the image, reducing the effectiveness of the copy-on-write

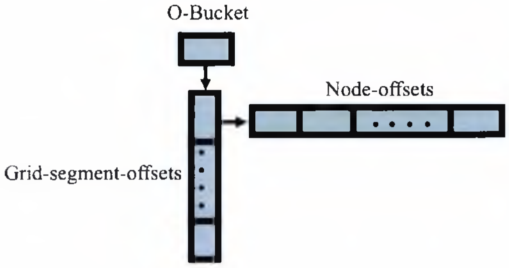


Fig. 9. Schematic drawing of an O-Bucket. The O-Bucket is associated with a set of grid-segment-offsets, each one associated with a set of node-offsets.

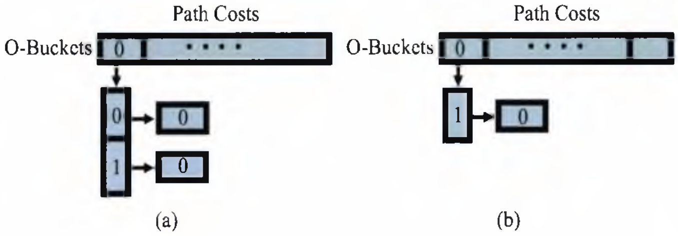


Fig. 10. Examples of operations in the O-Buckets. (a) Example of enqueuing showing insertion of nodes with positions (0,0) and (1,0) in O-Bucket 0. (b) Example of dequeuing showing removal of node (0,0) from the O-Bucket 0 shown in (a).

approach. This is not the case of the O-IFT watershed presented in this paper, where no predecessor map is required.

Using the O-Images in conjunction with the copy-on-write strategy, the O-IFT watershed keeps the information used for the segmentation process stored in a very efficient manner. In addition, an O-Image can store both two-dimensional and three-dimensional images, keeping the multi-dimensional nature of the original IFT watershed algorithm. The next section presents a new representation of the buckets based in the O-Buffer framework, which significantly reduces the memory consumption of the priority queue. The new buckets integrates with the O-IFT watershed, producing a memory efficient algorithm to compute the watershed transform from markers of a three-dimensional image.

4. The O-Buffered buckets

Lotufo and Falcão [11] proposed implementing the priority queue as an ordered list of *buckets* [20], see Figure 4. This section presents a new way to represent the buckets, thereby allowing a memory-efficient priority queue for the O-IFT watershed.

In practice, a bucket is nothing more than a container storing the actual graph nodes or some kind of reference to them. In the O-IFT watershed, the position of a node is uniquely defined by its grid-segment-offset and its node-offset. If the buckets are used to store O-Image positions, each bucket can contain positions of nodes stored in the same grid-segment, which can be grouped together by their grid-segment-offset. Therefore, the O-Image positions in each bucket can be represented as a set of grid-segment-offsets, each one associated with a set of node-offsets. This representation of a bucket is called an *O-Bucket*, see Definition 4.

Definition 4 An *O-Bucket* is a sorted map where the keys are grid-segment-offsets and each key is associated with a list of node-offsets, see Figure 9.

Given this definition of a bucket, the priority queue of the O-IFT watershed can be represented as a sorted map of O-Buckets, each one corresponding to a node cost. Note that an O-Bucket only stores O-Image positions grouped by their grid-segment-offset. It is not necessary to store the value of temporary labels or temporary costs since these values are stored in the O-Images L and C, see lines 18 and 19 of the O-IFT algorithm. Furthermore, although the O-Buckets store O-Image positions, cartesian coordinates can still be used given that they are transformed to O-Image positions before accessing the O-Image. This transformation can be performed instantly whenever a node value is accessed. Transforming O-Image coordinates to cartesian coordinates can also be done instantly. This is especially useful to compute the neighbors of p, since they are usually defined by a connectivity neighborhood around the node, which can be obtained from the position of p in cartesian coordinates. Felkel et al. [15], for instance, used a 6-connectivity neighborhood around p. Moreover, note that storing an O-Image position requires only two integer offsets rather than the three integers required for a cartesian coordinate. The following paragraphs describe how to perform the dequeuing and enqueuing operations in this new representation of the priority queue.

Enqueuing a node p requires computing both the grid-segment-offset of the grid-segment where p is contained, and also its node-offset inside the grid-segment. The node-offset is then inserted into the list corresponding to its grid-segment-offset, which in turn is in the O-Bucket corresponding to its cost. Figure 10a shows an example where nodes with O-Image positions (0,0) and (1,0) have been inserted in the O-Bucket corresponding to cost 0.

Dequeuing a node p is slightly more complicated. First, the O-Image position of p is computed and it is located in the O-Bucket corresponding to its cost. Then, the node-offset is removed from the list corresponding to the grid-segment of p. After removing the node-offset, it is necessary to determine whether the list is empty, so that the grid-segment-offset entry may be removed from the bucket. Similarly, when a grid-segment-offset entry is removed it is necessary to determine whether the O-Bucket is empty, so that the O-Bucket entry may be removed. Figure 10b shows an example where the node with O-Image position (0,0) has been removed from the O-Bucket with cost 0 shown in Figure 10a. Both the node-offset and the grid-segment-offset entries have been removed from the map. By removing empty O-Buckets and empty grid-segment-offset entries from the map as nodes are dequeued in lines 8 and 17 of the O-IFT watershed algorithm, the map structure provides a

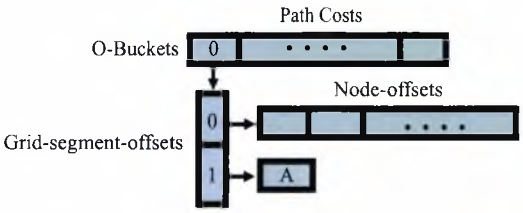


Fig. 11. Example of nodes grouped by their grid-segment-offset in O-Bucket 0. Since nodes are always dequeued from the first grid-segment-offset entry in the O-Bucket, node A is only dequeued after all nodes with grid-segment-offset 0 are.

more efficient storage and prevents any attempts to extract a node from an empty bucket in line 8 of the O-IFT algorithm, see Algorithm 2.

