The Pennsylvania State University
The Graduate School
Department of Electrical Engineering

ROBUST METHODS FOR HUMAN AIRWAY-TREE SEGMENTATION
AND ANATOMICAL-TREE MATCHING

A Dissertation in
Electrical Engineering
by
Michael W. Graham

© 2008 Michael W. Graham

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

August 2008
The dissertation of Michael W. Graham was reviewed and approved* by the following:

William E. Higgins  
Distinguished Professor of Electrical Engineering  
Dissertation Advisor, Chair of Committee

David J. Miller  
Professor of Electrical Engineering

Nirmal K. Bose  
HRB-Systems Professor of Electrical Engineering

Robert T. Collins  
Associate Professor of Computer Science and Engineering

W. Kenneth Jenkins  
Professor of Electrical Engineering  
Head of the Department of Electrical Engineering

*Signatures are on file in the Graduate School.
Abstract

Robust and accurate segmentation of the human airway tree from multi-detector computed-tomography (MDCT) chest scans is vital for many pulmonary-imaging applications. Accurately segmented airways are particularly important for image-based bronchoscopic guidance. Here, the segmented airways provide a patient-specific 3D airway model, which is used to generate Virtual Bronchoscopic (VB) images simulating the view of a real bronchoscope inside the patient’s airways. The VB views are compared with the live bronchoscopic video to lead the bronchoscopist to a predetermined airway site, where a diagnostic tissue sample is taken. VB guidance has a demonstrated potential for improving the diagnostic utility of bronchoscopy, especially for peripheral sites located many generations deep into the airway tree. Existing segmentation methods are insufficient for peripheral VB guidance, however, as they frequently fail to segment small peripheral airways with weak image signatures.

This thesis proposes a novel automatic airway-segmentation algorithm, which searches the entire lung volume for airway branch signals and poses segmentation as a global graph-theoretic optimization problem. The algorithm is extensively validated on scans of both healthy and diseased airways obtained from several different sources. In particular, comparisons with ground-truth airway segmentations demonstrate that the proposed algorithm extracts substantially more peripheral airways than existing methods while producing very few false-positive branches. The automatic algorithm is combined with a suite of interactive tools for cleaning and extending critical local areas of the airway tree to form a complete computer-based segmentation system. We present results from a clinical pilot study validat-
ing the utility of the full segmentation system. During this study, airway-tree segmentations produced by the proposed system were used to help plan and guide live peripheral bronchoscopic surgery in humans. The successful outcome of the pilot study was due, in large part, to the representational accuracy of patient-specific 3D airway models derived from segmented airways produced by the proposed system.

This thesis also addresses the problem of matching pairs of anatomical trees depicted in two different high-resolution 3D images. Three basic steps are used to match the trees: (1) image segmentation, to extract the raw trees from the 3D image data; (2) axial-analysis, to define the underlying centerline structure of the trees; and (3) tree matching, to determine corresponding branches and branchpoints between the centerline structures of the trees. This thesis focuses on step (3). The matching task is complicated by several problems associated with current segmentation and axial-analysis methods, including missing branches, false branches, and other topological errors in the extracted trees. A model-based approach is proposed in which the input trees are assumed to arise from an initially unknown common tree corrupted by a sequence of topological deformations. A set of valid matches consistent with this model is constructed and the optimal match is defined to be the valid match that maximizes a similarity measure. We derive and analyze a dynamic programming (DP) algorithm that efficiently locates a globally-optimal match. The proposed algorithm is validated by comparing automatically-generated matches with ground-truth hand-generated matches. In particular, the proposed algorithm is shown to increase an important measure of matching accuracy from the 57% achieved by the best existing method to more than 96% for human airway-tree data. Finally, the proposed tree-matching methodology is shown to be useful in automatically labeling human airway-tree anatomy.
# Table of Contents

**List of Figures**                              ix  
**List of Tables**                               xiii  
**List of Abbreviations**                        xiv  
**Acknowledgments**                             xv  

**Chapter 1**  
**Introduction**                              1  

**Chapter 2**  
**Background and Significance**                14  
2.1 Bronchoscopic Lung-Cancer Assessment        14  
   2.1.1 Standard Practice                      15  
   2.1.2 Bronchoscopic Guidance                17  
   2.1.3 The Virtual Navigator System          19  
2.2 Human Airway-Tree Segmentation             20  
2.3 Tree Matching                              23  
   2.3.1 Anatomical-Tree Extraction            23  
   2.3.2 Graph-Theoretic Matching              25  
   2.3.3 Anatomical Matching                   28  
2.4 Human Airway-Tree Labeling                30  
   2.4.1 Anatomy of the Human Airway Tree      30  
   2.4.2 Automatic Tree Labeling              33  
2.5 Significance of this Thesis                36  

**Chapter 3**  
**Human Airway-Tree Segmentation**            37  
3.1 Automatic Segmentation Algorithm          37
# TABLE OF CONTENTS

3.1.1 Conservative segmentation ............................................ 40
3.1.2 Airway section filter .................................................. 40
   3.1.2.1 Isotropic resampling ......................................... 41
   3.1.2.2 Efficiently locating strong connected components .... 42
   3.1.2.3 Constructing airway sections from connected components. 44
3.1.3 Branch segment definition ........................................... 46
3.1.4 Branch segment connection .......................................... 49
3.1.5 Global graph-partitioning algorithm ............................... 52
3.2 Interactive Tool Suite .................................................. 57
3.3 Results ............................................................................ 60
   3.3.1 Automatic Algorithm ............................................... 60
   3.3.2 Segmentation for Image-Guided Peripheral Bronchoscopy 64

Chapter 4
Anatomical-Tree Matching .................................................. 70
4.1 Problem Statement and Notation ...................................... 70
4.2 Methods ........................................................................... 75
   4.2.1 Valid Match Definition ............................................. 75
   4.2.1.1 Primary Deformation Model ................................. 76
   4.2.1.2 Secondary Deformation model .............................. 80
   4.2.2 Similarity Measure Definition ................................. 85
   4.2.3 Theory .................................................................... 87
      4.2.3.1 Local Deformation Models ............................... 88
      4.2.3.2 Locally-Computable Similarity Measures ............ 96
      4.2.3.3 Recursive Definition of an Optimal Match ......... 98
   4.2.4 Dynamic Programming Algorithm ............................ 111
      4.2.4.1 Main Body of the DP Algorithm ...................... 111
      4.2.4.2 Optimal Primitive Subroutine ......................... 114
      4.2.4.3 Algorithm Analysis ........................................ 119
4.3 Results ............................................................................ 119
   4.3.1 TreeMatch Tool and Standard for Comparison ............. 120
   4.3.2 Human Airway-Tree Matching ................................. 121
   4.3.3 Other Matching Results ........................................... 130
   4.3.4 Comparison With Existing Work ............................... 132
   4.3.5 Empirical Running Time Analysis .............................. 133

Chapter 5
Human Airway-Tree Labeling .............................................. 136
5.1 Interactive Labeling Tool .................................................. 136
5.2 Automatic Labeling Algorithm ......................................... 138
   5.2.1 Major Bronchi Labeling .......................................... 140
   5.2.2 Lobar Bronchi Labeling .......................................... 144
   5.2.3 Segmental Bronchi Labeling .................................... 147
TABLE OF CONTENTS

5.3 Results ......................................................... 150
  5.3.1 Interactive labeling ................................... 150
  5.3.2 Automatic labeling ................................... 153

Chapter 6
Image-Guided Peripheral Bronchoscopy Pilot Study 156
  6.1 Methods ....................................................... 158
    6.1.1 Subjects .................................................. 158
    6.1.2 VB Guidance System .................................. 158
      6.1.2.1 Procedure Planning .................................. 158
      6.1.2.2 Live Image-guided Procedure ...................... 159
    6.1.3 Procedure Details ...................................... 161
    6.1.4 Data Analysis .......................................... 162
  6.2 Results ....................................................... 163
  6.3 Discussion .................................................. 171

Chapter 7
Summary ............................................................. 177

Appendix A
  Interpolated Surface Definition 182

Appendix B
  Graph-partitioning Optimality Proofs 185

Appendix C
  Additional Segmentation Results 189

Appendix D
  User Manual for the SegTool 193
    D.1 Automatic Algorithm ..................................... 194
    D.2 Interactive Toolkit ...................................... 200
      D.2.1 Oblique Slice and ROI Dialog Controls .......... 200
      D.2.2 Verify Automatic Segmentation Editing Mode .... 201
      D.2.3 Visualize Segmentation Mode ......................... 208
      D.2.4 Clean Segmentation Editing Mode .................... 208
      D.2.5 Livewire Editing Mode .............................. 208

Appendix E
  Proof of Theorem 4.1 .......................................... 214

Appendix F
  Isomorphism Results for Local Deformation Models 218
TABLE OF CONTENTS

Appendix G
Analysis of LocateOptimalPrimitive Subroutine 224

Appendix H
User Manual for TreeMatch Tool 226
H.1 System Initialization and Inputs 226
H.2 Simultaneous Visualization of Two Trees 228
H.3 Automatic Matching Algorithm 231
H.4 Hand Matching 232

Appendix I
User Manual for the TreeLabel Tool 240

References 247
# List of Figures

1.1 The role of airway-tree segmentation in an image-guided bronchoscopy system. 4
1.2 Difficulty of segmenting peripheral airways. 5
1.3 The tree-matching problem. 7
1.4 Workflow for peripheral bronchoscopic procedure planning. 13

3.1 Illustration of the proposed automatic segmentation algorithm’s operation. 38
3.2 Operation of the airway section filter. 42
3.3 A smooth interpolated surface connecting two airway sections. 48
3.4 A toy example illustrating the proposed global graph-partitioning algorithm. 54
3.5 A global view of the airway tree in the proposed interactive tool suite. 59
3.6 An illustration of the livewire capability of the proposed system. 59
3.7 Visual comparison of human airway-tree segmentations produced by the proposed automatic method and a previously-proposed approach. 62
3.8 The numbering scheme used to index interactively-added branches. 65
3.9 Illustration of the complete airway segmentation system. 68
3.10 Example illustrating the necessity of visual landmark extraction. 69

4.1 A hierarchical tree. 72
4.2 Example of a tree and one of its induced subtrees. 74
4.3 Toy example illustrating the basic tree-matching problem. 77
4.4 Close-up view of a typical bifurcation in the common tree. 78
4.5 Close-up view of a typical trifurcation in the common tree. 80
4.6 Example of a secondary deformation. 81
4.7 Second example of a secondary deformation. 81
4.8 A physically implausible match. 82
4.9 Extraction of a common tree under the secondary deformation model. 84
4.10 Unique primitive decomposition. 89
4.11 The main loop of the proposed DP tree-matching algorithm. 114
4.12 Reduction of LocateOptimalPrimitive subroutine to a maximum-weight bipartite matching problem. 118
## LIST OF FIGURES

4.13 Use of TreeMatch Tool to generate hand matches for evaluation of proposed automatic tree-matching algorithm.  .................................................. 122
4.14 Automatic FRC/TLC match for case h002.  .................................. 124
4.15 Automatic FRC/TLC match for case h008.  .................................. 124
4.16 Automatic FRC/TLC match for case h018.  .................................. 125
4.17 Automatic FRC/TLC match for case 20349-3-28.  ......................... 125
4.18 Automatic FRC/TLC match for case 20349-3-36.  ......................... 126
4.19 Automatic FRC/TLC match for case emphysema.  ......................... 126
4.20 Oblique MDCT cross-sections of corresponding airways as depicted in FRC and TLC scans.  .......................................................... 127
4.21 Example of a secondary deformation in case h002.  ....................... 128
4.22 Automatic match between two airway trees extracted from the same scan (case 21405-3a) using different methods.  ......................... 129
4.23 Automatic match between two airway trees extracted from the same scan (case 20349-3-29) using different methods.  ......................... 129
4.24 Automatic match between two pig airway trees.  .......................... 130
4.25 Automatic match between two mouse coronary-arterial trees.  ........ 131
4.26 Visual comparison of matches obtained by an existing method and the proposed method.  ......................................................... 134
4.27 Experimental validation for asymptotic algorithm analysis.  ........... 135

5.1 Initial tree-labeling tool view. .................................................. 138
5.2 Label assignment in the tree-labeling tool. .................................. 139
5.3 Labeled model tree used in the proposed automatic labeling algorithm. 140
5.4 Progression of the proposed automatic tree-labeling algorithm ........ 141
5.5 Major bronchi labeling examples ............................................. 143
5.6 Lobar bronchi labeling example ............................................. 145
5.7 Four common branching patterns in the right upper lobe. ............... 149
5.8 Variation not handled by proposed automatic labeling algorithm ..... 150
5.9 View of tree-labeling tool with superimposed image data .............. 153
5.10 Example of a correct automatically-labeled tree .......................... 155
5.11 Example of errors in an automatically-labeled result ................... 155

6.1 The offline procedure-planning stage of the VB guidance system. .... 160
6.2 A live screen capture of the VB guidance system during a peripheral bronchoscopic procedure .................................................. 161
6.3 Visual cues provided to the bronchoscopist at the final diagnostic site 162
6.4 Guidance results for the right lower lobe nodule in case 20349-3-24 164
6.5 Guidance results for the right lower lobe nodule in case 20349-3-24 (continued) 165
6.6 Guidance results for the right upper lobe lesion in case 20349-3-42 166
6.7 A difficult maneuver in case 20349-3-25 .................................. 172
6.8 A global airway rendering for the case with the longest overall procedure time in the study ......................................................... 173
### LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>A.1</td>
<td>Constructing a smooth interpolated surface between two airway sections</td>
<td>184</td>
</tr>
<tr>
<td>B.1</td>
<td>Illustration of Lemma B.1</td>
<td>187</td>
</tr>
<tr>
<td>C.1</td>
<td>Segmentation results used for gold-standard comparison (case 21405-3a)</td>
<td>190</td>
</tr>
<tr>
<td>C.2</td>
<td>Segmentation results used for gold-standard comparison (case 20349-3-39)</td>
<td>191</td>
</tr>
<tr>
<td>C.3</td>
<td>Segmentation results used for gold-standard comparison (case h002-tlc)</td>
<td>192</td>
</tr>
<tr>
<td>D.1</td>
<td>Windows icon for the SegTool</td>
<td>194</td>
</tr>
<tr>
<td>D.2</td>
<td>SegTool button for loading a VN case study</td>
<td>194</td>
</tr>
<tr>
<td>D.3</td>
<td>Help icon in the SegTool</td>
<td>194</td>
</tr>
<tr>
<td>D.4</td>
<td>The context-sensitive SegTool help dialog</td>
<td>195</td>
</tr>
<tr>
<td>D.5</td>
<td>Verifying the conservative segmentation in the SegTool</td>
<td>196</td>
</tr>
<tr>
<td>D.6</td>
<td>Verifying the lung volume bounding box in the SegTool</td>
<td>198</td>
</tr>
<tr>
<td>D.7</td>
<td>The automatic segmentation algorithm output in the SegTool</td>
<td>199</td>
</tr>
<tr>
<td>D.8</td>
<td>The default SegTool file format</td>
<td>200</td>
</tr>
<tr>
<td>D.9</td>
<td>Oblique slice and ROI icons in the SegTool</td>
<td>201</td>
</tr>
<tr>
<td>D.10</td>
<td>The oblique slice and HU window dialogs</td>
<td>202</td>
</tr>
<tr>
<td>D.11</td>
<td>Effects of the SegTool oblique slice controls</td>
<td>203</td>
</tr>
<tr>
<td>D.12</td>
<td>The ROI display dialog</td>
<td>204</td>
</tr>
<tr>
<td>D.13</td>
<td>Visualization options in the Verify Automatic Segmentation editing mode</td>
<td>205</td>
</tr>
<tr>
<td>D.14</td>
<td>Subtree deletion in the Verify Automatic Segmentation editing mode</td>
<td>206</td>
</tr>
<tr>
<td>D.15</td>
<td>Branch addition in the Verify Automatic Segmentation editing mode</td>
<td>207</td>
</tr>
<tr>
<td>D.16</td>
<td>An illustration of the Clean Tree editing mode</td>
<td>209</td>
</tr>
<tr>
<td>D.17</td>
<td>End of the preplanned endobronchial route for the nodule in case 20349-3-25</td>
<td>210</td>
</tr>
<tr>
<td>D.18</td>
<td>Adding a missing branch in the Livewire editing mode</td>
<td>211</td>
</tr>
<tr>
<td>D.19</td>
<td>The effect of the “Use HU weights for Livewire” option</td>
<td>213</td>
</tr>
<tr>
<td>H.1</td>
<td>File input controls for the TreeMatch Tool</td>
<td>228</td>
</tr>
<tr>
<td>H.2</td>
<td>Initial view of the TreeMatch Tool after file input</td>
<td>229</td>
</tr>
<tr>
<td>H.3</td>
<td>The TreeMatch Tool help dialog</td>
<td>230</td>
</tr>
<tr>
<td>H.4</td>
<td>Effect of camera controls in the TreeMatch Tool</td>
<td>234</td>
</tr>
<tr>
<td>H.5</td>
<td>The oblique cross-section view in the TreeMatch Tool</td>
<td>235</td>
</tr>
<tr>
<td>H.6</td>
<td>Local rigid registration using three alignment darts in TreeMatch Tool</td>
<td>236</td>
</tr>
<tr>
<td>H.7</td>
<td>The match parameters dialog</td>
<td>237</td>
</tr>
<tr>
<td>H.8</td>
<td>TreeMatch Tool view after automatic matching algorithm has been run.</td>
<td>238</td>
</tr>
<tr>
<td>H.9</td>
<td>Example of a TreeMatch Tool output file.</td>
<td>239</td>
</tr>
<tr>
<td>I.1</td>
<td>Windows icon for the TreeLabel tool</td>
<td>240</td>
</tr>
<tr>
<td>I.2</td>
<td>TreeLabel tool main window at startup</td>
<td>241</td>
</tr>
<tr>
<td>I.3</td>
<td>An unlabeled input tree</td>
<td>241</td>
</tr>
<tr>
<td>I.4</td>
<td>The TreeLabel tool help dialog</td>
<td>242</td>
</tr>
<tr>
<td>I.5</td>
<td>Labeling the trachea</td>
<td>243</td>
</tr>
<tr>
<td>I.6</td>
<td>Labeling a segmental bronchus ($RB^1$)</td>
<td>244</td>
</tr>
</tbody>
</table>
I.7 A labeled $VN$.nptith file .................................................. 245
# List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Proposed automatic segmentation algorithm running times</td>
<td>61</td>
</tr>
<tr>
<td>3.2</td>
<td>Comparison of proposed automatic algorithm to manually defined “gold standard” trees for three cases</td>
<td>64</td>
</tr>
<tr>
<td>3.3</td>
<td>Summary of segmentation results for image-guided peripheral bronchoscopy</td>
<td>66</td>
</tr>
<tr>
<td>4.1</td>
<td>Summary of matching results for six pairs of airway trees extracted from FRC and TLC scans of the same patient</td>
<td>123</td>
</tr>
<tr>
<td>4.2</td>
<td>Summary of matching results obtained using a previously-proposed tree-matching method</td>
<td>132</td>
</tr>
<tr>
<td>4.3</td>
<td>Tree-matching algorithm running times for ten cases</td>
<td>135</td>
</tr>
<tr>
<td>5.1</td>
<td>Agreement results for an initial feasibility study of the proposed tree-labeling tool</td>
<td>151</td>
</tr>
<tr>
<td>5.2</td>
<td>Accuracy results for automatic labeling of segmental bronchi</td>
<td>154</td>
</tr>
<tr>
<td>6.1</td>
<td>Summary of target diagnostic sites</td>
<td>164</td>
</tr>
<tr>
<td>6.2</td>
<td>List of target diagnostic sites</td>
<td>167</td>
</tr>
<tr>
<td>6.3</td>
<td>Comparison of peripheral sites considered in the current study versus those from a previously-reported study</td>
<td>168</td>
</tr>
<tr>
<td>6.4</td>
<td>Summary of guidance times, sample types, and sample times</td>
<td>170</td>
</tr>
<tr>
<td>I.1</td>
<td>The label numbering scheme used by the TreeLabel tool</td>
<td>246</td>
</tr>
</tbody>
</table>
### List of Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>µm</td>
<td>micrometer</td>
</tr>
<tr>
<td>3D</td>
<td>three-dimensional</td>
</tr>
<tr>
<td>BAL</td>
<td>bronchoalveolar lavage</td>
</tr>
<tr>
<td>CT</td>
<td>computed tomography</td>
</tr>
<tr>
<td>DP</td>
<td>dynamic programming</td>
</tr>
<tr>
<td>EBUS</td>
<td>endobronchial ultrasound</td>
</tr>
<tr>
<td>EM</td>
<td>electromagnetic</td>
</tr>
<tr>
<td>FRC</td>
<td>functional residual capacity</td>
</tr>
<tr>
<td>GUI</td>
<td>graphical user interface</td>
</tr>
<tr>
<td>HU</td>
<td>Hounsfield units</td>
</tr>
<tr>
<td>LLL</td>
<td>left lower lobe</td>
</tr>
<tr>
<td>LMB</td>
<td>left main bronchus</td>
</tr>
<tr>
<td>LUL</td>
<td>left upper lobe</td>
</tr>
<tr>
<td>LW</td>
<td>live wire</td>
</tr>
<tr>
<td>MDCT</td>
<td>multi-detector computed tomography</td>
</tr>
<tr>
<td>mm</td>
<td>millimeter</td>
</tr>
<tr>
<td>PA</td>
<td>pulmonary artery</td>
</tr>
<tr>
<td>PC</td>
<td>personal computer</td>
</tr>
<tr>
<td>RLL</td>
<td>right lower lobe</td>
</tr>
<tr>
<td>RMB</td>
<td>right main bronchus</td>
</tr>
<tr>
<td>RML</td>
<td>right middle lobe</td>
</tr>
<tr>
<td>ROI</td>
<td>region of interest</td>
</tr>
<tr>
<td>RTKP</td>
<td>relaxed tree knapsack problem</td>
</tr>
<tr>
<td>RUL</td>
<td>right upper lobe</td>
</tr>
<tr>
<td>TBNA</td>
<td>transbronchial needle aspiration</td>
</tr>
<tr>
<td>TKP</td>
<td>tree knapsack problem</td>
</tr>
<tr>
<td>TLC</td>
<td>total lung capacity</td>
</tr>
<tr>
<td>VB</td>
<td>virtual bronchoscopy</td>
</tr>
<tr>
<td>VN</td>
<td>Virtual Navigator</td>
</tr>
</tbody>
</table>
Acknowledgments

I am most grateful to my thesis advisor, Dr. William Higgins, for his guidance and encouragement in my research for this thesis. I am also indebted to my many colleagues and friends in the Multidimensional Image Processing Laboratory including Justin Yu, Lav Rai, Kongkuo Lu, Pinyo Taeprasartsit, Jason Gibbs, Duane Cornish, Rahul Khare, Brett Flood, and Scott Merritt. Similarly, my research would not have been possible without the help of our collaborators at Penn State’s Hershey Medical Center, including Drs. Rebecca Bascom, Viral Patel, Muhammad Khan, and Syed Gilani.

I would also like to thank my committee members for their time and for their helpful comments. In particular, I would like to thank Dr. David Miller for his years of support as an advisor, mentor, and friend.

I gratefully acknowledge financial support from the American Society for Engineering Education, the National Institutes of Health, and Penn State University.

Most importantly, I would like to thank my family, especially my wife Stacy and my parents Tom and Carol, for their constant and unconditional love and support.
Introduction

Modern medical imaging scanners can readily produce high-resolution three-dimensional (3D) images of anatomical trees, such as the airway tree and the coronary or hepatic vasculature [1–6]. Such images are of interest to researchers studying the basic biological and physical properties of the trees and to clinicians diagnosing patients and planning treatments for disease.

This thesis addresses two related problems arising in this domain:

1. **Human airway-tree segmentation** is the problem of detecting and isolating the airway branches visible in a patient’s multi-detector computed-tomography (MDCT) chest scan [1, 7–15].

2. **Anatomical-tree matching** is the problem of locating corresponding branches between two different 3D images of anatomical trees. The images may depict two different trees or the same tree at two points in time [12, 16, 17].

We first discuss the airway-tree segmentation problem, and then consider anatomical-tree
Airway segmentation is a vital step in many pulmonary-imaging applications. For example, the segmented airways serve as input for algorithms that quantify airway geometry \[15,18–20\], extract surface data for virtual bronchoscopic renderings \[21–27\], and label human anatomy \[1,17,28\]. The methods in this thesis produce segmented airways suitable for these and other applications. The thesis will focus, however, on applications involving the bronchoscopic assessment of lung cancer.

The current state-of-the-art technique for bronchoscopic lung-cancer assessment proceeds in two stages. First, the physician examines a patient’s MDCT chest scan to identify a diagnostically-suspicious region of interest (ROI). Second, a thin flexible bronchoscope is inserted into the patient and navigated through the airways to a predetermined location, from which a diagnostic sample is taken \[29–33\]. This task is challenging under any circumstances, as the physician must navigate through a complex branching structure using only the bronchoscopic video feed and a mental picture of the airway tree obtained by prior MDCT analysis. The difficulty increases dramatically, however, when the target ROI is located deep in the lung periphery, as the physician must traverse many airway generations to reach the sample site \[34,35\]. Diagnostic yields for peripheral bronchoscopic procedures using current practices are correspondingly low \[29,36\].

Image-based planning and guidance methods have shown promise towards improving peripheral bronchoscopy performance \[35–39\]. Such methods create a patient-specific 3D airway model. The model is used to generate virtual bronchoscopic (VB) renderings, which are compared to the live bronchoscopic video to guide the physician along a predefined route to the ROI. Figure 1.1 gives an example of this process, highlighting the critical role
played by the segmented airways. First, the physician indicates a target ROI in the patient’s MDCT chest scan. Next, the airway tree is segmented. The segmented airways are input to algorithms that extract the centerlines, triangulated surface data, and other information necessary to plan an appropriate route to the ROI and to provide live guidance information to the physician during the procedure [21,40].

Generating an appropriate airway-tree segmentation for VB-guided peripheral bronchoscopy is difficult. In order to provide guidance information along the complete route to a peripheral ROI, the segmentation used to construct the 3D airway model must capture even small airways located many generations deep into the tree. Such airways typically have weak image signatures and are easily missed by automated segmentation algorithms [1,7–9,15]. Figure 1.2 highlights the difficulty of segmenting peripheral airways. Manual methods and semi-automated methods requiring significant user interaction, on the other hand, are impractical for producing a full global segmentation, which may consist of hundreds of airway branches. A primary goal for this thesis is therefore the development of novel segmentation methods that capture the small airways necessary for peripheral VB guidance while operating efficiently enough to be suitable for clinical applications.

The second problem addressed by this thesis, anatomical-tree matching, also arises in several applications. As one example, two 3D images of the same tree may represent “before” and “after” snapshots with respect to a treatment [43]. In this case, matching enables direct comparison of corresponding tree branches to determine the effects of the treatment. As a second example, a 3D image of a tree of known type, such as the airway tree or coronary vasculature, may be matched to previously studied “model” trees of the same type with the goal of labeling named structures or determining anatomical variation [1,17,44–46].
Figure 1.1. An example illustrating the role of airway-tree segmentation in an image-guided bronchoscopy system for case 20349-3-24. The input to the system is a high-resolution multi-detector CT (MDCT) scan. The image data is a $512 \times 512 \times 608$ array of intensities measured in Hounsfield Units (HU) with voxel size $\Delta x = \Delta y = 0.62$ mm, $\Delta z = 0.50$ mm. (a) A transverse (constant $z$-coordinate) slice through the image data. The physician-indicated region-of-interest (ROI) is a lung nodule indicated by the white arrow. (b) The segmented airway tree, obtained by the methods described in Chapter 3. The segmentation consists of approximately 200,000 voxels. The red object in the lower-left is the target ROI [41]. (c) An external rendering of a 3D airway tree model derived in large part from the segmented airways [21, 40]. The optimal 3D route to the ROI is highlighted in blue [42]. (d) An example of live-guidance information provided to the physician during bronchoscopy [23, 24, 35]. The image on the left is a frame of the live bronchoscopic video. The image on the right is a Virtual Bronchoscopic (VB) rendering created from the 3D airway model in (c). The blue line unambiguously tells the physician which airways to follow to reach the ROI.
Figure 1.2. An example illustrating the difficulty of segmenting peripheral airways. (a) A 512 x 512 transverse (constant z-coordinate) slice through an MDCT chest scan. The red boxes present zoomed-in views highlighting the image signatures of a 1st generation airway (right) and an 8th generation airway (left). The 8th generation airway is difficult to distinguish from the surrounding image structure in a single slice. (b) An airway-tree segmentation generated by the automatic algorithm described in Chapter 3 of this thesis. The horizontal line represents the location of the transverse slice and the black arrows indicate the highlighted airways from (a). The illustrated case is 20349-3-41. Image size: 512x512x699. Voxel dimensions: $\Delta x = \Delta y = 0.77$mm, $\Delta z = 0.5$mm.
The images themselves are impractical to match, as they are defined only by a large 3D discretely sampled array of volume elements, or voxels, and their associated intensities [47, 48]. For this reason, we consider high-level models of the anatomical trees captured in the image data. The models are defined by a hierarchical graph. Edges and vertices in this graph represent the branches and branchpoints of the physical tree. Similar models have been used in a wide variety of applications; e.g., [1, 2, 5, 6, 15, 17, 49, 50].

Trees must therefore be extracted from the 3D image data before matching can proceed. Extraction typically proceeds in three steps. First, an image-segmentation algorithm produces a raw voxel-based tree. Next, an axial-analysis algorithm defines the 3D central axes of the tree branches. Finally, the graph-theoretic tree model collects the central-axis points into branches and branchpoints. The extracted tree is augmented by geometric attributes, such as branch lengths, cross-sectional areas, and branchpoint locations, measured from the image data [18, 19]. Thus, the extracted tree provides a rich, yet analytically tractable, representation of the physical tree.

Tree-extraction methods, however, are imperfect and limited by the quality of the original image data [52]. Errors introduced during imaging and extraction significantly affect tree matching [17]. Most notably, extracted trees frequently suffer from missing or broken branches, extra false branches, and ill-defined bifurcation regions [53, 54]. Figure 1.3 presents a tree-matching example highlighting the effects of image quality and extraction errors. The trees in Figure 1.3 (a) and (b) represent human airway trees depicted in two different MDCT chest scans of the same patient. The two trees therefore represent the same physical airways, yet differ significantly in appearance. Indeed, the second tree has more than twice as many branches as the first. After applying a tree-matching algorithm,
Figure 1.3. The tree-matching problem, illustrated by two airway trees extracted from different CT chest scans of the same patient (h002). Branches in the trees are represented by lines connecting branchpoints in their central-axis models. (a) A tree extracted from a functional residual capacity (FRC) scan. The tree consists of 132 branches. (b) A tree extracted from a total lung capacity (TLC) scan consisting of 342 branches. Both trees are extracted using the same segmentation [9] and axial-analysis methods [51] and are viewed from the same position in their respective image coordinate systems. (c) An illustration of a match between the branchpoints of the two trees obtained by the method in Chapter 3. Matched branches are depicted in blue for the first tree and black for the second. Unmatched branches are depicted in red and yellow. The 117 corresponding branchpoints are connected by thin green lines. For visualization purposes, the trees are rigidly aligned in a common coordinate system so as to minimize the sum of squared distances between matched branchpoints.

However, the underlying similarity between the two trees becomes apparent. Figure 1.3(c) illustrates the resulting match. The branches in red and yellow were present in only one of the two trees.
This discussion leads us to the problem statement of this thesis.

**Problem Statement** - The goal of this thesis is to develop robust methods for segmenting the human airway tree from MDCT chest scans and for matching graph-theoretic representations of anatomical trees extracted from 3D image data. To be useful for practical application, the segmentation methods must reliably reconstruct the airway tree, including the small airways required for image-based guidance of peripheral bronchoscopy, yet operate efficiently enough to fit within a standard clinical workflow. The tree-matching methods must efficiently locate good matches despite the presence of errors in the extracted trees.

The broad hypothesis that drives this proposal is:

*R robust methods for human airway-tree segmentation and anatomical-tree matching enhance the usefulness of high-resolution 3D tree images.*

Our specific goals include the development of both automatic and interactive methods for airway segmentation. The automatic methods quickly produce a full global airway-tree segmentation and capture a large fraction of the visible airways in a given MDCT scan. The interactive methods provide a means for cleaning and extending local areas of the automatically-segmented result that are critical for image-based bronchoscopic guidance. The automatic and interactive components are combined in a computer-based airway-tree segmentation system. Segmentations produced by the system are input to surface-extraction, axial-analysis, route-planning, and other image-processing algorithms, which combine to produce a data structure referred to as a case study [21, 40, 42]. The case study is used by Virtual Navigator, a computerized lung-cancer assessment and broncho-
scopic guidance system under development in our laboratory, to provide pre-bronchoscopic reports and live VB guidance information for use prior to and during a bronchoscopic procedure [21,23,24].

Our goals also include the development of a theoretical tree-matching framework. This framework facilitates the design and analysis of robust matching algorithms tailored to specific applications. We focus on matching pairs of human airway trees depicted in MDCT chest scans, but the framework also applies to different types of trees and different 3D-imaging modalities. We also apply the tree-matching framework to the problem of labeling human airway-tree anatomy.

The specific aims of this thesis can therefore be summarized.

Specific Aim 1: Develop automatic and interactive methods for human airway-tree segmentation. An important goal of this aim is to devise an efficient and reliable automatic airway-segmentation algorithm. The aim also includes the development of interactive techniques for refining an automatically-segmented tree. Together, these methods will provide a means for rapidly obtaining a full global airway segmentation and ensuring that important local areas, such as the airways visited by a bronchoscope during a live procedure, are complete and correct.

Specific Aim 2: Develop a robust airway-tree segmentation system. This aim focuses on integrating the automatic and interactive segmentation methods of Specific Aim 1 into a complete computer-based system. The system provides visualization tools for verifying automatically-segmented airways, locating small peripheral airways missed by the automatic algorithm, and interactively extracting the missed airways.

Specific Aim 3: Develop a general theory and methods for matching anatom-
ical trees. This aim involves designing algorithms capable of automatically locating corresponding branches between two input trees. The algorithms should perform robustly in the presence of common defects introduced by imperfections in the raw image data and tree-extraction process. The algorithms should have well-defined optimality criteria and acceptable worst-case running times.