When dequeuing a node in line 8 of the O-IFT algorithm, the last node-offset entry of the first grid-segment-offset entry in the first O-Bucket is dequeued. This dequeuing policy favors the copy-on-write approach, since every time nodes are dequeued they are very likely to be contained in the same grid-segment. This is illustrated in Figure 11, where node A with grid-segment-offset 1 in the first O-Bucket will only be dequeued after all the nodes with grid-segment-offset 0 are. All nodes with grid-segment-offset 0 are contained in the same grid-segment. As the propagation stage proceeds, more nodes are dequeued from the same grid-segment, and with each dequeuing operation one more node in L within the grid-segment is assigned a permanent label. This behavior allows all the nodes in the grid-segment to be permanently labeled very quickly, therefore allowing the grid-segments in L and C to be shared in lines 12 and 13 of the O-IFT watershed, see Algorithm 2.

The next section studies the time-complexity of the enqueuing and dequeuing operations in the O-Buckets and then the time-complexity of the O-IFT watershed with O-Buckets is presented. The memory complexity of the O-Image is also discussed in the next section. The test and result section, presented later on, reports execution time results and memory consumption results to support this study, showing the efficiency of the O-IFT watershed with O-Buckets.

5. Complexity of the O-IFT watershed with O-Buckets

Let the sorted map of O-Buckets and each sorted map of grid-segment-offsets be implemented as a self-balancing binary search tree. Then, the time complexity of searching for an entry or creating a new entry in the map is guaranteed to be $O(\log n)$ in the worst case, where n is the number of elements in the map. Therefore, the worst-case time complexity of accessing the O-Buckets map is $O(\log Cm)$, and the worst-case time complexity of accessing a grid-segment-offset map is $O(\log Gs)$, where Gs is the number of grid-segments in the O-Image. In turn, the time complexity of erasing an entry from any of the maps is $O(1)$, given the location of the entry. Next, let each list of node-offset be implemented as a dynamic array. This list is not required to be sorted and inserting or removing an element has a worst-case time complexity of $O(Nv)$, where Nv is the number of nodes per grid-segment. For such data structures, the enqueuing and dequeuing operations in the priority queue would have a worst-case time complexity of $O(\max(\log Cm, \log Gs, Nv))$. The maximum of these values is more likely to be Nv, for a time complexity of $O(Nv)$.

Given the complexity of the queue operations, the time complexity of the loop in line 7 of the O-IFT watershed with O-Buckets is $O(|V| \times Nv)$, given that every node is processed at least once in the loop. Similarly, every arc in E is processed at least once. This results in a time complexity of $O(|E| + |V| \times Nv)$. Given that the time complexity of the IFT watershed with the linear-time queue is $O(|E| + |V| \times Cm)$, the O-IFT watershed with O-Buckets is more likely to consume more time than the IFT watershed, since the number of node values Nv within a grid-segment could be considerably larger than the maximum arc weight Cm. Furthermore, it is expected that the dynamic memory management required to maintain the O-Images L and C in the O-IFT watershed introduce some execution time overhead which is not perceived in the IFT watershed with the linear-time priority queue. However, in the last decade CPU speed has improved at a much higher rate than memory size. Therefore, in many cases it is worth trading execution time for memory space.

The memory consumption of an O-Image accounts for one pointer per grid-segment, plus the memory consumed by each linear arrangement of node values. While the number of grid-segment entries in the O-Image is fixed, the number of required linear arrangements of nodes depends on the effectiveness of the copy-on-write approach. The total number of linear arrangements of nodes is given by the number of non-shared arrangements Nsh plus the number of shared arrangements Sh. Given that in the case of medical images each node value consumes 2 bytes, the memory consumption of the O-Image can be computed using Equation 1. The memory consumed by a pointer is 4 bytes in a 32-bit system and 8 bytes in a 64-bit system.

$$Mem(\textit{O-Image}) = \begin{array}{l} (Mem(\textit{pointer}) \times Gs) + \\ (2 \times Nv \times (Nsh + Sh)) \end{array} \tag{1}$$

Figure 12 shows a comparison between the memory consumption of a traditional three-dimensional array and an O-Image for the storage of a 1024^3 image in a 32-bit system. The size of the grid-segment is 32^3. Note that if there are no shared grid-segments, the O-Image consumes more memory in comparison with the array, since the memory consumed by the pointers in the arrangement of grid-segments is not compensated by the copy-on-write approach. However, after a few grid-segments are shared the memory consumption of the O-Image is lower than that of the array, and it becomes even lower as more grid-segments are shared. For the example in Figure 12, every two grid-segments share a

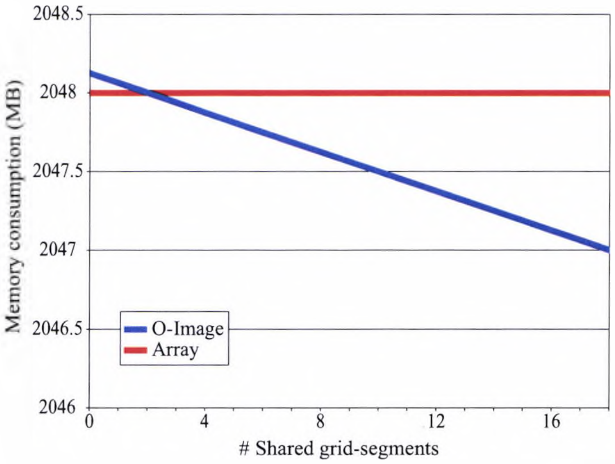


Fig. 12. Comparison of memory consumption between a traditional three-dimensional array and an O-Image in a 32-bit system, with respect to the number of shared grid-segments in the O-Image. The image has a size of 1024^3 and the grid-segment size is 32^3.

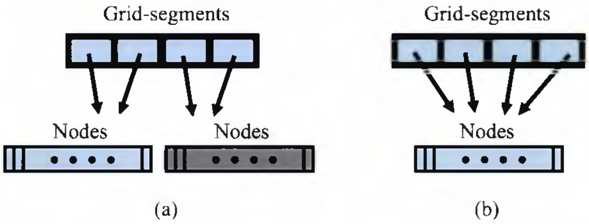


Fig. 13. Grid-segments sharing linear arrangements of node values. (a) Every two grid-segments share a different arrangement of nodes. (b) One arrangement of node values is shared among four grid-segments.

different arrangement of nodes as illustrated in Figure 13a, however note that an arrangement of node values could be shared among more than two grid-segments, as illustrated in Figure 13b. The more grid-segments are sharing a single arrangement of nodes the lower are Sh and Nsh, reducing further the memory consumption of the O-Image.

From Equation 1 it can also be inferred that if the grid-segment size is too small then the linear arrangement of grid-segments becomes too large, incrementing the number of pointers stored and increasing the memory consumption of the O-Image. The number of grid-segments in the O-Image as well as the grid-segment size also influence the memory consumption of the priority queue, since they determine the maximum size of the grid-segment-offset maps and the node-offsets dynamic arrays on each O-Bucket. For instance, a grid-segment size of 1x1x1 will result in a large memory consumption even if a considerable number of grid-segments are shared. However, this is not likely to occur if the images are large enough and contain regions with homogeneous values, since in such a case a reasonable grid-segment size can be chosen. Then, the O-Image representation would very useful for the storage of the labels and costs of the IFT watershed as a considerable amount of grid-segments could be shared. In the next section, the O-IFT watershed with O-Buckets is tested and compared with the work of Felkel et al. [15] and Falcão et al. [11], demonstrating the memory efficiency of the new IFT watershed algorithm.

6. Tests and results

In this section the O-IFT algorithm is tested, demonstrating its low memory consumption due to the better usage of memory resources of the O-Buckets and the O-Images. First, the memory consumption of the O-Buckets is studied with respect to the grid-segment size. In addition, the O-Buckets are compared with straightforward buckets to demonstrate that the O-Buckets favor the copy-on-write approach of the O-Image. After showing that the O-Buckets yield the best results, the memory-efficiency of the O-Image is tested by comparing the memory consumption of the O-IFT with O-Buckets with the memory consumption of an implementation using the linear-time priority queue proposed by Falcão et al. [11] and the memory consumption of Felkel's fifth implementation [15]. The implementation of the O-IFT watershed with O-Buckets proved to be the best of all, consuming less memory than the other implementations in all tests. The results also include the execution time of all implementations so as to compare the efficiency in execution time of the O-IFT watershed with O-Buckets with the other implementations. In the following subsections, each implementation and its associated test is described.

All implementations were applied to each of the medical datasets shown in Figure 14, on a computer with a 2.4 GHz CPU and 2 GB of RAM, running a 32-bit Microsoft Windows XP operating system. Since the arc weight function employed in the O-IFT is the dissimilarity function $w(p,q)=|f(p)-f(q)|$, the maximal arc weight Cm employed in the following tests is the maximum possible node value for CT imaging technologies, i.e., $Cm = 2^{12} - 1$. Dataset 2 is a 16-bit MR dataset but only 12 bits are actually used for each node value and therefore the same Cm as for the CT data is assumed. In all implementations, the arcs in E are defined by a 6-connectivity neighborhood around every node. Figure 14 shows the placement of the markers and the different color labels used to obtain the segmentation results shown in Figure 17.

6.1. *Testing the O-Buckets*

In this test, the O-Buckets are tested in the context of the O-IFT watershed. The test compares the memory consumption of the priority queue in two implementations of the O-IFT while varying the grid-segment size. Three different grid-segment sizes were tested: 16^3, 32^3 and 64^3. The two O-IFT implementations are identical except for the way the buckets are implemented. In the first implementation, Q uses the bucket representation shown in Figure 4. This implementation is referred to as an O-IFT with Straightforward Buckets (O-IFT-SB). Q uses O-Buckets in the sec-

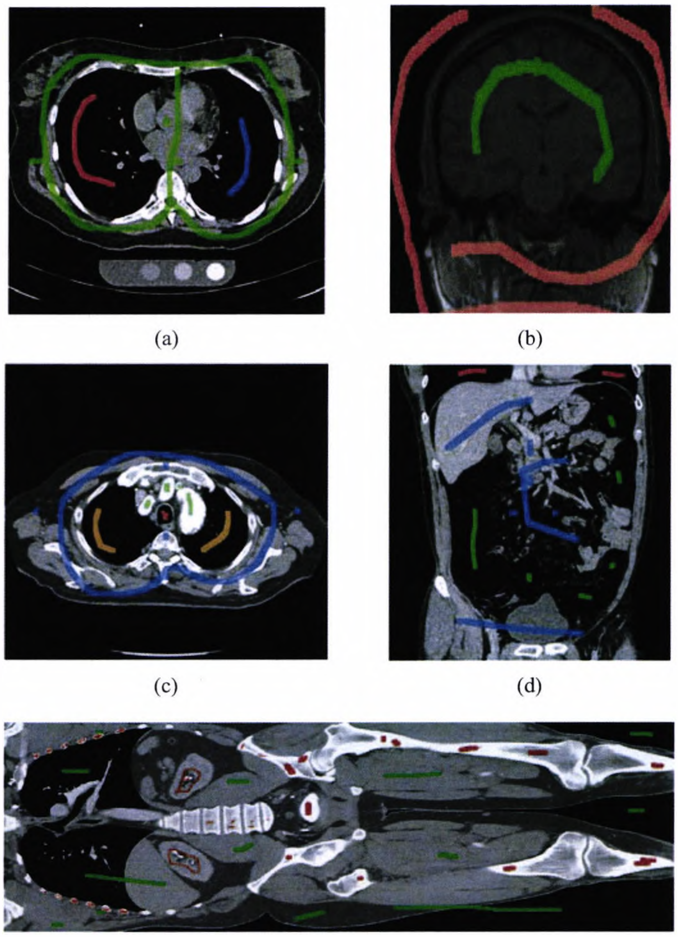

(a) (b) (c) (d) (e)

Fig. 14. Test datasets. (a) Dataset 1: 512x512x32 nodes; markers were set for the left lung (red), the right lung (blue) and the remaining data (green). (b) Dataset 2: 256x320x192 nodes; markers were set for the brain (green) and the remaining data (red). (c) Dataset 3: 512x512x96 nodes; markers were set for the lungs (orange), the aorta (green), the esophagus (red), and the remaining data (blue). (d) Dataset 4: 512x512x544 nodes; markers were set for the lungs (red), the colon (green) and the remaining data (blue). (e) Dataset 5: 512x512x1216 nodes; markers were set for the bones, the arteries and the kidneys (red), and the remaining data (green).

ond implementation, which is referred to as an O-IFT with O-Buckets (O-IFT-OB). In both algorithms, L and C are stored as O-Images. L is an O-Image where the node values are 1-byte unsigned integers. C is an O-Image where the node values are 2-byte unsigned integers. The test consisted of applying both algorithms to all test datasets, and record the maximum memory consumption of the priority queue during the watershed transformation for each different grid-segment size.