Specific Aim 4: Validate method and system performance on real MDCT datasets and human subjects. Validation of the automatic segmentation algorithm is performed on MDCT scans of both healthy and diseased airways obtained from several different sources. The clinical utility of the complete segmentation system is validated in a pilot study of image-guided peripheral bronchoscopic procedures in human patients. In this study, segmentations produced by the proposed system are validated in direct comparison with the physical anatomy of the patient. Finally, validation of the proposed tree-matching methodology is performed by comparing automatically generated matches between pairs of anatomical trees to matches produced by a human observer.

We now discuss our overall plan for accomplishing these aims.

A novel automatic segmentation algorithm is the basis for the first part of Specific Aim 1. The algorithm searches the entire lung volume for airway branches and poses segmentation as a global graph-theoretic optimization problem. The algorithm’s global nature is important, as it enables the segmentation of airways with locally-weak image signatures that lead to subtrees containing obvious, strong airways. The interactive methods of Specific Aim 1 are highlighted by an extension of the “livewire” interactive-segmentation paradigm to enable the semiautomatic addition of single-voxel-thick peripheral airways with just a few mouse clicks [41,55,56].
The segmentation system of Specific Aim 2 is designed for clinical use in planning image-guided peripheral bronchoscopic procedures. The segmentation system must therefore itself operate as a component in a larger procedure-planning system. Figure 1.4 illustrates the workflow of the procedure-planning system currently used in our clinical trials [21, 57]. First, an MDCT scan is acquired and the physician indicates appropriate ROIs. Next, the automatic algorithm of Specific Aim 1 produces a full global airway-tree segmentation. Automated algorithms then extract interior airway surface data, airway centerlines, and other useful data from the segmented airways [18, 40, 57]. A planning algorithm uses this information to determine an appropriate route to each ROI. Next, the interactive segmentation methods of Specific Aim 1 are used to refine the automatically generated segmentation along each preselected route. As the automatic algorithm effectively locates a large number of peripheral airways on its own, this step typically involves adding only one or two small missed airways. Next, the surface data and centerlines are updated to reflect the extended segmentation. Finally, the MDCT scan, ROI data, surface data, and airway centerlines are collected into a Virtual Navigator case study and used to preview and guide the peripheral bronchoscopic procedure.

A general dynamic programming (DP) framework forms the basis of Specific Aim 3. A model-based approach is proposed in which the input trees are assumed to arise from an initially unknown common structure corrupted by a sequence of topological deformations. A set of valid matches consistent with this model is constructed. The optimal match is defined to be the member of this set that maximizes a user-definable similarity measure. For broad classes of deformation models and similarity measures, it is possible to efficiently locate an optimal solution.
The segmentation methods and system of Specific Aims 1 and 2 and the tree-matching methods of Specific Aim 3 must be carefully validated. The automatic segmentation methods are compared with segmentations produced by existing methods and with hand-defined “gold standard” segmentations of real MDCT chest scans. These comparisons demonstrate that the proposed automatic algorithm captures significantly more peripheral airways than previously-proposed approaches. The clinical utility of both the interactive segmentation methods and the integrated segmentation system is validated in a live peripheral bronchoscopy pilot study. During this study, segmentations generated using the system of Specific Aim 2 are used to plan and guide live peripheral bronchoscopy in human patients. Finally, the tree-matching methods of Specific Aim 3 are validated by comparing automatically generated matches between pairs of human airway trees, pig airway trees, and mouse coronary arterial trees to hand-generated ground-truth matches. In particular, the proposed methods are shown to significantly outperform the best existing algorithm for matching pairs of human airway trees.

The remainder of the thesis is organized as follows. Chapter 2 provides background for the airway-segmentation and tree-matching problems, surveys related work, and describes the significance of this thesis. Chapter 3 describes the proposed automatic and interactive airway-tree segmentation methods and details the operation of the integrated segmentation system. Chapter 4 introduces the proposed DP tree-matching framework and develops an efficient and practical tree-matching algorithm. Chapter 5 describes an application of the proposed tree-matching methodology to the problem of labeling human airway-tree anatomy. Chapter 6 presents results from a clinical pilot study demonstrating the efficacy of the proposed segmentation methodology and the entire Virtual Navigator system for
Figure 1.4. Workflow for peripheral bronchoscopic procedure planning. The highlighted components are contributions of this thesis.

planning and guiding peripheral bronchoscopy in humans. Finally, Chapter 7 summarizes the contributions for this thesis and describes possible avenues for future work.
Chapter 2

Background and Significance

The goals of this thesis include human airway-tree segmentation for application in image-based peripheral bronchoscopic guidance, anatomical-tree matching, and human airway-tree labeling. This chapter will review literature related to these goals and discuss the overall significance of the thesis. The chapter is organized in five parts. Section 2.1 describes literature related to the bronchoscopic assessment and treatment of lung cancer, with an emphasis on both bronchoscopy to the lung periphery and methods for bronchoscopic guidance. Section 2.2 provides background for the airway-tree segmentation problem and discusses prior airway-segmentation work. Section 2.3 discusses work related to anatomical-tree matching. Section 2.4 provides background for the problem of automatically labeling human airway-tree anatomy. Finally, Section 2.5 reviews the significance of this thesis.

2.1 Bronchoscopic Lung-Cancer Assessment

Bronchoscopy is a minimally-invasive method for obtaining diagnostic samples of suspect lung tissue [33]. Successful bronchoscopy enables the early diagnosis and assessment of
lung cancer while avoiding the increased risk of complications associated with more invasive procedures such as mediastinoscopy, thoracoscopy, or transthoracic needle aspiration [58,59].

This section discusses work related to bronchoscopic lung-cancer assessment in three parts. First, Section 2.1.1 describes the current standard for bronchoscopic assessment, which requires significant expertise on the part of the physician. Next, Section 2.1.2 discusses bronchoscopic guidance systems, which aim to simplify procedures and improve diagnostic yields. Finally, Section 2.1.3 provides an overview of the Virtual Navigator system, an image-based bronchoscopic guidance system under development in our laboratory.

2.1.1 Standard Practice

In the current standard practice for bronchoscopic lung-cancer assessment, the physician first scans a patient’s MDCT chest scan for suspect regions-of-interest (ROIs) then performs follow-up bronchoscopy to obtain diagnostic samples of the ROIs [32]. Target ROIs can be large or swollen lymph nodes, solitary lung nodules, diffuse masses, inflamed airways, or other diagnostically-suspicious tissue [30,31,33]. The bronchoscope itself is a thin, flexible tube containing optical fibers that transmit video images from inside a patient’s airways. Modern bronchoscopes typically have a hollow working channel through which tools or probes can be inserted into the airways and sample tissue can be extracted [33].

The bronchoscopic assessment of a target ROI involves two related navigation problems. First, the physician must navigate through the airway tree to an appropriate biopsy site. Second, the physician must sample the ROI. Sampling may entail a simple visual inspection of airway-wall tissue. More commonly, however, cytologic or histopathologic samples are obtained by use of brushes, transbronchial aspiration needles, biopsy forceps, bronchoalveo-
olar lavage (BAL) catheters, or other tools inserted through the bronchoscope’s working channel [33].

Under standard practice, the physician must navigate to the biopsy site and determine how to best sample the target ROI using only a mentally-reconstructed picture of the patient’s anatomy obtained from the MDCT data. This task requires significant experience and expertise, as evidenced by research indicating significant variation in diagnostic yield between physicians and a significant learning curve for new physicians [60,61].

Procedures become even more difficult when the target ROI is located in the lung periphery [29]. In such cases, the bronchoscope must be correctly navigated through several airway generations before even reaching the biopsy site. It has been recently demonstrated that even experienced bronchoscopists make navigation errors by the fourth airway generation when relying only on the MDCT data as per the standard practice [34,35].

Despite the practical difficulties associated with the standard bronchoscopic practice, there remains considerable interest in peripheral bronchoscopy. The introduction of ultra-thin bronchoscopes, which can pass deep into the airway tree, and sub-millimeter-resolution MDCT scanners, which can detect small lung nodules and accurately depict peripheral airways, have helped to spur this interest [62]. Furthermore, investigations into the use of MDCT chest scans as a screening mechanism for early lung-cancer detection in high-risk patients highlight the need for safe, accurate, and cost-effective methods for diagnosing suspect peripheral tissue [63].
2.1.2 Bronchoscopic Guidance

Several methods exist to aid the physician in successfully navigating to and sampling a target ROI. Intraoperative fluoroscopy, for example, provides the physician with real-time x-ray images of patient anatomy in which the bronchoscope is clearly visible [33]. Fluoroscopic views, however, provide only a 2D projection of the anatomy and do not suffice to recover the 3D position of the bronchoscope. Furthermore, airways and small lung lesions are frequently obscured by dense structures such as the rib cage, and extensive use of fluoroscopy during a procedure results in increased radiation exposure for both patient and physician.

More recently, CT-fluoroscopes capable of producing low-resolution 3D images have been introduced to bronchoscopy [64, 65]. CT-fluoroscopy can be used to determine the approximate 3D position of the bronchoscope during a procedure. Interpretation of 3D fluoroscopic images, however, involves the same mental reconstruction problem posed by the interpretation of the original MDCT chest scan. Also, small lesions and peripheral airways can be difficult to see in the low-resolution images produced by CT-fluoroscopes. Finally, CT-fluoroscopy subjects both patient and physician to at least five times the radiation dosage of conventional fluoroscopy [65]. Thus, prolonged use of CT-fluoroscopy for navigation through many airway generations to a peripheral biopsy site is to be avoided if other, safer techniques are available [64].

An alternate approach to bronchoscopic guidance tracks an electromagnetic (EM) probe that is inserted through bronchoscope’s working channel [66–73]. The 3D location of the probe with respect to an EM field generator located under the patient can be accurately determined in real-time during the procedure. Determination of the probe’s location within
the patient’s airways requires alignment of the coordinate systems of the EM-field generator and the pre-operative MDCT chest scan. This is typically accomplished by touching the probe to several anatomical landmarks visible in the MDCT scan prior to beginning guidance [66,70]. Of course, the patient’s anatomy is non-rigid and the global coordinates of both the target ROI and the airways within the EM field change constantly with the respiratory cycle. Thus, the probe’s location within the airways can only be approximately known [74,75].

Endobronchial ultrasound (EBUS) probes have also demonstrated the potential to aid physicians in performing successful bronchoscopy [76–79]. EBUS probes provide local ultrasound images of extraluminal tissue when placed up against an airway wall. The ultrasound images can help locate lymph nodes and other lung lesions that would otherwise be hidden from view. EBUS probes have therefore been particularly successful in improving the diagnostic yields of procedures involving transbronchial needle aspiration (TBNA), in which the physician must pierce an extraluminal ROI using a thin, hollow needle [77]. As EBUS probes provide only local ultrasound images, they are not suitable for globally localizing the bronchoscope within the airway tree [73]. Furthermore, current generation EBUS probes are too large to fit within the small working channel of ultrathin bronchoscopes. Therefore, although EBUS is a promising technology for improving diagnostic yield once an appropriate biopsy site has been reached, it is not sufficient for peripheral bronchoscopic applications.

Recently, image-based guidance has shown promise towards improving bronchoscopic performance without the use of probes or other significant additional hardware [24,36,39, 62,78,80–90]. Image-based guidance techniques are rooted in the fields of computer graphics and computer vision, and guide the physician using Virtual Bronchoscopic (VB) renderings.
VB renderings are derived solely from a patient’s pre-operative MDCT chest scan and can simulate the view seen by a physical bronchoscope located in an arbitrary location and orientation within the patient’s airways \[91–93\]. Image-based systems compare VB renderings with the live bronchoscopic video to lead the physician down a pre-planned route through the airways to an appropriate biopsy site. Once the biopsy site has been reached, the VB views can be enhanced with quantitative information or fused with renderings of extraluminal objects such as the lung vasculature or the target ROI. Image-based systems have been shown to provide effective guidance for both lymph-node biopsy in the central chest \[24,81,82,87\] and for peripheral nodule biopsy \[36,39,62,78,88,89\].

2.1.3 The Virtual Navigator System

*Virtual Navigator* (VN) is an image-based system for planning and guiding bronchoscopic procedures under development in our laboratory. Like other image-based navigation systems, VN guides the physician using informative endoluminal VB renderings. The system, however, contains several novel components that move beyond the simple presentation of VB views. First, the pre-operative planning system used by VN is almost completely automatic \[21,57\]. This is in contrast to other image-based guidance methods, which, while useful and successful, place a significant pre-operative burden on the physician \[36,39,88,89\]. Also, the system contains a novel web-based reporting mechanism, which provides the physician with both a one-page preview of the key VB views to be encountered during the procedure and an interactive VB movie of the pre-planned route \[94,95\]. Finally, the system uses state-of-the-art computer-vision techniques that enable real-time registration and fusion between the “real world” of the live bronchoscopic video feed and the “virtual world”
provided by the MDCT-based VB views [96,97].

The automated procedure-planning, report generation, real-time registration, and other novel components of the VN system make it ideal for use in bronchoscopic guidance. Indeed, the system’s performance has already been carefully validated for central chest lymph-node biopsy in human and animal studies and for accurate peripheral ROI localization in a rigid plastic phantom [24, 35]. Until recently, however, the effectiveness of the system had not been demonstrated for peripheral guidance in humans.

A major barrier in the path towards clinical use of VN for peripheral guidance had been the system’s inability to generate accurate VB renderings in the periphery. This had been due, in large part, to inadequate methods for airway-tree segmentation. Segmentations produced by previous methods were sufficient for image-based guidance through the large airways of the central chest, but failed to capture the small airways necessary for guidance to the periphery. Segmentations produced by the methods proposed in this thesis are effective for peripheral guidance. These improved segmentations, along with numerous other recent improvements to the system, have enabled a pilot study demonstrating the safety and effectiveness of VN for peripheral guidance in humans. Details of the pilot study are presented in Chapter 6.

2.2 Human Airway-Tree Segmentation

An MDCT chest scan is represented by a large 3D array of voxels and associated intensities [47]. Voxel intensities are measured in Hounsfield units (HU) and scanners are calibrated such that air has intensity around -1000 HU, water around 0 HU, and blood and soft
tissue around 50-200 HU [48]. Thus, airways nominally appear in CT scans as tubes of low-intensity airway lumen surrounded by high-intensity walls. Each voxel, however, spans a non-trivial volume measured by the voxel dimensions $\Delta x$, $\Delta y$, and $\Delta z$. Voxels on the lumen/wall boundary therefore frequently have intermediate intensity between that of air and soft tissue. This effect is particularly pronounced for peripheral airways with thin walls. Thus, peripheral airway lumen voxels often have intensities significantly higher than -1000 HU and peripheral wall voxels frequently have intensity far less than 0 HU [9, 98].

Additional complications arise during image reconstruction, which involves a choice of convolution kernels. Soft kernels, such as the Siemens B31f kernel, have a smoothing effect and tend to blur small airways [99]. Sharp kernels, such as the Siemens B50f and B70f kernels, highlight image gradients but amplify high-frequency noise [9, 100]. Motion artifacts, non-standard patient anatomy, and airway obstructions introduce additional challenges.

Many existing airway segmentation methods use region-growing algorithms, which attempt to separate air and soft-tissue voxels using a global HU threshold [1, 9, 11, 14]. The final segmented result is a set of air voxels connected to a seed point. Region growing is fast and assumes no prior knowledge of the shape or size of the airways. Choosing an appropriate global HU threshold is difficult, however, as the lungs are filled with air and misclassifying a single wall voxel can allow the segmentation to leak into the lung parenchyma. Filtering the image prior to initializing region growing can mitigate the leakage problem, but filtering removes small peripheral airways [9].

Several methods based upon mathematical morphology have also been proposed [7–9, 12]. Morphological methods pass a set of filters over the image to locate candidate airway locations. A reconstruction step then attempts to reject false candidates to produce the
final segmented result. Morphological approaches are appealing because they scan the entire lung volume. Thus, unlike region-growing algorithms, they can “see” strong airway signals not directly connected to a seed voxel. Morphological filters are frequently slow, however, and developing an appropriate reconstruction algorithm has proven to be difficult.

Some recent methods incorporate locally-adaptive segmentation algorithms and seek to detect and stem leakages. Several algorithms propagate a wavefront through the airway tree. The front splits into smaller, child fronts at bifurcations. Leakage is detected when the front splits too rapidly [10,13]. Another approach characterize leakages as having a “spongy” texture [15]. When leakage is detected, locally-adaptive algorithms typically switch to a more conservative set of parameters and re-run the segmentation. Such algorithms can be myopic, as the decision to stop segmenting at a particular branch or bifurcation is made without benefit of global information.

None of the existing methods discussed in this section address the problem of segmentation for image-guided peripheral bronchoscopy. Specifically, no existing method is capable of both segmenting a full global airway tree and reliably extracting all of the small airways visible to the bronchoscope along a route to a peripheral ROI.

We propose an automatic algorithm that introduces several novel airway-segmentation techniques and captures positive qualities from many of these previously proposed approaches. The algorithm uses seeded region growing to quickly and reliably extract the major airways. Leakage is avoided by aggressively smoothing the image. The algorithm runs an efficient nonlinear filter over the entire lung volume in search of peripheral airway signatures. The filter output guides and constrains a locally-adaptive region-growing algorithm, which segments and connects entire airway branches. Finally, a novel graph-
partitioning algorithm makes an optimal global decision on which branches to retain in the segmentation. In Section 3.3 this automatic algorithm will be shown to quickly and reliably produce a full global segmentation from MDCT chest scans of both healthy and diseased airways obtained using a variety of scanners and image-reconstruction algorithms.

The proposed automatic algorithm is paired with an interactive segmentation toolkit to ensure accurate reconstruction of the small peripheral airways that are critical for image-based guidance. Together, the automatic algorithm and interactive toolkit form a unique and robust segmentation system for image-guided peripheral bronchoscopy.

2.3 Tree Matching

Anatomical-tree matching is the problem of locating corresponding branches and branch-points in graph-theoretic tree models extracted from two different 3D images. This section reviews literature related to the tree-matching problem in three parts. First, Section 2.3.1 surveys automatic tree-extraction methods, as tree extraction is a necessary and important precursor to tree matching. Next, Section 2.3.2 reviews the graph-theoretic tree-matching literature as general-purpose graph-theoretic matching algorithms frequently serve as a basis for anatomical-matching methods. Finally, Section 2.3.3 discusses algorithms specifically designed to match anatomical trees.

2.3.1 Anatomical-Tree Extraction

Before tree matching or labeling can proceed, the trees must be extracted from the image data. The extraction methods used to define input trees can significantly affect the performance of the subsequent matching algorithm. Hence, we review tree-extraction methods to
understand the limitations and imperfections this stage adds to the matching problem.

Extraction typically proceeds in three steps. First, an image-segmentation algorithm defines the raw tree \([2, 5, 9, 15, 20, 49, 53, 98, 101–104]\). Next, an axial-analysis algorithm extracts the tree’s 3D central axis structure \([2,5,6,15,20,40,51,105–107]\). Finally, the central-axis points are converted into a data structure representing the underlying tree’s branching pattern \([2,15,20,50,51,108]\). Many other tree-extraction methods are possible \([52]\). Front-based and branch-following algorithms, for example, simultaneously segment and extract central axes \([6,109,110]\).

Image segmentation is difficult and error-prone. As noted in Section 2.2, automatic airway-tree segmentation algorithms frequently miss visible branches or “leak” outside the visible airway walls \([1, 9, 15, 111]\). Automatic methods for segmenting vascular trees suffer from similar problems \([6,53,102]\). Many vessel-segmentation algorithms operate by thresholding a likelihood image. The threshold represents a trade-off between detecting small branches and separating nearby branches \([2,52,54]\). Finally, finite image resolution, the point-spread function of the imaging system, and image-reconstruction artifacts limit the efficacy of image segmentation \([5,54,112]\).

Axial-analysis algorithms can also introduce errors. Many methods rely on 3D thinning or skeletonization \([113–116]\). The skeletonization process can produce false branches \([5,6, 15,49,51,53,117,118]\). Furthermore, the 3D skeleton of a segmented tree is only defined to voxel-level precision. Thus, post processing is necessary to produce smooth and centered axes \([51]\). Some recent works have proposed axial-analysis methods based on the geometry of surfaces \([40,54,105]\). Such methods enable more precise central-axis definition. Yu et al. take particular care to correctly locate branchpoints \([40]\).
The 3D central axes produced by axial-analysis methods represent a considerable simplification of the image data. Further simplification is achieved by considering the branches and branchpoints of the central axes as the edges and vertices of a graph. The graph representation is an integral part of numerous tree-analysis systems [2, 15, 19, 117, 119, 120]. Despite the best efforts of researchers, however, no automatic method can guarantee complete and accurate tree extraction. For this reason, Yu et al. developed a computer-based system for semi-automatically refining both the graph-theoretic topology and the central axes of extracted trees [53].

Several researchers have proposed methods for augmenting an extracted tree with detailed quantitative measurements [2,12,19,20,49,108,110,119]. Such measurements capture geometric properties of the underlying anatomical tree, such as branch lengths, branching angles, and cross-sectional areas. For the matching and labeling algorithms developed in this thesis, we will represent quantitative measurements as vector-valued attributes associated with the edges and vertices of our input trees.

The tree-extraction methods surveyed in this section have been carefully developed and validated by numerous researchers over many years, and are capable of extracting detailed models of anatomical trees depicted in 3D medical images. The methods, however, are automatic and therefore introduce errors into the extracted tree models. The matching methodology developed in this thesis is specifically designed to tolerate such errors.

2.3.2 Graph-Theoretic Matching

The result of the tree-extraction process is a graph-theoretic tree. The vertices and edges of this tree represent branchpoints and branches of the physical tree. As anatomical trees
usually have a well-defined hierarchy and emanate from a single source, the extracted trees are also hierarchical and have a designated source, or root, vertex. Many methods for matching anatomical trees are therefore based on graph-theoretic algorithms for matching rooted trees. This section surveys the graph-theoretic tree-matching literature.

We characterize tree-matching methods by three features: (1) the set of valid matches, or search space, considered by the method; (2) the cost function, similarity measure, or other criteria used to evaluate matches; and (3) the existence of an efficient algorithm to locate a locally or globally optimal solution [121–123]. A robust tree-matching method must tolerate missing branches, false branches, and other topological defects introduced during tree extraction. The match evaluation criteria must consider the quantitative measurements associated with the input trees. The algorithm should efficiently locate a good match.

The simplest tree-matching algorithms assume the two input trees exhibit identical branching patterns. An isomorphism is a match between two trees that exactly preserves the hierarchical (parent/child) relationships of both trees [123, 124]. Isomorphisms, however, exist only between trees having the same number of branches and vertices. The subtree isomorphism framework removes this restriction by allowing matches between identical subtrees of the input trees [123,125]. Typically, subtree isomorphism algorithms search for the match that identifies the largest identical subtrees. Torsello et al. extended this framework by introducing a maximum-similarity subtree isomorphism, which evaluates valid matches by summing a similarity measure between matched vertices [126]. In both cases, algorithms exist for efficiently locating a globally optimal match.

The subtree isomorphism approach does not consider the possibility of missing or false branches in the extracted trees [17]. Such topological errors are handled within the tree-edit
distance framework [127–131]. The optimal tree-edit match describes a minimum-cost sequence of elementary edit operations transforming one tree into the other [128,129]. Thus, edit distance algorithms can tolerate most errors introduced during tree extraction. The elementary edit operations are vertex deletion, insertion, and substitution. Early work assigned a constant cost to each elementary edit operation [128]. Researchers later introduced cost functions incorporating categorical vertex attributes [122,129]. More complex costs incorporating real-valued vertex and edge attributes have also been proposed [121,130,131].

For ordered trees, in which the children of each vertex have a fixed sequence, the optimal edit-distance mapping can be located efficiently [127–131]. For the unordered trees we consider in this proposal, however, computing the edit distance for even the simplest cost function is intractable [132].

The subtree isomorphism and edit-distance frameworks represent two extremes of complexity for tree-matching algorithms. Several researchers have proposed approaches of intermediate complexity, such as tree alignment, tree inclusion, and constrained tree-edit distance [133–137]. Such algorithms tolerate topological errors that are excluded by the subtree isomorphism framework, but are less permissive than the general edit-distance framework. None of these algorithms, however, adequately address the full range of errors arising during tree extraction. In the tree-inclusion framework, for example, false branches are tolerated in only one of the two input trees [123,137]. The constrained edit-distance mapping allows matches involving false and missing branches in both trees, but rejects matches which reverse the branching order of two close bifurcations [135]. Such errors are known to occur during tree extraction [17].
2.3.3 Anatomical Matching

Several authors have proposed matching algorithms tailored to anatomical trees [12,17,111, 138,139]. Pisupati et al. proposed the earliest method, in which they assumed one input tree could be obtained from the other by pruning branches [12]. The method employed a dynamic programming algorithm to check for the existence of valid matches, but had no well-defined criteria for comparing matches.

Tschirren et al. extended the subtree isomorphism framework to incorporate geometrical attributes and tolerate significant topological errors in both trees. The method defines the optimal match as the largest clique in an association graph [125]. As the problem of locating a maximum clique is NP-complete, the method employs heuristic strategies to keep running time within reasonable limits. The method first prunes small branches from the input trees to reduce the overall problem size. Next, the method finds and matches the “major” branchpoints of the two trees. Finally, the method matches the subtrees located beneath corresponding major branchpoints. This strategy can only guarantee a locally optimal solution, but has produced reasonable results for pairs of human airway trees extracted from MDCT chest scans.

Charnoz et al. propose a tree-matching algorithm for intra-patient hepatic vasculature registration [16,138]. The correspondences found by the algorithm are used to estimate non-rigid liver deformations and track the evolution of liver tumors in CT scans. The algorithm allows for the possibility that edges may be missing from either of the two trees, but does not consider more complex topological errors. Matching proceeds in a top-down fashion, beginning with the roots of the two trees. At each step, hypothesized matches
are generated according to local criteria, and the most promising matches are retained for further exploration. Results are presented for a virtual liver consisting of 380 branches. The virtual liver is deformed using a biomechanical model designed for hepatic surgery simulation and random branches are pruned from the trees. The algorithm produced good results on most synthetic cases and also performed well on two much smaller trees obtained from a real patient.

Several authors have proposed point-based approaches motivated by the topological errors that complicate graph-theoretic algorithms [111, 139]. Point-based methods discard the graph-theoretic topology of the tree and directly match its central-axis points. Kaftan et al. match paths from the tree root to the terminal point of a terminating branch. A cost function compares distances and angles between central-axis points [139]. This path-matching procedure, however, does not generally produce a one-to-one matching. An iterative post-processing step is necessary to exclude many-to-one matches. Bulow et al. match central-axis points with similar 3D shape contexts [111, 140]. The reasonable results produced by this method demonstrate that the 3D shape context captures much relevant information. The authors leave the integration of 3D shape context features with the graph-theoretic topology of the tree for future work.

The existing methods detailed in this section are specifically designed to match anatomical trees. By incorporating domain-specific knowledge, such methods improve upon the general purpose matching algorithms discussed previously [17, 139]. The point-based methods, however, discard much useful information by ignoring the error-prone, but largely accurate, graph-theoretic topology of the input trees [111, 139]. The methods designed to exploit the graph-theoretic nature of the input trees either fail to account for common
tree-extraction errors [16, 138], or tolerate such errors by sacrificing optimality [17].

We propose an algorithm that tolerates common extraction errors and efficiently locates an optimal match. Our algorithm can be easily extended to tolerate more severe errors and incorporate a variety of rich geometrical cost functions. Our work therefore represents what is to our knowledge the first general treatment of the tree-matching problem.

2.4 Human Airway-Tree Labeling

A goal of this thesis is to apply the proposed tree-matching methodology to the problem of labeling human airway trees. It is therefore necessary to review both the relevant anatomical literature and prior work on tree labeling. Section 2.4.1 discusses the anatomy of the human airway tree with a focus on branching pattern variations that complicate automatic tree labeling. Section 2.4.2 reviews existing automatic methods for labeling both the human airway tree and other types of anatomical trees.

2.4.1 Anatomy of the Human Airway Tree

We adopt the nomenclature of Netter [141], which is based on Jackson and Huber [142] and Boyden [143]. We also consult a number of works that catalog significant variations in airway-tree structure [144–149].

The main function of the lungs is to oxygenate the blood [148]. This function is carried out by the bronchoalveolar system, which is comprised of the airway tree and the alveolar region extending from the respiratory bronchioles to the alveoli [144]. In this system, the airway tree serves as a pathway through which air is moved into and out of the lungs.

The root branch of the airway tree is referred to as the trachea. The trachea extends
downward from the larynx and divides into the main, or first-order, bronchi. The right main bronchus (RMB) enters the right lung. It is larger, but somewhat shorter, than the left main bronchus (LMB), which supplies the left lung. The main bronchi divide into lobar (second-order) bronchi, each of which enters a lobe of a lung. The right lung consists of an upper lobe (RUL), a middle lobe (RML), and a lower lobe (RLL). The left lung is divided into upper (LUL) and lower (LLL) lobes. Visible horizontal or oblique fissures separate the lobes [148]. Lobar bronchi further divide into segmental (third-order) bronchi, each of which supplies a bronchopulmonary segment. Bronchopulmonary segments are not, in general, separated by visible fissures [147].

Ten bronchopulmonary segments are typically found in the right lung. The upper lobe is divided into apical (RB\textsuperscript{1}), posterior (RB\textsuperscript{2}), and anterior (RB\textsuperscript{3}) segments. Frequently, the three segmental bronchi branch out independently, although a variety of branching patterns can occur [144, 149]. The middle lobe consists of lateral (RB\textsuperscript{4}) and medial (RB\textsuperscript{5}) segments. The lower lobe contains the superior (RB\textsuperscript{6}), medial basal (RB\textsuperscript{7}), anterior basal (RB\textsuperscript{8}), lateral basal (RB\textsuperscript{9}), and posterior basal (RB\textsuperscript{10}) segments. In addition, a subsuperior basal segment (RB\textsuperscript{*}) is present in approximately 44-60\% of the population [144, 146]. In 3-22\% of the population, the superior segment is separated from the rest of the lower lobe by an aberrant horizontal fissure. In cases where this aberrant fissure is present, RB7 tends to share a common trunk with RB8 or RB\textsuperscript{*} [147]. It has been further suggested that RB7 may be missing entirely in as many as 17\% of the population [145].

The left lung most commonly consists of eight bronchopulmonary segments. In the upper lobe, the apicoposterior (LB\textsuperscript{1+2}) and anterior (LB\textsuperscript{3}) segments are connected to the upper lobe bronchus through the superior division bronchus. The superior (LB\textsuperscript{4}) and inferior (LB\textsuperscript{5})
segments, on the other hand, are connected through the lingular bronchus. The lower lobe is comprised of the superior (LB\textsuperscript{6}), anteromedial basal (LB\textsuperscript{7+8}), lateral basal (LB\textsuperscript{9}), and posterior basal (LB\textsuperscript{10}) segments. Again, a subapical or subsuperior basal (LB\textsuperscript{*}) bronchus is frequently present [144]. An independent medial basal segment (LB\textsuperscript{7}) is observed in a small number (4\%) of the population [145], although it is typically absent due to the presence of the heart [146].

More substantial variations in airway tree anatomy are known to occur in a small percentage of the population. For instance, one study observed a right tracheal (pre-eparterial bronchus) in approximately 2\% of the population [149]. The authors also observed unexpected segmental bronchi incident upon the right main and right intermediate bronchi. An extra “accessory” lobe, the azygos lobe, appears in the right lung in approximately 1\% of the population [148].

It is also possible to assign anatomical names to smaller, subsegmental bronchi. Netter indicates fourth-order bronchi by lower-case Roman letters (e.g., RB\textsuperscript{2a}), fifth-order bronchi by Roman numerals (e.g., RB\textsuperscript{2ai}), and sixth-order bronchi by lower-case Greek letters (e.g., RB\textsuperscript{2ai\beta}). Oho and Amemiya present a more detailed description of the naming conventions for subsegmental bronchi [150].

Basic textbook presentations of human airway tree anatomy typically consist of a single model tree representing the “dominant” branching pattern [141,148]. The many branching pattern variations cataloged in this section, however, demonstrate the insufficiency of single-tree model for describing even the segmental anatomy of the airways. Thus, robust labeling algorithms must tolerate both the extraction errors that complicate tree matching and the anatomical variations observed in human anatomy.
2.4.2 Automatic Tree Labeling

Automatic tree-labeling algorithms typically match an input tree of known type, such as the airway tree or coronary arterial tree, to an anatomical model representing the population of similar trees. The model then provides labels for the matched branches of the input tree.

Early research into automatic labeling methods focused on coronary arterial trees extracted from X-ray angiograms [44–46,151–154]. Angiograms provide only two-dimensional projections of the coronary arterial tree. Thus, the extracted trees frequently suffer from overlapping vessels and anatomically implausible loops. Such extraction errors complicate the labeling problem. The scale of the problem, however, is relatively small with only six to ten labels assigned to each tree.

Garreau et al. propose a knowledge-based approach that is applied only to phantom data [152]. Dumay et al. describe the first graph-theoretic approach [154]. In this approach, the extracted tree is matched to a model tree consisting of eleven labeled branches. The matching is performed using an approximate edit-distance algorithm [153]. Ezquerra et al. match the extracted tree to a model tree using a branch-and-bound algorithm [45]. This process starts by labeling the root of the tree and greedily chooses the best labels from a pool of candidates at each bifurcation point. Harris et al. match by locating the maximum clique in a weighted association graph [46]. The matching algorithm has exponential complexity in the number of assigned labels, but can be solved in reasonable time due to the small scale of the problem.

The problem of labeling the human airway tree is more involved. A description of the segmental anatomy of the airway tree requires more than 30 anatomical names [141,148].
Osborne et al. published the first detailed study of the segmental anatomy as depicted in CT chest scans [155]. The authors manually located approximately 70% of the segmental bronchi in 50 scans of healthy patients with 10mm thick slices. The images used in this study would be considered poor by current standards, as we frequently encounter CT chest scans with 0.5mm thick slices. Failures in locating individual bronchopulmonary segments were attributed to motion artifacts and the poor resolution of the input data. Jardin and Remy studied 113 scans with 9mm thick slices. Deviations from the typical branching pattern were catalogued and found to be in good agreement with classical anatomy texts [145]. Neither study considered trees extracted from CT images. Both, however, confirmed that even early, low-resolution CT scans provided enough detail to recognize named anatomical structures.

Mori et al. proposed the first fully-automatic algorithm for labeling the airway tree [1]. The algorithm used a knowledge base to label trees extracted from scans with 2 to 5mm-thick slices. The trees themselves were incomplete and the knowledge base was unable to tolerate anatomical variations. The encouraging results presented for these incomplete trees, however, demonstrated the feasibility of the automatic labeling problem.