In the O-IFT-SB, a linear array with a fixed size of $Cm+1$ entries is used to store the buckets. Each bucket is implemented as a single-linked list, where each element corresponds to the position of a node. The position is stored as three 2-byte integers corresponding to the (x,y,z) coordinates of the node in the input image.

In the O-IFT-OB, a sorted map of costs is used to store the O-Buckets. The keys to this map are 2-byte unsigned integers, corresponding to the possible node values of C. For each O-Bucket, a map is also used to store the grid-segment-offsets. For each entry in the grid-segment-offset map, a dynamic array is used to store the node-offsets. The grid-segment-offsets are stored as 4-byte unsigned integers for a grid-segment size of 16^3, and as 2-byte unsigned integers for a grid-segment size of either 32^3 or 64^3. The increased memory size of the grid-segment-offset for a 16^3 grid-segment is required to handle Dataset 5, which requires more than 65 536 grid-segments. The node-offsets are stored as 4-byte unsigned integers for a grid-segment size of 64^3, and as 2-byte unsigned integers for a grid-segment size of either 16^3 or 32^3. The increased memory size of the node-offset for a 64^3 grid-segment is required to handle the large number of node values within the grid-segment. Both the dequeuing and the non-dequeuing versions of the O-IFTs were evaluated in the test.

The results are shown in Table 1 and indicate that the O-IFT-OBs performed better than the O-IFT-SBs in terms of memory consumption for all grid-segment sizes and for all datasets. The O-IFT-SB with dequeuing is not shown for the entries corresponding to Datasets 4 and 5 since the implementation takes several hours to finish. This is because searching for a node in the priority queue in line 17 of Algorithm 2 takes too long. The non-dequeuing version takes less time to terminate but at the cost of more memory. However, the O-IFT-SB without dequeuing is not shown for the entries corresponding to Dataset 5 because the queue consumes too much memory, causing the implementation to fail. The O-IFT-OB with dequeuing proved to be the best implementation in terms of memory consumption, for all grid-segment sizes.

Note in Table 1 that Q consumes less memory when the grid-segment size is 32^3 than when it is 16^3 or 64^3, for all tests cases of the O-IFT-OB. The memory consumption is higher when the grid-segment size is 16^3 because of the increased memory consumption of the grid-segment-offsets with respect to the grid-segment-offsets for a 32^3 grid-segment. In addition, the number of grid-segments required for the O-Image when the grid-segment size is 16^3 is larger than that required when the grid-segment size is 32^3. The memory consumption is higher when the grid-segment size is 64^3 because of the increased memory consumption of the node-offsets with respect to the node-offsets for a 32^3 grid-segment. In addition, the number of node-offsets required for the O-Image when the grid-segment size is 64^3 is larger than that required when the grid-segment size is 32^3. Additionally, note the different maximum sizes of Q for the different grid-segment sizes shown in Table 1. This is because the O-Bucket map entry in which a node is stored might also change when the grid-segment size changes, therefore causing the node to be dequeued from the O-Bucket at a different moment during the propagation stage. This affects the propagation of the labels and the number of nodes in the priority queue. Table 1 shows that the maximum memory consumption of Q is lower when the grid-segment size

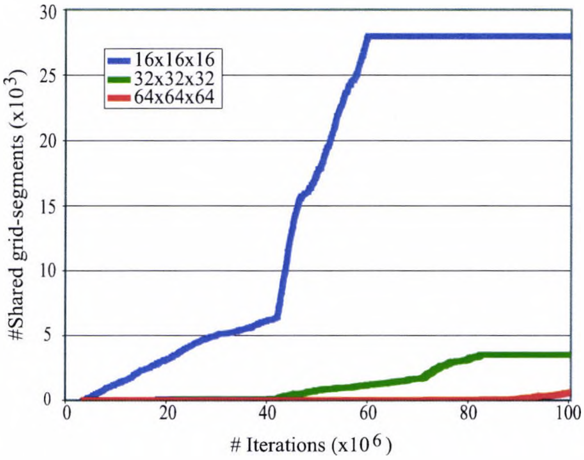


Fig. 15. Varying the grid-segment size in the O-IFT-OB. Comparison of number of shared grid-segments per iteration during the propagation stage in the O-IFT-OB for Dataset 2.

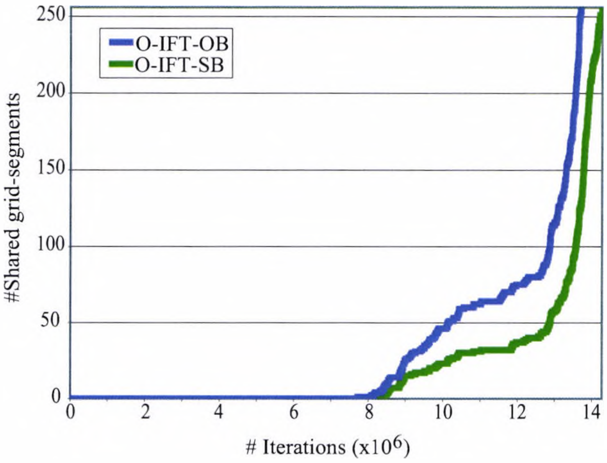


Fig. 16. Comparing the O-IFT-OB with the O-IFT-SB. Number of shared grid-segments per iteration during the propagation stage in the O-IFT-OB and the O-IFT-SB for Dataset 1.

is 32^3 than when it is 16^3 or 64^3, for all tests cases of the O-IFT-OB. The maximum number of nodes in the priority queue does not change in the O-IFT-SBs since the grid-segment size does not influence the priority queue.