A more recent work by Mori et al. proposed a method which divides the airway tree into four parts and constructs multiple models for each part [28]. The report considers relatively sparse trees extracted from CT scans with 1.0 - 2.5 mm slice spacing. The method proceeds in a top-down fashion, selecting the best candidate model at each branchpoint and discarding all models that do not exactly match the topology of the input tree. There is no mechanism for tolerating false or missing branches. The authors validate the method on 25 hand-labeled trees using the leave-one-out method and report that 90% of the automatically-
assigned labels agreed with the hand-labeled result.

Kitaoka et al. label an input tree by matching it to a single hand-constructed model tree [156]. The matching algorithm employs as weighted association graph based on the method of Pelillo et al. [125]. The method cannot, however, tolerate false branches and the input tree must be manually pruned before labeling. Tschirren et al. also match the input tree to a single model tree [17]. Here, the model represents a population average constructed from a set of hand-labeled example trees. The optimal match is defined as the maximum weight clique in an association graph, but the exponential complexity of the maximum-clique problem necessitates an approximate solution. The method was validated on 17 trees extracted from scans with 0.6mm-thick slices using the leave-one-out approach. Most of the assigned segment labels were correct, but almost a quarter of the hand-assigned labels did not appear in the automatically generated labelings.

Of the existing methods for labeling the human airway tree, only two significantly address branching pattern variations. Tschirren et al. implicitly considered variations by incorporating population averages into their model [17]. The matching algorithm used by this method tolerated further variations, but produced only a locally optimal solution. Mori et al. proposed using multiple models to explicitly capture branching pattern variations [28]. The method employs a greedy matching algorithm, however, and immediately discards models whose branching patterns do not exactly match the input tree.

In Chapter 5 we describe an automatic labeling algorithm that matches an unlabeled input tree to a single labeled model tree using the proposed DP matching framework. Even with a simple single-tree model, the algorithm will be shown to tolerate common anatomical variations. We also propose a straightforward extension of this algorithm that
would incorporate a multiple-tree anatomical model.

2.5 Significance of this Thesis

The current standard practice for bronchoscopic lung-cancer assessment frequently results in unacceptably-low diagnostic yields, especially for small lesions in the lung periphery. Image-based planning and guidance systems have demonstrated a potential to reduce the effects of skill variation between physicians and improve overall bronchoscopic yields. Successful image-based guidance of peripheral bronchoscopy, however, requires accurate and reliable segmentation of small peripheral airways with weak MDCT-image signatures. This thesis describes the first clinically-validated airway-segmentation system for image-guided peripheral bronchoscopy.

This thesis also proposes state-of-the-art automatic methods for anatomical-tree matching. The problem of matching anatomical trees arises frequently in the study of high-resolution 3D medical image data. Robust tree matching enables meaningful comparative studies by identifying corresponding branches between two different tree images. Accurate tree matching is also vital for automatic tree labeling, which enables broader studies of corresponding branches across a whole population of similar trees and provides a direct link between the rich anatomical knowledge of physicians and the detailed, but raw, information contained in a 3D image.
Human Airway-Tree Segmentation

This chapter describes methods used for segmenting the human airway tree from MDCT chest scans. Section 3.1 describes automatic methods for quickly and reliably extracting a large fraction of the visible airways in a given scan. Section 3.2 describes a collection of interactive methods used to refine and extend critical local areas of an automatically segmented result. The automatic and interactive methods combine into a complete system for robust human airway-tree segmentation. Finally, Section 3.3 presents experimental results validating both the automatic algorithm and the complete system.

3.1 Automatic Segmentation Algorithm

The proposed global segmentation algorithm proceeds in five steps illustrated in Fig. 3.1:

1. **Conservative segmentation**—A conservative segmentation consisting of the major airways is extracted via 3D region growing. To prevent leakage, the algorithm considers only heavily smoothed image data (Section 3.1.1, Fig. 3.1a).
Figure 3.1. Illustration of the proposed automatic segmentation algorithm’s operation for case 20349-3-28. The top images provide a global view of the algorithm’s progress. The bottom images provide a focused local view of a small bifurcation in the right lower lobe. The slices in the local view are oblique cross sections with isotropic pixel dimensions of 0.5mm windowed to HU ∈ [−1000, −200].

(a) **Conservative segmentation**—The conservative segmentation contains 172,650 voxels. (b) **Airway section filter**—The filter produces 50,000 candidate airway sections. Each section is drawn as a light-green ellipse approximating the cross section and running direction of an airway. The filter ignores large branches that should be captured by the conservative segmentation. (c) **Branch segment definition**—Airway sections are connected into potential branch segments. Each of the 1,157 potential branch segments located by the algorithm is represented by a tubular wireframe surface. Branch segments touching the conservative segmentation are colored green. The rest are yellow. (d) **Branch segment connection**—Interpolated surfaces connecting neighboring branch segments are colored orange. Only the 624 branch segments whose connections eventually reach the conservative segmentation are drawn. (e) **Global graph partitioning algorithm**—The algorithm retains 515 branch segments. The segments in the local view are all retained. (f) The corresponding binary segmentation contains 317,271 voxels.
2. **Airway section filter**—An efficient nonlinear filter scans the image for short airway sections. Each section provides a local airway cross-section and running-direction estimate (Section 3.1.2, Fig. 3.1b).

3. **Branch segment definition**—Sequences of airway sections sharing a common wall are connected into potential branch segments. Each branch segment is represented by a tube-like surface approximating an interior airway wall and a score measuring the magnitude of the image gradient near its surface (Section 3.1.3, Fig. 3.1c).

4. **Branch segment connection**—Neighboring branch segments are connected by smooth interpolated surfaces. The connections are optimized to minimize a cost derived from the HU value of voxels enclosed by the connection surface. At this point, the segmentation problem is represented by a graph in which weighted vertices represent potential branch segments and their scores, weighted edges represent branch connections and their costs, and a single root vertex represents the conservative segmentation (Section 3.1.4, bottom of Fig. 3.1d).

5. **Global graph-partitioning algorithm**—A global optimization algorithm partitions the branch segment connection graph to separate true-positive from false-positive branch segments. The final binary segmentation returned by the algorithm includes the conservative segmentation and all branch segments retained in the optimal graph partition. (Section 3.1.5, Fig. 3.1e).

The remaining subsections of Section 3.1 describe each of the above five steps in detail.
3.1.1 Conservative segmentation

The proposed algorithm begins by extracting the major airways using the adaptive 3D region-growing approach of Mori et al. initialized with an interactively selected root voxel in the proximal trachea. [1] To prevent leakage, the image is prefiltered by clamping the maximum voxel intensity to -500 HU and convolving with a 3D isotropic Gaussian kernel of standard deviation $\sigma \approx 0.55mm$. Note that this smoothed image is used only for generating the conservative segmentation. The remainder of the algorithm uses the raw (unfiltered) image data.

The choice of region growing for the conservative segmentation is made of convenience. It would be possible to substitute any 3D segmentation method, such as those in the references [7, 8, 15]. Region growing, however, is easy to implement and quickly produces acceptable results for most cases. On the rare occasion that the segmentation escapes the airways, the user can interactively select an appropriate termination threshold.

3.1.2 Airway section filter

As the major airways are extracted in the conservative segmentation, the remainder of the method focuses on locating and extracting small peripheral airways. This section describes a nonlinear filter that searches the lung volume for short sections of peripheral airways. Each airway section is represented by a normal vector describing the airway’s local running direction and an ellipse approximating its cross section in the associated normal plane. The airway sections will later constrain and guide a locally-adaptive 3D segmentation algorithm.

The filter proceeds in three steps described in Sections 3.1.2.1-3.1.2.3. First, the image is resampled to obtain an isotropic volume. Next, the filter considers the 2D binary images
obtained by thresholding each transverse, coronal, and sagittal slice of the isotropic volume at each possible intensity. Each four-connected component of a binary threshold image receives a score measuring the likelihood that the component represents the intersection of an orthogonal viewing plane and a branch. Finally, high-scoring connected components from neighboring slices combine to form airway sections.

Fig. 3.2 illustrates the basic idea of the filter. The yellow ellipses in A,B, and C approximate connected components that lie in contiguous coronal slices. The coronal slices, however, are not normal to the local branch running direction and the elliptical approximations are therefore more eccentric than the true branch cross section, which is nearly circular. The ellipse centers in A,B, and C do, however, lie along the medial axis of the airway and a 3D regression line fit to these points yields a good approximation for the local running direction of the branch. The filter output D is obtained by projecting A,B, and C into the plane normal to the regression line and averaging the three resulting ellipses.

3.1.2.1 Isotropic resampling

The airway section filter is most conveniently described and implemented for isotropic images. The input image is therefore resampled to have isotropic voxel dimensions \( \Delta x = \Delta y = \Delta z = 0.5 \text{mm} \). Resampled voxel intensities are obtained via trilinear interpolation. To minimize the number of voxels processed by the filter, the resampled image is limited to the bounding box of an approximate lung-volume segmentation. The lungs are segmented by subsampling the input image by a factor of eight in each dimension and applying a region-growing algorithm with a threshold of -600 HU seeded near the distal end of each leaf branch of the conservative segmentation. The intensities of conservative segmentation
Figure 3.2. Operation of the airway section filter for a small peripheral airway in case 20349-3-29 (Image dimensions: 512 × 512 × 678; voxel dimensions: \( \Delta x = \Delta y = 0.52 \text{mm}, \Delta z = 0.5 \text{mm} \)). All slices are windowed to HU \( \in [-1000, 200] \). (a) Idealized view of filter operation. On the right, the three connected components lie in orthogonal viewing planes not normal to the branch running direction. The dotted line is a line of best fit for the component centers and provides a good estimate of the local airway running direction. (b) Filter operation for real data. Components A, B, and C lie in consecutive coronal slices. Component D lies in an oblique cross-section normal to the estimated local running direction. Its elliptical representation is taken to be the average of the three ellipses from the coronal slice components projected onto the oblique cross-section.

voxels are temporarily set to 0 HU to prevent the lung segmentation from escaping through the trachea.

3.1.2.2 Efficiently locating strong connected components

The filter considers all connected components defined at each integer threshold between -1000 HU and -600 HU in each transverse, coronal, and sagittal slice of the isotropically resampled image. Each component is assigned a score measuring the likelihood that it belongs to a peripheral airway. For concreteness, consider the transverse (constant \( z \)-coordinate) slice at index \( k \). Let \((i, j)\) represent a pixel in the slice with corresponding intensity \( I(i, j) \). Let \( C \) represent a four-connected component of sub-threshold pixels and \( \partial C \) represent the component’s border pixels, here taken to be the set of pixels \((i, j) \notin C\) with a four-neighbor
in $C$. The score of $C$ is
\[ S(C) = \min_{(i,j) \in \partial C} \{ I(i,j) \} - \frac{1}{|C|} \sum_{(i,j) \in C} I(i,j). \tag{3.1} \]

Equation (3.1) strongly favors airway components, which consist of dark pixels surrounded on all sides by bright wall pixels. No score threshold, however, reliably separates airway components from non-airway components in all scans. It is therefore important to retain many components and postpone the vital airway/no-airway decision until more information is available. In practice, we retain the strongest 750,000 components.

Each surviving component is compactly represented by the following statistics.

\[ |C|, \sum I(i,j), \sum i, \sum j, \sum i^2, \sum j^2, \sum ij, \text{ and } \min_{(i,j) \in \partial C} \{ I(i,j) \} \tag{3.2} \]

Each sum in (3.2) is over all $(i, j) \in C$. Both the component’s score and an elliptical approximation to the component’s boundary can be constructed from these statistics. To construct the elliptical approximation, first calculate the covariance matrix

\[ \text{Cov}(C) = \frac{1}{|C|} \left( \begin{array}{cc} \sum i^2 & \sum ij \\ \sum ij & \sum j^2 \end{array} \right) - \frac{1}{|C|^2} \left( \begin{array}{cc} (\sum i)^2 & (\sum i)(\sum j) \\ (\sum i)(\sum j) & (\sum j)^2 \end{array} \right) + \frac{1}{12} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}. \tag{3.3} \]

The $\frac{1}{12}I$ term arises from treating each pair $(i, j)$ as a continuous random vector uniformly distributed on the unit square centered at $(i, j)$. In practice, this correction is vital for accurately representing small components. The boundary of the component is approximated
by the 3D ellipse

\[ x(\theta) = P_z^T M \theta + c, \quad \theta \in [0, 2\pi), \text{ where} \]

\[ P_z = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}, \quad \theta = \begin{pmatrix} \cos \theta \\ \sin \theta \end{pmatrix}, \quad c = \begin{pmatrix} \frac{1}{|C|} \sum_i \lambda_1 \sum_j \lambda_2 \\ k \end{pmatrix}, \text{ and} \]

\[ M = 2 \cdot \begin{pmatrix} e_1 & e_2 \end{pmatrix} \begin{pmatrix} \sqrt{\lambda_1} & 0 \\ 0 & \sqrt{\lambda_2} \end{pmatrix}. \]

Here, \( \lambda_1 \) and \( \lambda_2 \) are the eigenvalues of \( \text{Cov}(C) \) with associated eigenvectors \( e_1 \) and \( e_2 \). The ellipse swept out by \( x(\theta) \) is parallel to the \( x\)-\( y \) plane and has center \( c \). Elliptical approximations for components in coronal and sagittal slices are defined analogously.

To ensure reasonable running times, it is necessary to efficiently search the components in a slice. This is accomplished by processing the pixels in a slice in order of increasing intensity. As the intensity values are integers in a small range, the pixels are ordered in linear time using counting sort [157]. The set of components is maintained using the disjoint-set forest data structure with the path-compression and union-by-rank heuristics [158].

### 3.1.2.3 Constructing airway sections from connected components

The filter attempts to construct an airway section from each of the 750,000 components retained by the method of Section 3.1.2.2. Consider a component \( C^{(k)} \) belonging to the \( k^{th} \) transverse slice. Let the pair \((M^{(k)}, c^{(k)})\) describe the elliptical approximation for \( C^{(k)} \) per (3.4). If \( C^{(k)} \) belongs to an airway, it should partially overlap other components from the same airway in transverse slices \( k - 1 \) and \( k + 1 \) when viewed along the \( z \)-axis. Let
45

$C^{(k-1)}$ and $C^{(k+1)}$ be one such pair with elliptical approximations $(M^{(k-1)}, c^{(k-1)})$ and $(M^{(k+1)}, c^{(k+1)})$.

If all three components belong to the same airway, the centers $c^{(k-1)}$, $c^{(k)}$, and $c^{(k+1)}$ all lie near the airway’s medial axis, and a good estimate for the local running direction can be obtained by fitting a line to the three points. A good fit is provided by the line passing through the point

$$q = \frac{1}{3} \left( c^{(k-1)} + c^{(k)} + c^{(k+1)} \right)$$

with running direction

$$n = \arg \max_p \left\{ \sum_{l=k-1}^{k+1} \left[ p^T (c^{(l)} - q) \right]^2 \right\} \text{ such that } p^T p = 1 .$$

In general, there may be multiple overlapping components in slices $k - 1$ and $k + 1$, and therefore many possible choices for $C^{(k-1)}$ and $C^{(k+1)}$, with each pair producing a different running direction estimate. To choose the best pair, compute an airway section score given by

$$\text{median} \{ S(C^{(k-1)}), S(C^{(k)}), S(C^{(k+1)}) \} \cdot G(C^{(k-1)}, C^{(k)}, C^{(k+1)}),$$

where $S$ is the score of a component per (3.1) and $G$ measures the degree to which the three components agree on the local airway geometry. To measure agreement, project the components’ elliptical approximations into a common viewing plane normal to $n$ with origin at $q$. Let $r$ and $u$ be orthonormal vectors chosen such that $r$, $u$, and $n$ form a right-handed coordinate frame. Projecting $C^{(k)}$, for example, into this frame yields the 2D ellipse

$$x_{2D}(\theta) = \mathcal{P}(n) \left[ \left( P^T M^{(k)} \theta + c^{(k)} \right) - q \right] = M_p^{(k)} \theta + c_p^{(k)},$$

where

$$(3.9)$$
As peripheral airways are gently curved with slowly varying cross sections, all three components should project to roughly the same ellipse. A reasonable choice for $G$ is therefore the area of the intersection of the three projected ellipses divided by the area of their union.

The final airway section representation for $C^{(k)}$ is constructed from the pair of components maximizing (3.8). The representation is given by the 3D ellipse

$$ x(\theta) = P(n)^T \bar{M} \theta + \bar{c}, \quad (3.11) $$

which lies in the plane orthogonal to the estimated airway running direction $n$. Here, $\bar{c} = q + P(n)^T c_p$, and the pair $(\bar{M}, c_p)$ defines a 2D average ellipse of $(M_p^{(k-1)}, c_p^{(k-1)})$, $(M_p^{(k)}, c_p^{(k)})$, and $(M_p^{(k+1)}, c_p^{(k+1)})$. Again, it is important to retain as many airway sections as possible to avoid discarding sections belonging to small peripheral airways. Our current implementation retains the 50,000 highest-scoring sections.