In addition, as the grid-segment size increases, it takes more time for the grid-segments to be shared. Figure 15 shows a graph depicting the number of shared grid-segments in C with respect to the number of iterations during the propagation stage for the O-IFT-OB over Dataset 2, for all grid-segment sizes. Note that the number of shared grid-segments increases faster as the size of the grid-segment decreases. Furthermore, the bigger the grid-segment the more difficult it is for a grid-segment in L to be homogeneous, and therefore it is more difficult for grid-segments to be shared. In order to exploit both the copy-on-write approach of the O-Image and the memory efficiency of the O-Buckets, a grid-segment size of 32^3 was chosen for the test described in the next subsection.

This test also demonstrated the rapid grid-segment-sharing of the O-IFT-OB. Figure 16 shows a graph depicting the number of shared grid-segments in C with respect to the number of iterations during the propagation stage for the O-IFT-OB and the O-IFT-SB, when applied over Dataset 1. Note that the number of shared grid-segments increases considerably faster in the O-IFT-OB than in the O-IFT-SB, demonstrating the rapid grid-segment sharing of the O-IFT-OB. This behavior allows the O-IFT-OB to perform better during the propagation stage and maintains the memory efficiency of L and C.

These results demonstrate that the use of O-Buckets significantly reduces the memory consumption of the priority queue and favors the copy-on-write approach of the O-Image. In the next test, the O-IFT-OB is compared with Felkel's best implementation of the IFT watershed as well as with an implementation which uses Falcão's linear-time priority queue.

6.2. *Testing the O-Image*

In this section, the O-Image is assessed as a memory-efficient container for the O-IFT watershed temporary data. This test compares the memory consumption of an IFT watershed using the linear-time priority queue proposed by Falcão et al. [11] and Felkel's best implementation [15], with the O-IFT with O-Buckets presented in this paper. In particular, the memory consumption of Q, L and C is compared. For Felkel's implementation, the use of boolean flags has been avoided because they are not necessary, as suggested in [13].

The IFT watershed implementation with a linear-time priority queue is referred to as LT-IFT; it consists of an IFT algorithm with dequeuing and with Q being implemented as the linear-time priority queue illustrated in Figure 5. Each element in each bucket stores two 4-byte pointers for the next and previous elements in the list, a 2-byte integer corresponding to the cost of the node, and a 1-byte integer storing the label of the node. Therefore, the size of a queue element is 11 bytes.

Felkel's best implementation in memory consumption corresponds to their fifth implementation and it is referred to as F-V; it consists of an IFT without dequeuing, without a maximum test and with a queue element grouping mechanism. This in turn consists of storing a group of nodes in one single queue element. Basically, each bucket consists of a single-linked list of elements, where each element keeps a linear array of 256 nodes, and two local indices indicating the position of the first and the just-past-the-last nodes. Each node keeps a temporary label value and a pointer to a position in a fixed three-dimensional array, which is used to store the flags and the final label values. A 1-byte integer is used to store the final label of a node. Each node keeps a 4-byte pointer to the fixed array and 1-byte temporary label, therefore the size of a node element is 5 bytes. Each queue element keeps a 4-byte pointer to the next element, two 1-byte unsigned integers for the local indices, and a linear array of 256 nodes, therefore the size of a queue element is $(256 \times 5) + 4 + 2$ bytes $= 1286$ bytes. In this implementation, the node costs are not stored and therefore C does not consume any memory. Two fixed arrays of $Cm + 1$ pointers to the buckets are also used in this implementation. Note that Felkel's original implementation stored only a 4-byte integer in the nodes, where 1 bit was used for the label and 31 bits for the position. This has been changed in this test so that the implementation supports more than two different labels.

The O-IFT implementation used in this test is an O-IFT-OB with dequeuing. Q is implemented using the same O-Buckets described in the previous test. L and C are O-Images as described in the previous test, with a grid-segment size of 32^3.

Table 2 shows a comparison of the maximum memory consumption of each implementation, divided into the memory consumed for static data structures and the maximum memory consumed by dynamic structures. Table 2 indicates, for each algorithm, whether L, C and Q are implemented using static or dynamic structures. Implementation LT-IFT uses a static linear-time priority queue that includes the node labels and node costs; therefore no dynamic structures are used. Implementation F-V uses a dynamic priority queue that includes temporary label values, and a static three-dimensional array that stores the final label values. The node costs are not stored in implementation F-V. The O-IFT-OB stores node labels and costs using O-Images while O-Buckets are employed for the priority-queue; these are all dynamic structures. Table 2 also includes the percentage of memory saved by the F-V and O-IFT-OB implementations, with respect to the memory consumed by LT-IFT.

Table 2 also shows the execution time of each implementation, for each of the test datasets. Note that implementation LT-IFT is faster than the other implementations, due to the use of the static linear-time priority queue. Implementations F-V and O-IFT-OB are slower than LT-IFT, mostly due to the time overhead introduced by the managing of the dynamic data structures employed in both implementations. The queue operations of the O-IFT-OB take more time than in the other implementations, resulting in higher execution times. However, this is compensated by the considerable memory saving shown by the O-IFT-OB with respect to the other implementations.

Note that Table 2 also indicates that the execution time of implementation LT-IFT could not be registered for Dataset 5, the largest of the test datasets. This is due to the memory consumption of the linear-time priority queue, which exceeds 3 GB of memory. Since the implementation was tested in a 32-bit system, the static array could not be allocated in memory. After this, the allocation of a dynamic array was attempted, but this only resulted in an excessive use of the virtual memory space, which eventually caused a failure in the allocation. The same implementation was tested in a 64-bit machine with 4 GB of memory, running the 64-bit version of Microsoft Windows XP Professional. The compiler was Microsoft Visual C++ 2005. Once again, the static allocation of the array failed and a dynamic allocation was attempted. In this case the allocation was successful, but it caused the system to start paging to disk constantly, causing the implementation to take long hours to finish. Similarly, the exact final execution time of implementation F-V could not be registered either for Dataset 5. This is due to the 512x512x1216 × 2 bytes = 608 MB of memory consumed by the test dataset, which added to the memory consumption of implementation F-V exceeds the 2 GB of virtual memory space. Then, the allocation of the node groups eventually failed in a 32-bit system. However, the largest memory consumption of the implementation and its execution time was registered before it failed, and it is shown in Table 2.

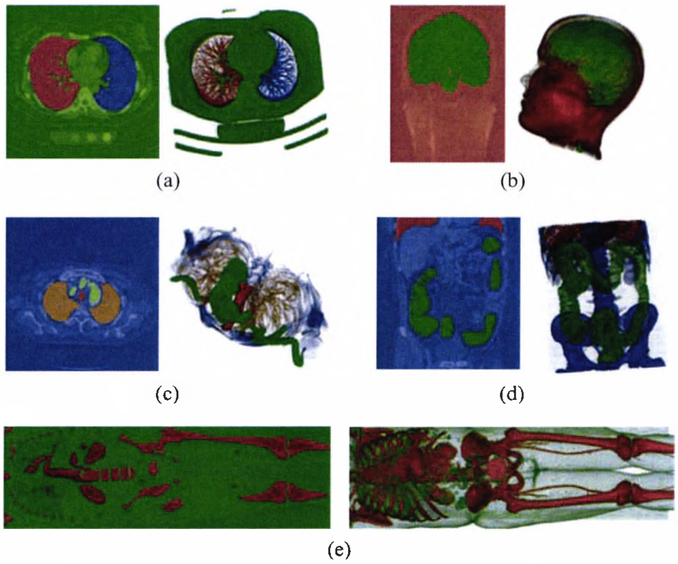


Fig. 17. Catchment basins produced by the O-IFT watershed transform performed over (a) Dataset 1, (b) Dataset 2, (c) Dataset 3, (d) Dataset 4, and (e) Dataset 5. Each illustrated result consists of a 2D image showing the catchment basins in one of the slices of the dataset, along with a semi-transparent volume rendered image of the resulting basins.

Table 2 shows that the O-IFT-OB achieves an outstanding memory saving ranging from 74 to 77% of the memory consumed by the LT-IFT implementation. These results demonstrate that the O-IFT-OB achieves a very good balance between memory consumption and execution time, while saving the largest amount of memory space from all implementations. Figure 17 shows the anatomical structures that were successfully segmented by the O-IFT-OB from the test datasets. The next section discusses the obtained results as well as some limitations of the O-IFT and other possible applications.

7. Conclusions and Future Work

A new type of IFT watershed algorithm based on the O-Buffer framework [17] was presented, as well as a new representation of the priority queue based on the same framework. Two possible implementations of the O-IFT were presented, one of them employing O-Buckets and the other employing a straightforward implementation of the buckets. The O-Buckets showed that their queue element grouping mechanism favors the copy-on-write approach [21] of the O-Image, in comparison with the straightforward buckets, consequently reducing the memory consumption of the O-Images in the O-IFT.

The O-IFT with O-Buckets was applied to five three-dimensional medical datasets, and its memory consumption was compared with the best of Felkel's implementations in terms of memory consumption, and with an implementation using Falcão's linear-time priority queue. The O-IFT with O-Buckets successfully segmented all structures and consumed less memory than the other implementations. The O-IFT with O-Buckets demonstrated a very high memory saving ranging from 74 to 77% of the memory consumed by the implementation using the linear-time priority queue, in comparison to 43 to 61% by Felkel's implementation.

Although the O-IFT watershed with O-Buckets can achieve a considerable lower memory consumption in comparison with implementations LT-IFT and F-V, its memory saving over the node costs consists of erasing the cost values of nodes that are no longer used during the propagation stage. In addition, the predecessor map is not stored. This limits the application of the O-IFT approach to other IFT-based operators [13] as well as to the Differential IFT [14], since both the cost and the predecessor of each node are required. Furthermore, the execution time of the O-IFT watershed limits its application to interactive segmentation, and it might discourage the user of iteratively trying several different choices of markers in the same image. However, since CPU speed has improved at a much higher rate than memory size in the last decade, this problem could eventually be alleviated by oncoming CPU technology.

Furthermore, note that although the largest test dataset is only 512x512x1216 in size, when the grid-segment size is 32^3 the O-Image and the O-Buckets use 2-byte unsigned integers for the offsets. Therefore, the O-IFT watershed algorithm with O-Buckets and a grid-segment size of 32^3 presented in the previous tests allows the addressing of 65 536 grid-segments, each containing 65 536 node values. Consequently, this implementation can handle datasets with a size up to 1024x1024x4096. This is larger than what Felkel's original fifth implementation can handle. Furthermore, this maximum size could be incremented further by allowing 4-byte unsigned integers to store the offsets.

The O-Buckets could also be useful in the application of other segmentation tasks based in the Dijkstra's algorithm [22], such as the computation of skeletons and centerlines based on the algorithm of Bitter et al. [23], or the traditional 3D Seeded Region Growing algorithm based on a priority queue [24]. The O-Images could be used in any application where datasets with large homogeneous regions are common. Furthermore, multi-resolution techniques can be introduced as suggested in [17], which could considerably speed up the traversal of the O-Image. In addition, this could allow the application of the IFT watershed at a level of detail higher than the node level, followed by an up-sampling of the result. This would considerably reduce the execution time of the algorithm, but it could also reduce its high resolution depending on the effectiveness of the up-sampling technique.

Acknowledgment

This work was partially funded by the Scientific and Humanistic Development Council (CDCH) of the Central University of Venezuela (UCV). The medical datasets employed during the tests described in this paper were kindly provided by Biotronics3D Limited, UK. The authors would like to thank Stephan Bruckner from the Vienna University of Technology for his support.

References

[1] S. Beucher, C. Lantujoul, Use of watersheds in contour detection, in: Proceedings of the International Workshop on Image Processing, Real-Time Edge and Motion Detection/Estimation, Rennes, France, 1979, pp. 17–21.

[2] S. Beucher, F. Meyer, The morphological approach to segmentation: The watershed transform, in: E. R. Dougherty (Ed.), Mathematical Morphology in Image Processing, Marcel Dekker Inc., New York, 1993, pp. 433–82.

[3] L. Vincent, Morphological grayscale reconstruction in image analysis: applications and efficient algorithms, IEEE Transactions on Image Processing 2 (2) (1993) 176–201.

[4] F. Meyer, S. Beucher, Morphological segmentation, Journal of Visual Communications and Image Representation 1 (1) (1990) 21–46.