### 3.1.3 Branch segment definition

Each airway section represents only a small fraction of a branch. This section describes a method for connecting airway sections sharing a common wall into branch segments. As it is not a priori known which airway sections belong to the same branch, the method separates the airway sections into disjoint groups of likely branch segments using a graph algorithm. Let $\mathcal{G}$ be a graph in which each vertex represents an airway section and each edge connects two vertices that may represent neighboring sections on the same branch. Each
airway section is described by the quadruple \( E_i = (n_i, P_i(n_i), \mathbf{c}_i, M_i) \), per (3.11). If \( E_i \) and \( E_j \) are neighboring sections, they must be physically close. They must also be represented by roughly parallel ellipses. Furthermore, the line connecting the centers of \( E_i \) and \( E_j \) should be roughly parallel to the estimated running direction of both \( E_i \) and \( E_j \). The following requirements are therefore necessary in order for \( E_i \) and \( E_j \) to be connected by an edge.

\[
\|\mathbf{c}_i - \mathbf{c}_j\| \leq 3\text{mm}, \quad |n_i^T n_j| \geq \cos(60^\circ),
\]
\[
\frac{|n_i^T (\mathbf{c}_i - \mathbf{c}_j)|}{\|\mathbf{c}_i - \mathbf{c}_j\|} \geq \cos(60^\circ), \quad \text{and} \quad \frac{|n_j^T (\mathbf{c}_i - \mathbf{c}_j)|}{\|\mathbf{c}_i - \mathbf{c}_j\|} \geq \cos(60^\circ)
\] (3.12)

These simple geometric tests eliminate the vast majority of potential edges and ensure that \( \mathcal{G} \) is sparse.

A more important requirement is that \( E_i \) and \( E_j \) be joined by a six-connected set of airway lumen voxels that can be segmented without leakage. This requirement is enforced by constructing a smooth tubular surface connecting \( E_i \) and \( E_j \) and attempting to locally segment the 3D sub-image of voxels lying between the planes normal to \( n_i \) and \( n_j \). The interpolated surface enables easy detection of segmentation leakage by approximating the interior airway wall. Appendix A provides the method for constructing the interpolated surface and Fig. 3.3 gives an illustrative example. An adaptive region-growing algorithm seeded at \( \mathbf{c}_i \) performs the local segmentation. Voxels are added to the local segmentation as the region-growing threshold is gradually increased from an initial value of -1000 HU. The segmentation is successful, and the edge connecting \( E_i \) and \( E_j \) is added to \( \mathcal{G} \), if \( \mathbf{c}_j \) is added before leakage occurs.
Each edge \((E_i, E_j)\) of \(G\) is augmented with the following additional information. Let \(\tau(E_i, E_j)\) be the interval of region-growing thresholds such that \(c_i\) and \(c_j\) are connected but leakage has not yet occurred. For each threshold in \(\tau(E_i, E_j)\), assign a score to the resulting segmentation equal to the difference between the minimum intensity of a six-connected border voxel and the mean segmented voxel intensity. Let \(S(E_i, E_j)\) be the maximum score over all thresholds and define \(\sigma(E_i, E_j)\) to be the corresponding set of segmented voxels.

Define a branch segment as a sequence of airway sections \(E^{(1)}, E^{(2)}, \ldots, E^{(N)}\), with \(2 \leq N \leq 20\), such that:

1. The sections \(E^{(i)}\) and \(E^{(i+1)}\) are connected by an edge in \(G\) for all \(i < N\).

2. The intersection \(\bigcap_{i=1}^{N-1} \tau(E^{(i)}, E^{(i+1)})\) is non-empty; i.e., there exists a common threshold at which the entire branch segment can be segmented without leakage.

3. The section centers \(\bar{c}^{(1)}, \bar{c}^{(2)}, \ldots, \bar{c}^{(N)}\) connect \(\bar{c}^{(1)}\) to \(\bar{c}^{(N)}\) without backtracking. Formally, \((\bar{c}^{(N)} - \bar{c}^{(1)})^T (\bar{c}^{(i)} - \bar{c}^{(1)}) < (\bar{c}^{(N)} - \bar{c}^{(1)})^T (\bar{c}^{(i+1)} - \bar{c}^{(1)})\) for all \(i < N\).

Each valid branch segment receives a score equal to the sum of its constituent edge scores. Concatenating the smooth interpolated surfaces associated with each edge produces a piecewise-smooth surface for the entire branch segment, and the union of the optimal local
segments for each edge yields a connected segmentation of the entire branch segment without leakage.

An airway section can belong to many potential branch segments. The airway sections are partitioned into a collection of high-scoring and pairwise-disjoint branch segments in two steps. First, a dynamic programming algorithm locates the highest-scoring branch segment connecting each pair of airway sections. Note that most pairs are not connected by any valid branch segment, as segment length is limited and $G$ is sparse. Next, the optimal branch segments are sorted in order of decreasing score and all segments sharing an airway section with a higher-scoring segment are discarded. The 1,500 strongest remaining branch segments are retained.

3.1.4 Branch segment connection

At this point, the segmentation consists of a connected set of conservative segmentation voxels and a collection of potential branch segments enclosed by tubular surfaces. While some branch segments contact the conservative segmentation, most are isolated “islands” of lumen voxels. This section describes a method for linking isolated branch segments to the conservative segmentation.

The output of the section is naturally represented by a hierarchical tree $T_{\text{conn}}$, whose root vertex represents the conservative segmentation. The remaining vertices represent branch segments. The edge connecting a non-root vertex in $T_{\text{conn}}$ to its parent represents a connected set of segmented voxels joining a branch segment to either the conservative segmentation (if the parent vertex is the root) or to another branch segment. Each edge is augmented with a nonnegative weight measuring the cost of adding its connection voxels
to the segmentation. Connections containing high-intensity voxels receive high cost.

Each edge in $T_{\text{conn}}$ represents one of three distinct connection types. The first is the trivial connection for a branch segment that already shares lumen voxels with the conservative segmentation. Examples of such branch segments are colored bright green in Fig. 3.1(d). Such edges require no additional voxels and receive zero cost.

The second connection type joins two branch segments by constructing an interpolated surface between an end-cap of the child branch and a cross-section of the parent branch. The orange interpolated surfaces in Fig. 3.1(d) give examples of such connections. The cost of an interpolated-surface connection is defined to be

$$l \cdot C_{\text{HU}}(I_{\text{max}}).$$

(3.13)

Here, $l$ represents the length of the connection’s medial axis in mm, $I_{\text{max}}$ the maximum HU value encountered along the medial axis and

$$C_{\text{HU}}(I) = \begin{cases} 
1, & I \leq -950 \text{ HU} \\
1 + 0.02 \cdot (I + 950), & -950 < I \leq -900 \\
2 + 0.09 \cdot (I + 900), & -900 < I \leq -800 \\
11 + 0.15 \cdot (I + 800), & -800 < I \leq -700 \\
\min(500, 26 + 0.8 \cdot (I + 700)), & \text{otherwise.}
\end{cases}$$

(3.14)

The cost function in (3.14) assigns low values to voxels with intensity values below -900 HU, which are almost certain to be air, and high values to voxels with intensity values greater than -200 HU, which are likely to be soft tissue. Voxels with intensity values in the range
between -900 HU and -200 HU are likely to straddle a boundary between air and soft tissue and receive intermediate cost. The exact form of (3.14) was determined experimentally.

The voxels of the connection are extracted via region growing constrained by the interpolated surface and seeded with a 26-connected voxel approximation to the connection medial axis.

The final connection type joins a branch segment directly to the conservative segmentation using a locally-adaptive region-growing algorithm. The algorithm is seeded at the center of the proximal end-cap of the branch segment. Such connections are assigned a cost of

\[
\frac{V}{V_{\text{exp}}}C_{\text{HU}}(\tau_c) + 300,
\]

where \(V\) is the volume of the segmented connection (in mm\(^3\)), \(V_{\text{exp}}\) is the expected volume of the connection given by the average cross-sectional area of the target branch (in mm\(^2\)) times the minimum distance between the target branch and a voxel of the conservative segmentation (in mm), \(\tau_c\) is the threshold at which the connection reaches the conservative segmentation, and 300 is a constant penalty term. In practice, the volume ratio penalizes leakage and the constant penalty term ensures that such connections are used infrequently in favor of the more tightly constrained interpolated-surface connection.

The tree \(T_{\text{conn}}\) is constructed from a rooted digraph \(D\) using Prim’s minimum spanning tree algorithm [158]. As in \(T_{\text{conn}}\), the root of \(D\) represents the conservative segmentation and the non-root vertices of \(D\) represent branch segments. An edge connecting the root to a non-root vertex is added to \(D\) for each branch segment lying within 20 mm of the conservative segmentation. Edges are added in both directions between pairs of vertices
representing branch segments separated by less than 20mm. Many interpolated surface connections are possible between two such branch segments. We discard anatomically implausible connections using a set of geometric tests similar to those in (3.12) and select the plausible connection minimizing (3.13).

3.1.5 Global graph-partitioning algorithm

To this point, we have steadfastly avoided deciding which connected components, airway sections, and branch segments represented true airway signals and which represented spurious signals. Thus, the tree $T_{\text{conn}}$ may contain as many as 1,500 branch segments, many of which do not belong in the airway tree. The final stage of the proposed segmentation algorithm is therefore a global graph-partitioning algorithm that retains valid branch segments and discards spurious branch segments.

The graph-partitioning algorithm operates on a hierarchical tree $T$ on $V$ vertices labeled $\{0, 1, ..., V - 1\}$ with root$(T) = 0$. Each non-root vertex $k > 0$ has a unique parent $P[k]$, a non-negative cost $c_k$, and a non-negative benefit $b_k$. As in Section 3.1.4, the root of $T$ represents the conservative segmentation and non-root vertices of $T$ represent branch segments. The benefit $b_k$ of a non-root vertex is its branch segment score, as defined in Section 3.1.3, and the cost $c_k$ is the cost of its connection to its parent in $T_{\text{conn}}$, as defined in Section 3.1.4. Vertices associated with strong branches therefore have high benefit, while branches separated from their parent by a high HU barrier have high cost. The root vertex is automatically part of the segmentation, so $b_0$ and $c_0$ are both assumed to be zero. The algorithm’s output is a subtree of $T$ represented by a binary vector, $t = (t_0, t_1, ..., t_{V-1})^T$. As $t$ must include the conservative segmentation, $t_0 = 1$. Also, $t_k = 1 \Rightarrow t_{P[k]} = 1$ must
hold for all $k > 0$ to ensure $t$ describes a connected segmentation; i.e., if a branch is included in the segmentation, its parent must also be included.

A natural measure for the “quality” of a vertex $k$ is the ratio $b_k/c_k$, which represents the benefit per unit cost of adding $k$ to the segmentation. This suggests a naive algorithm that defines $t$ to include all vertices connected to the root by a path consisting only of vertices with benefit/cost ratios greater than some threshold $r$. The toy example in Fig. 3.4 illustrates a problem with this approach. Here, the ratio threshold is $r = 1.0$. The subtree suggested by the naive algorithm misses many low-cost, high-benefit vertices because it does not look beyond the poor ratio at vertex 1.

This problem can be overcome by considering $T$ globally and examining the maximum achievable cumulative benefit for a subtree as a function of cumulative cost. Define the cost of a subtree to be

$$C(t) = \sum_{k=0}^{V-1} t_k c_k$$

(3.16) and its benefit to be

$$B(t) = \sum_{k=0}^{V-1} t_k b_k.$$  

(3.17)

Given a cost budget $\gamma$, the maximum achievable benefit is

$$\beta(\gamma) = \max\{B(t) \text{ such that } t \text{ is a subtree of } T \text{ and } C(t) \leq \gamma\}.$$  

(3.18)

Equation (3.18) is the Tree Knapsack Problem (TKP) and the function $\beta(\gamma)$ is the TKP curve [159]. The solid line in Fig. 3.4(b) plots $\beta$ for the example in Fig. 3.4(a). The TKP curve is a nondecreasing function of $\gamma$. Rather than apply a ratio threshold to each vertex
independently, a better approach inspects the global TKP curve and determines the solution point at which the marginal cumulative benefit-to-cost ratio dips below the ratio threshold $r$. Unfortunately, the TKP is NP-Hard, and, although reasonably efficient algorithms exist when the $c_k$ are small integers, tracing the TKP curve is expensive. Furthermore, the TKP curve is piecewise-constant and it can be difficult to pinpoint where the ill-defined “slope” of the curve drops below $r$.

Figure 3.4. A toy example illustrating the proposed global graph-partitioning algorithm. (a) The tree $T$. The benefit ($b_k$, top value) and cost ($c_k$, bottom value) of a vertex are listed at its left. (b) A plot of cumulative benefit versus cumulative cost. The solid blue line and dotted red line trace the tree knapsack problem and relaxed tree knapsack problem curves defined in (3.18) and (3.20). The black ‘x’ represents the cost/benefit pair achieved by the output of Algorithm 3.1 for $r = 1.0$. The solution is both a point of intersection of the two curves and the breakpoint at which the slope of the RTKP curve drops below $r$. The algorithm output achieves benefit $B(t^*) = 419$ for cost $C(t^*) = 210$ and excludes only vertices 11, 12, 14, and 15. The black ‘o’ represents the cost/benefit pair achieved by the output of a naive algorithm described in the text. The naive algorithm fails to segment the high-benefit, low-cost vertices 4, 5, 9, and 10 because it cannot see beyond vertex 1.

Instead, consider the “envelope” function defined by a continuous-valued relaxation of the TKP. Define a relaxed subtree of $T$ to be a real-valued vector $f$ satisfying:

$$f_k \in [0, 1] \ \forall k, \quad f_0 = 1, \quad \text{and} \quad f_k \leq f_{P[k]} \ \forall k > 0.$$  (3.19)
Every subtree is a relaxed subtree, but the reverse is not generally true. Define $C(f)$ and $B(f)$ as in (3.16) and (3.17) and, for a cost budget $\gamma$, define the maximum-achievable benefit for a relaxed subtree to be

$$\beta_R(\gamma) = \max \{ B(f) \text{ such that } f \text{ is a relaxed subtree of } T \text{ and } C(f) \leq \gamma \}. \tag{3.20}$$

Equation (3.20) is the Relaxed Tree Knapsack Problem (RTKP) and the function $\beta_R(\gamma)$ is the RTKP curve [160]. The dotted line in Fig. 3.4 plots $\beta_R$. The RTKP curve is nondecreasing and provides an upper-bound for the TKP curve. Also, $\beta_R$ is a piecewise-linear function of $\gamma$ with non-increasing slope as (3.20) represents a parametric linear programming problem [161, 162]. Thus, for any choice of $r$, the RTKP curve has a well-defined breakpoint below which its slope is at least $r$ and above which its slope is at most $r$.

Remarkably, there exists a simple algorithm, which has not, to our knowledge, been presented elsewhere, for locating the slope breakpoint without tracing the RTKP curve. Furthermore, for any $r$, the slope breakpoint occurs at a point of intersection between the RTKP and TKP curves and is therefore achievable by a non-relaxed subtree of $T$. The solution to the TKP at this point on the curve is given by the subtree

$$t^* = \arg \max \{ B(t) - rC(t) \text{ such that } t \text{ is a non-relaxed subtree of } T \}. \tag{3.21}$$

Appendix B provides proof of these claims. Algorithm 3.1 gives a linear-time method for locating $t^*$.

Our current implementation uses a conservative slope breakpoint of $r = 3.0$. Thus, the
segmentation terminates when the marginal global utility of adding additional branches drops to less than three times the marginal global cost of connecting the newly added branches to the tree. The returned segmentation is the union of the conservative segmentation, the lumen voxels of branch segments for which \( t^*_k = 1 \), and the segmented connection voxels joining such branch segments to their parents in \( T_{\text{conn}} \).

**Algorithm 3.1** Locate optimal subtree \( t^* = \arg \max \{B(t) - rC(t)\} \) such that \( t \) is a non-relaxed subtree of \( T \)

1: Let \( \{m_j : j = 0, 1, ..., V - 1\} \) be a depth-first ordering of the vertices in \( T \).
2: // Note that \( m_i = P[m_j] \Rightarrow i > j \). Specifically, \( m_{V-1} = 0 \), the root of \( T \).
3: \( S_k \leftarrow 0 \) for all \( k \in \{0, ..., V - 1\} \)
4: // \( S_k \) is the maximum achievable score for a subtree of \( T \) rooted at \( k \). On termination, \( S_0 = B(t^*) - rC(t^*) \).
5: \( v_k \leftarrow 0 \) for all \( k \in \{0, ..., V - 1\} \)
6: // Binary indicator variables used to reconstruct \( t^* \). Here, \( v_k = 1 \Rightarrow k \) is needed by \( P[k] \) to achieve \( S_{P[k]} \).
7: for all \( j = 0 : V - 1 \) do
8:    \( S_{m_j} \leftarrow b_{m_j} - r_{m_j} \)
9:    for all \( k \) such that \( P[k] = m_j \) do
10:       // Because the vertices are considered in a depth-first order, \( S_k \) has
11:       // already been computed.
12:          if \( S_k \geq 0 \) then
13:             \( S_{m_j} \leftarrow S_{m_j} + S_k \)
14:             \( v_k \leftarrow 1 \)
15:          end if
16:    end for
17: end for
18: // A top-down algorithm builds \( t^* \) from the \( \{v_k\} \).
19: for all \( j = V - 1 : 0 \) do
20:    if \( t^*_{m_j} = 1 \) then
21:        for all \( k \) such that \( P[k] = m_j \) do
22:            if \( v_k = 1 \) then
23:                \( t^*_k \leftarrow 1 \)
24:            end if
25:        end for
26:    end if
27: end for
28: return \( t^* \)
3.2 Interactive Tool Suite

The graph partitioning algorithm described in Section 3.1.5 highlights the global character of the proposed automatic algorithm. The algorithm searches the entire lung volume for branches, and considers all potential branches while making its final segmentation decisions. Thus, locally expensive connection costs that occur early in the tree can be overcome because the algorithm can “see” strong downstream branches that make incurring such costs worthwhile. Segmentation decisions for the last few visible generations of airway, however, must be made using primarily local image data. As peripheral branches frequently have weak image signatures and can appear to be disconnected from the rest of the airway tree, the automatic algorithm can miss true branches and mistakenly extract extra false branches.