[5] J. Roerdink, A. Meijster, The watershed transform: definitions, algorithms, and parallelization strategies, Fundamenta Informaticae 41 (2000) 187–228.

[6] A. Meijster, J. Roerdink, Computation of watersheds based on parallel graph algorithms, in: P. Maragos, R. Shafer, M. Butt (Eds.), Mathematical Morphology and its Applications to Image and Signal Processing, Kluwer Academic Publishers, 1996, pp. 305–12.

[7] A. N. Moga, B. Cramariuc, M. Gabbouj, Parallel watershed transformation algorithms for image segmentation, Parallel Computing 24 (14) (1998) 1981–2001.

[8] L. Vincent, P. Soille, Watersheds in digital spaces: An efficient algorithm based on immersion simulations, IEEE Transactions on Pattern Analysis and Machine Intelligence 13 (6) (1991) 583–98.

[9] F. Meyer, Topographic distance and watershed lines, Signal Processing 38 (1) (1994) 113–25.

[10] R. Lotufo, A. Falcão, The ordered queue and the optimality of the watershed approaches, in: J. Goutsias, L. Vincent, D. Bloomberg (Eds.), Mathematical Morphology and its Application to Image and Signal Processing, Vol. 18, Kluwer Academic Publishers, Boston, 2000, pp. 341–50.

[11] A. Falcão, J. Udupa, F. Miyazawa, Ultrafast user-steered image segmentation paradigm: live-wire-on-the-fly, IEEE Transactions on Medical Imaging 19 (1) (2000) 55–62.

[12] R. Lotufo, A. Falcao, F. Zampirolli, Ift-watershed from gray-scale marker, in: XV Brazilian Symposium on Computer Graphics and Image Processing, Fortaleza, Brazil, 2002, pp. 146–52.

[13] A. X. Falcão, J. Stolfi, R. Lotufo, The image foresting transform: Theory, algorithms, and applications, IEEE Transactions on Pattern Analysis and Machine Intelligence 26 (1) (2004) 19–29.

[14] A. X. Falcão, F. P. G. Bergo, Interactive volume segmentation with differential image foresting transforms, IEEE Transactions on Medical Imaging 23 (9) (2004) 1100–08.

[15] P. Felkel, M. Bruckwschwaiger, R. Wegenkittl, Implementation and complexity of the watershed-from-markers algorithm computed as a minimal cost forest, Computer Graphics Forum 20 (3) (2001) C26–C35.

[16] A. Sasov, Ultrafast micro-ct for in vivo small animal imaging and industrial applications, in: U. Bonse (Ed.), Developments in X-Ray Tomography IV, Vol. 5535 of Proceeding of SPIE, 2004, pp. 733–39.

[17] H. Qu, A. E. Kaufman, O-buffer: A framework for sample-based graphics, IEEE Transactions on Visualization and Computer Graphics 10 (4) (2004) 410–21.

[18] J. Crespo, R. W. Schafer, J. Serra, C. Gratin, F. Meyer, The flat zone approach: a general low-level region merging segmentation method, Signal Processing 62 (1) (1997) 37–60.

[19] P. Salembier, P. Brigger, J. Casas, M. Pardàs, Morphological operators for image and video compression, IEEE Transactions on Image Processing 5 (6) (1996) 881–98.

[20] R. B. Dial, Algorithm 360: shortest-path forest with topological ordering, Communications of the ACM 12 (11) (1969) 632–33.

[21] E. Abrossimov, M. Rozier, M. Shapiro, Generic virtual memory management for operating system kernels, SIGOPS Operating Systems Review 23 (5) (1989) 123–36.

[22] E. W. Dijkstra, A note on two problems in connection with graphs, Numerische Mathematik 1 (1959) 269–71.

[23] I. Bitter, A. E. Kaufman, M. Sato, Penalized-distance volumetric skeleton algorithm, IEEE Transactions on Visualization and Computer Graphics 7 (3) (2001) 195–206.

[24] R. Adams, L. Bischof, Seeded region growing, IEEE Transactions on Pattern Analysis and Machine Intelligence 16 (6) (1994) 641–47.

Dataset	Implementation	Max# Nodes(Q) 16x16x16	Max Mem(Q) 16x16x16	Max# Nodes(Q) 32x32x32	Max Mem(Q) 32x32x32	Max# Nodes(Q) 64x64x64	Max Mem(Q) 64x64x64
1	O-IFT-SB	1 388 982	13.26 MB	1 388 982	13.26 MB	1 388 982	13.26 MB
	O-IFT-SB*	3 664 508	34.96 MB	3 664 508	34.96 MB	3 664 508	34.96 MB
	O-IFT-OB	1 229 254	4.01 MB	875 605	1.75 MB	1 230 061	4.85 MB
	O-IFT-OB*	3 904 054	9.86 MB	3 562 563	6.89 MB	3 903 757	15.10 MB
2	O-IFT-SB	3 367 295	32.12 MB	3 367 295	32.12 MB	3 367 295	32.12 MB
	O-IFT-SB*	6 769 672	64.57 MB	6 769 672	64.57 MB	6 769 672	64.57 MB
	O-IFT-OB	1 523 284	3.45 MB	1 531 503	2.94 MB	1 540 438	5.90 MB
	O-IFT-OB*	5 364 590	10.82 MB	5 382 229	10.28 MB	5 341 620	20.40 MB
3	O-IFT-SB	4 902 227	46.76 MB	4 902 227	46.76 MB	4 902 227	46.76 MB
	O-IFT-SB*	8 100 265	77.26 MB	8 100 265	77.26 MB	8 100 265	77.26 MB
	O-IFT-OB	2 516 279	7.82 MB	1 504 877	2.99 MB	2 545 150	9.89 MB
	O-IFT-OB*	10 153 516	28.44 MB	7 413 510	14.28 MB	8 454 834	32.48 MB
4	O-IFT-SB*	47 631 125	454.26 MB	47 631 125	454.26 MB	47 631 125	454.26 MB
	O-IFT-OB	23 273 813	69.25 MB	8 760 705	16.82 MB	14 283 121	55.13 MB
	O-IFT-OB*	74 059 659	171.92 MB	41 972 399	80.20 MB	59 621 295	228.85 MB
5	O-IFT-OB	30 297 580	91.54 MB	30 294 982	64.71 MB	30 347 888	117.39 MB
	O-IFT-OB*	136 666 715	300.17 MB	136 857 360	268.47 MB	136 656 389	522.99 MB

Table 1
Comparison of the memory consumption of Q in the O-IFT with Standard Buckets (O-IFT-SB) and the O-IFT with O-Buckets (O-IFT-OB), using grid-segments with sizes 16^3, 32^3 and 64^3. Non-dequeuing versions are marked with a star (*).