Accurately segmenting these last few generations is vital for peripheral airway applications, such as planning and guiding bronchoscopy to distant lesions [21, 36, 38, 39]. Here, the extracted airways serve two purposes. First, they provide a route for the bronchoscope to follow towards a peripheral region of interest (ROI) [42]. Second, the airways that break off from the main route provide important visual landmarks for guiding the bronchoscope to its destination. Two key interactive segmentation tasks therefore arise in planning for and guiding bronchoscopy:

1. **Route extension**—The user appends airways to an existing route. The extended path may lead the bronchoscope closer to the peripheral ROI or provide a better angle from which to view or biopsy the ROI.

2. **Visual landmark extraction**—Once an appropriate route has been identified and,
if necessary, extended, the user scans the entire route and extracts any small missed branches that may be visible to a bronchoscope traveling along the route.

To accomplish these related tasks, we have developed a suite of tools for interactively cleaning and extending an automatically-segmented tree. The interactive tools are run through a graphical user interface (GUI) on a Windows PC. The user interacts with the segmentation and CT image in a 3D environment built using Visual C++ and OpenGL [163,164]. The software is maintained in Visual Studio .NET 2003. Appendix D presents a complete user manual for the segmentation software.

Errors introduced during the graph-partitioning algorithm described in Section 3.1.5 can be corrected with a few mouse clicks, as the branches and their inter-connections are simply the vertices and edges of a graph-theoretic tree. Thus, spurious branches erroneously included in the tree can be deleted and true branches separated from the segmented tree by stenosis, motion, or image noise can be added.

A more challenging interactive segmentation task is the extraction of weak, single-voxel-thick airways that are missed by the airway section filter. To assist the user in locating and segmenting such small peripheral airways, the system provides a fused view in which the 3D airway tree is combined with an oblique cross-sectional slice through the image data. Fig. 3.5 provides example views with the airway tree rendered at a global level. The oblique slice behaves as a projection screen for the image data—it holds a fixed position parallel to the viewing plane, but displays continually-updated cross-sections as the 3D scene is rotated and panned. The slice allows the user to quickly scan the segmented tree and locate small airways missed by the automatic algorithm. Once a missing airway has been identified, the user can zoom in to a local view of the missing airway. Fig. 3.5 provides an example of a
close-up local view.

Figure 3.5. A global view of the airway tree in the proposed interactive tool suite (case 20349-3-24). (a) The oblique cross-section view fused with the segmented airway tree. The cross-section is focused on the target ROI, which lies in the right lower lobe and is rendered in red. (b) A second global view of the same case rendered from the same location. Here, the automatic segmentation has been reduced to its medial axes and the ROI has been hidden to reduce visual clutter.

Figure 3.6. An illustration of the livewire capability of the proposed system. (a) A close-up view of the oblique cross-section view fused with the automatically segmented airway tree. The oblique slice is isotropically sampled with pixel dimension 0.5mm and windowed to HU∈ [−1000, −200]. The arrow highlights a small peripheral airway missed by the automatic algorithm. (b) Livewire operation. The red line connecting the segmentation to the mouse cursor indicates voxels that will be added to the segmentation.

To segment airways captured in the oblique slice, the system provides a “livewire” capability [41, 55, 56]. The user selects a seed point in the segmentation and hovers the
mouse cursor over a point along the missing airway. Dijkstra’s shortest-paths algorithm computes an eight-connected path from the seed point to the cursor [158]. Pixel weights favor paths through low-intensity regions. When the user selects an acceptable path, the path’s end point becomes a new seed point and the process is repeated until the branch has been completely extracted. As branches may not lie entirely in any single plane, the user can freely rotate the 3D scene at any time to obtain a better view. Pixel weights and shortest-path computations are updated on-the-fly. As livewire is performed in arbitrary oblique cross-sections, the procedure returns a piecewise-planar sequence of floating-point 3D image locations. A post-processing step produces a 26-connected sequence of integer-valued voxel coordinates, which are added to the segmentation.

3.3 Results

This section presents experimental results validating the proposed segmentation system. The results are organized in two parts. Section 3.3.1 describes results obtained using only the automatic algorithm. Section 3.3.2 describes an application of both the automatic and interactive aspects of the proposed system to the problem of segmentation for image-based planning and guidance of bronchoscopy to the periphery.

3.3.1 Automatic Algorithm

We have applied the proposed automatic segmentation algorithm to more than 40 MDCT chest scans to date. The images were acquired by four different scanners and reconstructed using three different kernels. All were successfully segmented using the same set of algorithm parameters. Table 3.1 details the algorithm's running time, which averages less than three
minutes on a dual-core 2.6GHz PC with 4GB RAM running Windows XP.

<table>
<thead>
<tr>
<th>Segmentation step (section number)</th>
<th>Mean running time (in seconds)</th>
<th>Standard deviation (in seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conservative segmentation (3.1.1)</td>
<td>4.3</td>
<td>0.9</td>
</tr>
<tr>
<td>Isotropic interpolation (3.1.2.1)</td>
<td>21.4</td>
<td>5.8</td>
</tr>
<tr>
<td>Connected component filter (3.1.2.2)</td>
<td>98.5</td>
<td>37.5</td>
</tr>
<tr>
<td>Airway section construction (3.1.2.3)</td>
<td>14.0</td>
<td>3.7</td>
</tr>
<tr>
<td>Branch segment definition (3.1.3)</td>
<td>22.6</td>
<td>3.1</td>
</tr>
<tr>
<td>Branch segment connection (3.1.4)</td>
<td>4.4</td>
<td>0.8</td>
</tr>
<tr>
<td>Graph partitioning algorithm (3.1.5)</td>
<td>&lt; 0.1</td>
<td>0.0</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>2 min 46 sec</strong></td>
<td><strong>41.7</strong></td>
</tr>
</tbody>
</table>

Table 3.1. Proposed automatic segmentation algorithm running times on a dual-core 2.6GHz PC with 4GB RAM running Windows XP. Running times are averaged over ten cases. No case required more than four minutes of processing time.

Fig. 3.7 depicts results comparing segmentations obtained using the proposed algorithm with those obtained by a previously proposed adaptive region-growing algorithm for nine typical cases [1,9,11]. The threshold for the adaptive region-growing algorithm was manually tuned to achieve the best possible tree and, when reasonable, small leakages were manually removed from the region-growing result. Even in cases where the region-growing approach appears to perform well, the proposed algorithm locates significant additional airways.

We have also compared both the proposed and adaptive region-growing algorithms with two other existing segmentation algorithms. The existing algorithms include a morphological algorithm based on published work by Aykac et al. and a hybrid algorithm proposed by Kiraly et al., which includes elements based both upon region growing and mathematical morphology [7,9]. All four algorithms (proposed, adaptive region growing, morphological, and hybrid) were compared with manually-defined “gold-standard” trees for three scans (21405-3a, h002-tlc, and 20349-3-39, with gold-standard trees consisting of 271, 340, and 501 branches). The scans were chosen to represent the typical output of the proposed and
Figure 3.7. Visual comparison of human airway-tree segmentations produced by the proposed automatic method and a previously-proposed adaptive region-growing approach [1,9,11]. The figures are surface renderings of the binary segmentation results. Blue voxels were segmented by both methods. Green voxels were uniquely segmented by the proposed method and red voxels were uniquely segmented by the region-growing method. Case numbers: (a) h001, (b) h007, (c) 21405-3a, (d) 21405-16, (e) 20349-3-7, (f) 20349-3-15, (g) 21405-57, (h) 21405-58, (i) 21405-60. MDCT scanners: (a)-(b) Phillips MX8000, (c)-(f) Siemens Sensation 16, (g)-(h) Siemens Emotion 16, (i) Siemens Sensation 40. Reconstruction kernels: (a)-(b) D kernel, (c)-(d) b31 kernel, (e)-(g),(i) b50 kernel, (h) b41 kernel.

adaptive region-growing algorithms on scans acquired from different sources (Phillips and Siemens scanners) and reconstructed with different kernels (b31, b50, D).
The proposed algorithm was run with one set of algorithm parameters as described in the text. The adaptive region-growing algorithm used a hand-tuned global HU threshold for each scan. Leakages were pruned from the region-growing result using the proposed interactive toolkit. The morphological and hybrid algorithms were run multiple times using different algorithm parameters on both raw and prefiltered image data [9]. The best result for each case was selected for comparison. Both the hybrid and morphological algorithms were slow. The hybrid algorithm required between 13 and 22 minutes per segmentation and the morphological algorithm required between 118 and 153 minutes per segmentation on a dual-core 3.2GHz PC with 4GB RAM running Windows XP. As both algorithms require multiple runs for each scan, neither is practical for clinical application.

Table 3.2 summarizes the results of the gold-standard comparison. All algorithms successfully extracted the main and lobar bronchi in each case. Performance differences between the four algorithms began to appear at the segmental level, where the adaptive region-growing algorithm missed one segmental bronchus entirely ($LB^*$ in 21405-3a), and extracted only a small portion of two others ($LB^5$ in 21405-3a and $RB^7$ in h002-tlc). Performance difference become dramatic in the periphery, where the proposed algorithm significantly outperforms both the hybrid and morphological approaches and identifies more than twice as many branches as the adaptive region-growing algorithm.

The proposed algorithm produced only two false-positive branches. In comparison seven areas of segmentation leakage were manually pruned from the three region-growing results. The hybrid and morphological results each exhibited five small areas of leakage. In addition, the hybrid algorithm exhibited one significant area of leakage in case 21405-3a. Finally, we note that the proposed algorithm also identified nine true branches that were missing from
the gold-standard trees, but were verified by direct inspection of the image data. These missed branches account for less than 1% of the 1,112 total manually-defined branches for the three cases. Appendix C presents full visual results of the segmentations used for the gold-standard comparison.

<table>
<thead>
<tr>
<th>Bronchial order</th>
<th>Total no. branches</th>
<th>Total</th>
<th>Proportion of Correctly Extracted Airways</th>
</tr>
</thead>
<tbody>
<tr>
<td>Main/lobar</td>
<td>53</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>Segmental</td>
<td>59</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>1(^{st}) gen. subseg.</td>
<td>116</td>
<td>96%</td>
<td>100%</td>
</tr>
<tr>
<td>2(^{nd}) gen. subseg.</td>
<td>200</td>
<td>91%</td>
<td>99%</td>
</tr>
<tr>
<td>≥ 3(^{rd}) gen. subseg.</td>
<td>684</td>
<td>65%</td>
<td>96%</td>
</tr>
</tbody>
</table>

Table 3.2. Comparison of proposed automatic algorithm and three existing algorithms to manually defined “gold standard” trees for three cases. Here, “ARG” refers to an adaptive region-growing algorithm [1, 9], “Morph.” is a morphological algorithm [7, 9], and “Hybrid” is a hybrid algorithm combining elements of both region-growing and mathematical morphology [9]. The cases considered were 21405-3a (image dimensions: 512×512×606, voxel dimensions: Δx = Δy = 0.67mm, Δz = 0.50mm, CT scanner: Siemens Sensation 16, Convolution kernel: b31), h002-tlc (512×512×515, Δx = Δy = 0.59mm, Δz = 0.60mm, Phillips MX8000, D), and 20349-3-39 (512×512×547, Δx = Δy = 0.57mm, Δz = 0.50mm, Siemens Sensation 40, b50). Anatomical labels were assigned following the nomenclature of Netter [141].

3.3.2 Segmentation for Image-Guided Peripheral Bronchoscopy

Segmentations produced using the automatic and interactive methods described in this paper have been used to help plan and guide live bronchoscopies to peripheral lesions in 15 cases to date [21, 38]. This section reports quantitative results for all 15 cases and provides illustrative examples for two cases.

The proposed system provides airway segmentations for image-guided bronchoscopy applications using the following procedure. First, the patient receives a CT chest scan and the physician indicates one or more target peripheral ROIs. Next, the automatic algorithm produces a full global tree, from which a route leading the bronchoscope to the
ROI is identified [42]. If necessary, this route is extended using the interactive toolkit. The entire route is then inspected and any small, missed branches that will be visible to the bronchoscope as it traverses the route are added. Finally, the extended segmentation serves as input for algorithms that extract centerlines, endoluminal surfaces, and other information necessary to provide live guidance during the procedure [21, 23, 24, 40].

![Diagram](image)

**Figure 3.8.** An example illustrating the numbering scheme used to index interactively-added branches in Table 3.3. (a) The automatic algorithm extracts a nearly-complete route to the ROI. (b) The interactive toolkit is used to extend the route and extract small visual landmark branches along the way. Here, a generation five airway is added to the route and visual landmarks are added at generations two and four.

Segmentation results for the 15 live-guidance cases processed to date are summarized in Table 3.3. Guidance was performed for a total of 31 peripheral ROIs with routes traversing as many as 13 airway generations. The automatic algorithm produced no false branches along any route in the study. The automatically-segmented route was complete or nearly-complete in most cases. More than half of the automatically-segmented routes missed zero or one branches and more than three-quarters missed two or fewer branches. Most of the interactively-added branches were located deep in the periphery. The median generation
<table>
<thead>
<tr>
<th>Patient</th>
<th>ROI location (lobe)</th>
<th>Route length (airway gens.)</th>
<th>Route branches added (gen. #)</th>
<th>Visual landmark branches added (gen. #)</th>
<th>Total num. branches added</th>
</tr>
</thead>
<tbody>
<tr>
<td>20349-3-24</td>
<td>RLL</td>
<td>13</td>
<td>none</td>
<td>8,12</td>
<td>2</td>
</tr>
<tr>
<td>20349-3-25</td>
<td>LUL</td>
<td>8</td>
<td>none</td>
<td>7</td>
<td>1</td>
</tr>
<tr>
<td>20349-3-26</td>
<td>LUL</td>
<td>8</td>
<td>8</td>
<td>none</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>RUL</td>
<td>8</td>
<td>8</td>
<td>none</td>
<td>1</td>
</tr>
<tr>
<td>20349-3-28</td>
<td>RUL</td>
<td>9</td>
<td>none</td>
<td>none</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>LUL</td>
<td>12</td>
<td>none</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td>20349-3-29</td>
<td>LUL-1</td>
<td>4</td>
<td>none</td>
<td>none</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>LUL-2</td>
<td>13</td>
<td>10,11,12,13</td>
<td>9,10,11,12</td>
<td>8</td>
</tr>
<tr>
<td>20349-3-30</td>
<td>RLL</td>
<td>11</td>
<td>none</td>
<td>8,9</td>
<td>3</td>
</tr>
<tr>
<td>20349-3-33</td>
<td>RUL</td>
<td>13</td>
<td>13</td>
<td>10,12</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>RLL</td>
<td>10</td>
<td>8,9,10</td>
<td>8,9</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>LLL</td>
<td>12</td>
<td>none</td>
<td>9,10</td>
<td>2</td>
</tr>
<tr>
<td>20349-3-34</td>
<td>LUL</td>
<td>9</td>
<td>9</td>
<td>7</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>RML</td>
<td>7</td>
<td>none</td>
<td>none</td>
<td>0</td>
</tr>
<tr>
<td>20349-3-35</td>
<td>RML</td>
<td>7</td>
<td>7</td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>RUL</td>
<td>7</td>
<td>none</td>
<td>none</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>LUL</td>
<td>7</td>
<td>none</td>
<td>none</td>
<td>0</td>
</tr>
<tr>
<td>20349-3-37</td>
<td>RUL</td>
<td>7</td>
<td>7</td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>RUL</td>
<td>4</td>
<td>none</td>
<td>none</td>
<td>0</td>
</tr>
<tr>
<td>20349-3-38</td>
<td>RUL-1</td>
<td>8</td>
<td>5,6,7,8</td>
<td>6,7</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>RUL-2</td>
<td>8</td>
<td>5,6,7,8</td>
<td>6,7</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>LUL</td>
<td>13</td>
<td>none</td>
<td>10,12</td>
<td>2</td>
</tr>
<tr>
<td>20349-3-39</td>
<td>RLL</td>
<td>5</td>
<td>none</td>
<td>none</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>LLL</td>
<td>7</td>
<td>none</td>
<td>none</td>
<td>0</td>
</tr>
<tr>
<td>20349-3-40</td>
<td>RML-1</td>
<td>4</td>
<td>none</td>
<td>none</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>RML-2</td>
<td>4</td>
<td>none</td>
<td>none</td>
<td>0</td>
</tr>
<tr>
<td>20349-3-41</td>
<td>RLL-1</td>
<td>13</td>
<td>none</td>
<td>11,12</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>RLL-2</td>
<td>12</td>
<td>none</td>
<td>10</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>RUL</td>
<td>8</td>
<td>none</td>
<td>none</td>
<td>0</td>
</tr>
<tr>
<td>20349-3-42</td>
<td>RUL-1</td>
<td>8</td>
<td>none</td>
<td>none</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>RUL-2</td>
<td>9</td>
<td>none</td>
<td>6</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 3.3. Summary of results for 15 cases in which segmentations obtained using the proposed system were used to help plan and guide live peripheral bronchoscopic procedures. The patients are numbered according to IRB Protocol 20349-3. Route length refers to the number of airway generations traversed on the path leading from the trachea to the peripheral ROI. The route and visual landmark branches added using the interactive toolkit are indexed by generation number according to the convention illustrated in Figure 3.3. The two RUL routes for case 20349-3-38 involved the same branches. The routes terminated at different points in the 8th generation branch. Note that it was not possible to insert an ultrathin 2.8-mm diameter bronchoscope beyond the 6th bronchial generation along either these routes or the 13-generation LUL route in case 20349-3-29.
Figure 3.9 illustrates the operation of the proposed system for case 20349-3-29. The illustrated route is the worst result obtained using the automatic algorithm in terms of missing branches, requiring four generations of route extension and the addition of four visual landmark branches. The missed branches are very small, however, and would be difficult to detect for any automatic algorithm. The oblique slice in Fig. 3.9 (d), for example, depicts the missing extended route branch at generation 10 and the missing visual landmark branch at generation 9. The image signatures of the generation 11-13 airways are even weaker. This case therefore illustrates the necessity of the proposed interactive toolkit for peripheral applications rather than a particular weakness of the proposed automatic algorithm.