Dataset	Implementation	Dynamic Data	Max. Mem. Dyn. Data	Static Data	Static Mem.	Total Mem.	%Mem. saved from LT-IFT	Time
1	LT-IFT	None	0 MB	L, C, Q	88.01 MB	88.01 MB	0%	6s
	F-V	L, Q	31.68 MB	L	8 MB	39.68 MB	54.91%	15s
	O-IFT-OB	L, C, Q	22.84 MB	None	0 MB	22.84 MB	74.04%	16s
2	LT-IFT	None	0 MB	L, C, Q	120.32 MB	120.32 MB	0%	10s
	F-V	L, Q	52.98 MB	L	15 MB	67.98 MB	43.5%	16s
	O-IFT-OB	L, C, Q	28.16 MB	None	0 MB	28.16 MB	76.59%	34s
3	LT-IFT	None	0 MB	L, C, Q	264.01 MB	264.01 MB	0%	27s
	F-V	L, Q	79.43 MB	L	24 MB	103.43 MB	60.82%	41s
	O-IFT-OB	L, C, Q	59.5 MB	None	0 MB	59.5 MB	77.46%	43s
4	LT-IFT	None	0 MB	L, C, Q	1.46 GB	1.46 GB	0%	119s
	F-V	L, Q	494.43 MB	L	136 MB	630.43 MB	57.83%	253s
	O-IFT-OB	L, C, Q	353.67 MB	None	0 MB	353.67 MB	76.34%	341s
5	LT-IFT	None	0 MB	L, C, Q	3.26 GB	3.26 GB	0%	∞
	F-V	L, Q	>1.20 GB	L	304 MB	>1.49 GB	<54.29%	>624s
	O-IFT-OB	L, C, Q	866.6 MB	None	0 MB	866.6 MB	74.04%	910s

Table 2
Comparison of memory consumption between the linear-time IFT (LT-IFT), Felkel's fifth implementation (F-V) and the O-IFT with O-Buckets (O-IFT-OB). An ∞ indicates that the execution time of the implementation could not be registered. The symbols $>$ and $<$ correspond to the 'greater than' and 'less than' mathematical operators respectively.

Appendix D

Feedback Questionnaires

Our system was shown to two medical doctors; Dr Sharif Sadek was a consultant radiologist from Whipps Cross hospital while Dr Nick Croft was an expert on IBD. In both cases the questionnaire was used to gauge the users opinion of the system and to determine ways in which it could be improved.

Virtual Colonoscopy System Feedback

The purpose of this questionnaire is to gain feedback relating to the Virtual Colonoscopy system and, in particular, to the Volumetric CPR component.

1) Name Dr Sherif Sadek

2) Affiliation Whipps Cross Hospital

3) What is your experience in Radiology?

General

4) What is your experience working with CT data?

CT abdomen, Colonography - prone, Supine - with contrast
Use scrolling through slices.

5) What is your experience working with CT data in 3D?

Cranial Vascular - inter cranial aneurisms

6) What is you experience with Virtual Colonoscopy or Virtual Endoscopy?

Some

7) What are you initial impressions of the Virtual Colonoscopy System?

Seems useful, not too overwelmed by no of views.

8) How would you rate the views in order of usefulness?

1) Flythrough, 2) Slices, 3) overview
when VCPR enabled this became No. 2.

9) Do you feel disoriented using the system?

no

10) Is it clear what your position and orientation is?

position - yes, orientation - sometimes

11) Do you believe the flythrough provides a useful alternative to a real colonoscopy?

yes.

12) Do you trust it?

yes, but also like to have raw data.

13) Is the Volumetric CPR component useful? In what ways?

Yes, it is useful as an overview of the colon

14) Is the surface coverage tool useful?

Yes

15) Is the Translucency window useful?

Hard to say without more experience here.

16) What other information might you like to see displayed on the Volumetric CPR?

17) Do you trust the Volumetric CPR?

Yes, but also like base data.

18) Generally, how could the system be improved?

Must see base data,
Might also like colon unfolding

19) Is the system clinically useful?

Yes, small polyps easy to see.

20) Will it speed up or improve the screening process?

Yes, faster
Focuses on correct area.

21) Does it reduce uncertainty?

Yes, esp in empty sections
User must know limitations.

Virtual Colonoscopy System Feedback

The purpose of this questionnaire is to gain feedback relating to the Virtual Colonoscopy system and, in particular, to the Volumetric CPR component.

1) Name: Dr Nick Croft

2) Affiliation: Queen Mary University of London

3) What is your experience in Radiology?

Gastrointestinal.

4) What is your experience working with CT data?

Significant on GI data.

5) What is your experience working with CT data in 3D?

Limited

6) What is you experience with Virtual Colonoscopy or Virtual Endoscopy?

Limited, but highly familiar with the optical approach.

7) What are you initial impressions of the Virtual Colonoscopy System?

Good.

8) How would you rate the views in order of usefulness?

1) Flythrough, 2) VCPR, 3) Slices.

9) Do you feel disoriented using the system?

No

10) Is it clear what your position and orientation is?

Yes

11) Do you believe the flythrough provides a useful alternative to a real colonoscopy?

Yes, provided it can go far enough in to reach the small bowel.

12) Do you trust it?

N/A for IBD

13) Is the Volumetric CPR component useful? In what ways?

Yes, it would be useful to measure wall thickness.

14) Is the surface coverage tool useful?

15) Is the Translucency window useful?

Not for IBD.

16) What other information might you like to see displayed on the Volumetric CPR?

Wall thickness.
Depth into colon

17) Do you trust the Volumetric CPR?

Yes.

18) Generally, how could the system be improved?

See (16)

19) Is the system clinically useful?

Hard to say at this point from an IBD perspective.

20) Will it speed up or improve the screening process?

21) Does it reduce uncertainty?