Case 20349-3-24 provides a striking example illustrating the necessity of visual landmarks for image-based guidance. In this case, the automatically-segmented tree provides a complete route to the lesion but misses two small visual landmark branches at generations 8 and 12 along the route. The branches comprise less than 0.1% of the total segmentation volume, but dramatically affect virtual bronchoscopic views encountered along the path to the lesion. Consider the VB renderings in Fig. 3.10 (b) and (c). Here, the interactively-added generation 8 branch changes the VB rendered view from a bifurcation to a trifurcation. A bronchoscopic video view captured during the live procedure confirms the correctness of the extended result.
Figure 3.9. Illustration of the complete airway segmentation system for case 20349-3-29. The green object is a physician-defined peripheral lesion [41]. The blue line represents the best 3D path to the lesion for each segmentation [42]. (a) **Adaptive region growing**—The best achievable segmentation using adaptive region growing with a manually-selected threshold. The best 3D route stops 74mm short of the lesion. (b) **Proposed automatic algorithm**—The segmentation produced by the proposed algorithm adds an entire subtree in the left upper lobe, but the best 3D route is still 38mm short of the lesion. (c) **Route extension**—The proposed interactive tool suite extends the best 3D route from (b) to within 15mm of the lesion. The route is still ill-suited for image-guided bronchoscopy as it lacks several key bifurcations. (d) **Visual landmark extraction**—The addition of four small branches along the extended route completes the result. (e),(f) Illustration of the interactive livewire tool during the route-extension step. Note the small airway at the bottom that will be added during visual landmark extraction. Pixels in the oblique cross section are isotropically sampled at 0.5mm and pixel intensities are windowed to HU ∈ [−1000, −200].
Figure 3.10. Example illustrating the necessity of visual landmark extraction. (a) Exterior view of the automatically segmented airway tree for case 20349-3-24. The red object is a physician-defined peripheral lesion. The blue line represents the best 3D route to the lesion. Both (b) and (c) present endoluminal renderings with the virtual bronchoscope positioned in the 8th generation airway along the route. The rendering in (b) was generated using the automatically segmented result. The segmentation used to generate (c) was extended using the livewire functionality of the proposed system. (d) A video frame captured with a 2.8mm Olympus XP160F ultrathin bronchoscope confirms the existence of the additional branch. The endoluminal surfaces used to produce the VB renderings in this figure were generated by the method of Gibbs et al. [21].
Chapter 4

Anatomical-Tree Matching

This chapter describes the methods used for matching two graph-theoretic tree models extracted from 3D images of the same underlying anatomical tree. The chapter is organized as follows. Section 4.1 introduces basic notation and poses tree matching as a graph-theoretic optimization problem. Section 4.2 details the search space and cost function of the optimization problem, develops a number of useful theoretical results, and describes an efficient DP algorithm for locating a globally-optimal match. Finally, Section 4.3 presents experimental results validating the algorithm’s performance on real anatomical trees.

4.1 Problem Statement and Notation

Let \( \mathbb{T} \) represent the set of hierarchical trees. Let \( T \in \mathbb{T} \) be one such tree with vertex set \( V(T) \), root vertex \( \text{root}(T) \), and directed edge set \( E(T) \subseteq V(T) \times V(T) \). Vertices and edges in \( T \) represent branchpoints and branches in the physical tree and are augmented with geometrical attributes such as branchpoint locations and branch lengths measured from the 3D image data.
We pose tree-matching as an optimization problem. Define a match between two trees $T_1$ and $T_2$ to be a one-to-one function $\phi$ with domain $D(\phi) \subseteq V(T_1)$ and range $R(\phi) \subseteq V(T_2)$ that maps vertices in $T_1$ to corresponding vertices in $T_2$. It is useful to place further restrictions on the $\phi$, as searching the space of all one-to-one functions between $V(T_1)$ and $V(T_2)$ is typically infeasible and many such functions represent physically meaningless matches. Valid matches, for example, will be required to map the root of $T_1$ to the root of $T_2$. Such restrictions implicitly define a set of valid matches, denoted $\Phi(T_1, T_2)$. A similarity measure $S(T_1, T_2 | \phi)$ evaluates each valid match $\phi \in \Phi(T_1, T_2)$ using branch and branchpoint attributes. We seek an optimal match $\phi^*[T_1, T_2]$ satisfying

$$\phi^*[T_1, T_2] \triangleq \text{arg max}\{S(T_1, T_2 | \phi) \mid \phi \in \Phi(T_1, T_2)\}.$$  

(4.1)

Denote the similarity achieved by $\phi^*[T_1, T_2]$ as $S^*[T_1, T_2]$.

We use $a_1, b_1, c_1, ...$ and $a_2, b_2, c_2, ...$ to represent vertices in $T_1$ and $T_2$. Vertices in a generic tree $T$, such as the one in Figure 4.1, are denoted without subscripts. Unless otherwise indicated, denote

$$r = \text{root}(T), \quad r_1 = \text{root}(T_1), \quad \text{and} \quad r_2 = \text{root}(T_2).$$

All directed edges point away from the root. Therefore, in Figure 4.1, $(b, e) \in E(T)$, but $(e, b) \notin E(T)$. A child of $u \in V(T)$ is any $v \in V(T)$ such that $(u, v) \in E(T)$. Equivalently, $u$ is the parent of $v$. Denote the set of children of $u$ in $T$ by $C^u_T$. Let $\delta(u) \triangleq |C^u_T|$ be the degree, or number of children, of $u$. Define the degree of $T$ as the maximum degree of its
vertices by \( \delta(T) \triangleq \max\{\delta(u) \mid u \in V(T)\} \). For example, \( \delta(i) = 2 \) and \( \delta(T) = 3 \) in Figure 4.1. Vertices with degree zero are called leaves. The leaves of the tree in Figure 4.1 are \( c, d, e, g, h, \) and \( j \).

A path in \( T \) is a graph \( P \) of the form

\[
V(P) = \{u^{(0)}, u^{(1)}, ..., u^{(n)}\},
\]
\[
E(P) = \{(u^{(0)}, u^{(1)}), (u^{(1)}, u^{(2)}), ..., (u^{(n-1)}, u^{(n)})\}
\]

where the \( u^{(i)} \in V(T) \) are all distinct vertices and \( E(P) \subseteq E(T) \). There is at most one path connecting any two vertices in a tree. If \( u, v \in V(T) \) are connected by a path in \( T \), it is denoted \( uPv \). In this case, \( u \) is a proper ancestor of \( v \) in \( T \), denoted

\[
u \rightsquigarrow v \text{ in } T.
\]

Otherwise, \( u \not\rightsquigarrow v \) in \( T \) [158]. In Figure 4.1, \( a \rightsquigarrow j \) in \( T \), with \( V(aPj) = \{a, f, i, j\} \) and

**Figure 4.1.** A hierarchical tree \( T \) with root(\( T \)) = \( r \).
\[ E(aPj) = \{(a, f), (f, i), (i, j)\}, \text{ but } f \not\rightarrow d \text{ in } T. \]

The ancestry relation \( \rightsquigarrow \) defines a tree ordering on \( V(T) \) [165]. Specifically, \( \rightsquigarrow \) is irreflexive \( (a \not\rightarrow a) \), antisymmetric \( (a \rightsquigarrow b \Rightarrow b \not\rightsquigarrow a) \), and transitive \( (a \rightsquigarrow b, b \rightsquigarrow c \Rightarrow a \rightsquigarrow c) \). Furthermore, as paths are unique in \( T \), \( a \rightsquigarrow c \) and \( b \rightsquigarrow c \) implies either \( a \rightsquigarrow b \) or \( b \rightsquigarrow a \).

Finally, the edge set of a tree can be determined from its ancestry relation \( \rightsquigarrow \), as

\[
(a, b) \in E(T) \iff a \rightsquigarrow b \text{ in } T \text{ and } \not\exists c \text{ such that } a \rightsquigarrow c \rightsquigarrow b \text{ in } T. 
\tag{4.4}
\]

The root vertex \( r \) is an ancestor of all \( v \in V(T) \). The number of edges of \( rPv \) is called the depth of \( v \) and denoted \( depth(v) \). The depth of a tree \( T \) is defined by

\[
\text{depth}(T) = \max\{\text{depth}(v) \mid v \in V(T)\}.
\]

The least common ancestor of two vertices, denoted \( \text{lca}(v, w) \), is the deepest vertex that is an ancestor of both \( v \) and \( w \) [158].

We will refer to two distinct types of subtrees of a tree \( T \). First, for any \( u \in V(T) \) define the bottom-up subtree of \( T \) at \( u \) to be \( T^u \), with

\[
V(T^u) = \{u\} \cup \{v \in V(T) : u \rightsquigarrow v \text{ in } T\}
\]

\[
E(T^u) = \{(v, w) \in E(T) : v \in V(T^u)\}. \tag{4.5}
\]

Thus, \( T^u \) is itself a tree with \( \text{root}(T^u) = u \) [123]. Second, for any subset of vertices \( K \subseteq V(T) \), let \( T[K] \) be the induced subtree of \( T \) obtained by deleting the vertices in \( V(T) \setminus K \) from \( T \). Here, deleting a vertex \( v \) entails making the children of \( v \) become the
Figure 4.2. An example of an induced subtree. (a) The tree $T$ from Figure 4.1. (b) An induced subtree of $T$ corresponding to the vertex set $K = \{r,b,c,f,g,j\}$. The tree $T[K]$ is obtained by deleting (coalescing) the vertices $a,d,e,i,$ and $h$ from $T$.

children of the parent of $v$ and then removing $v$ from $V(T)$ [132]. The resulting tree is characterized by the following theorem, which is proved in Appendix E.

Theorem 4.1 Consider $T \in \mathbb{T}$ and $K \subseteq V(T)$ with $r \in K$. Let $T[K]$ be obtained by deleting the vertices in $V(T) \setminus K$ from $T$ in any order. Then:

1. $T[K]$ does not depend on the order in which the vertices of $V(T) \setminus K$ are deleted

2. $V(T[K]) = K$ and $E(T[K]) = \{(u,v) | u,v \in K, u \rightsquigarrow v \text{ in } T, \text{ and } u \rightsquigarrow w \rightsquigarrow v \text{ in } T \Rightarrow w \notin K\}$

3. $\forall u,v \in K, u \rightsquigarrow v \text{ in } T[K] \iff u \rightsquigarrow v \text{ in } T$

4. $T[K]$ is a tree with $\text{root}(T[K]) = r$

The tree in Figure 4.2(b) is an induced subtree of the tree in Figure 4.2(a) with $K = \{r,b,c,f,g,j\}$. 
Finally, a function $\phi$ is an isomorphism between $T_1$ and $T_2$ if $\phi$ is a bijection between $V(T_1)$ and $V(T_2)$ with $\phi(r_1) = r_2$ such that $(u_1, v_1) \in E(T_1)$ iff $(\phi(u_1), \phi(v_1)) \in E(T_2)$. Intuitively, two trees are isomorphic if they have identical branching patterns.

## 4.2 Methods

Section 4.1 posed tree-matching as a global graph-theoretic optimization problem. This section describes methods used to construct and solve the optimization problem for practical tree-matching applications. The section is organized as follows. Section 4.2.1 defines the search space for the optimization. For tree matching, this search space corresponds to a set of “valid” matches between two input trees, which is defined using a model of common tree-extraction errors. Section 4.2.2 defines a similarity measure to evaluate valid matches. The similarity measure compares corresponding geometric attributes between matched branches and branchpoints. The measure favors, for example, matches that identify corresponding branches with similar lengths and branching angles. Section 4.2.3 develops theoretical results identifying several key properties of the set of valid matches and similarity measure. These properties are used to derive a recursive definition for the maximum-similarity match between two trees. Finally, Section 4.2.4 develops this recursion into an efficient DP algorithm for locating a globally-optimal match.

### 4.2.1 Valid Match Definition

This section defines the set of valid matches between two anatomical trees by modeling the extraction errors that likely affect the trees. The development proceeds in two steps. First, Section 4.2.1.1 defines valid matches for a simplified matching problem which assumes the
input trees suffer only from missing branches or extra false branches. Section 4.2.1.2 extends the set of valid matches to handle practical cases involving more severe errors.

4.2.1.1 Primary Deformation Model

This section defines a set of valid matches suitable for a simplified matching problem by focusing on the two most common errors affecting $T_1$ and $T_2$. First, branches that exist in the anatomical tree may be missing from an input tree. Second, extra false branches may appear in an input tree. The two errors have a similar effect on tree-matching in that both result in branches that are present in either $T_1$ or $T_2$, but not in both. Figure 4.3 gives an illustrative example. Here, branches that are present in only one of the input trees are colored red.

There is no general way to determine from $T_1$ and $T_2$ alone whether the uncorroborated red branches are indeed present in the physical tree. Our goals must therefore be limited to determining the correspondences between matching structures in the two trees, and highlighting those structures that do not match. For this reason, all uncorroborated branches are called “spurious,” with the tacit understanding that some uncorroborated branches may indeed exist in the ground-truth anatomical tree.

To match trees in the presence spurious branches, we hypothesize a model in which the observed input trees $T_1$ and $T_2$ arise as corrupted copies of a hidden “common” tree. This model is termed the primary deformation model, as it considers the two most common tree-extraction errors. The rest of the section is devoted to defining a set of requirements for determining whether or not a match $\phi$ is consistent with the model. Denote the set of such matches by $\Phi_P(T_1, T_2)$. 
The most important requirement is for valid matches to uncover a common tree between $T_1$ and $T_2$. Figure 4.3(a) illustrates this requirement. Here, the depicted match (with the function $\phi$ represented as a set of ordered pairs of corresponding vertices) is

$$\phi = \{(r_1, r_2), (b_1, a_2), (e_1, c_2), (h_1, e_2), (i_1, f_2), (j_1, h_2), (k_1, i_2), (l_1, j_2)\}.$$  

The matched vertices in $T_1$ and $T_2$ correspond to the domain and range of $\phi$. Specifically,

$$D(\phi) = \{r_1, b_1, e_1, h_1, i_1, j_1, k_1, l_1\} \quad \text{and} \quad R(\phi) = \{r_2, a_2, c_2, e_2, f_2, h_2, i_2, j_2\}.$$
The red vertices associated with “spurious” branches are exactly the vertices not matched by $\phi$; i.e. they are

$$V(T_1) \setminus D(\phi) = \{a_1, c_1, d_1, f_1, g_1, m_1\} \quad \text{and}$$

$$V(T_2) \setminus R(\phi) = \{b_2, d_2, g_2, k_2, l_2, m_2\}.\]

A common tree is recovered from $T_1$ and $T_2$ by deleting the red unmatched vertices to obtain the induced subtrees $T_1[D(\phi)]$ and $T_2[R(\phi)]$, depicted in Figure 4.3(b). Note that $T_1[D(\phi)]$ and $T_2[R(\phi)]$ have exactly the same branching structure; i.e., they are isomorphic. All valid matches under the primary deformation model must define a common tree in exactly this way. Specifically, any valid $\phi \in \Phi(T_1, T_2)$ must define an isomorphism between its induced subtrees $T_1[D(\phi)]$ and $T_2[R(\phi)]$.

**Figure 4.4.** A close-up view of a typical bifurcation in the common tree.

Additional requirements on valid matches can be obtained by focusing on a typical bifurcation in the common tree. Consider the situation in Figure 4.4 in which $u_1$, $v_1$, and $w_1$ are matched vertices and the unlabeled red vertices belong to spurious branches. Once the red vertices have been deleted, $u_1$ will be the parent of both $v_1$ and $w_1$. Similarly, $\phi(u_1)$
will be the parent of \( \phi(v_1) \) and \( \phi(w_1) \).

The spurious branches in Figure 4.4 have the effect of “splitting” edges of the common tree into paths traversing multiple edges in \( T_1 \) and \( T_2 \). The \((u_1, w_1)\) edge in the common tree, for example, is split into a path traversing two edges in \( T_1 \). Note that the paths connecting \( u_1 \) to its children in the common tree do not share any red vertices. This will be true in general under the primary deformation model, as no spurious branch can simultaneously split both the edge from \( u_1 \) to \( v_1 \) and the edge from \( u_1 \) to \( w_1 \). Formally,

\[
V(u_1 P v_1) \cap V(u_1 P w_1) = u_1 \text{ in } T_1. \tag{4.6}
\]

Recall here that \( V(u_1 P v_1) \) and \( V(u_1 P w_1) \) refer to the vertices encountered along the paths connecting \( u_1 \) to \( v_1 \) and \( u_1 \) to \( w_1 \) in \( T_1 \). An analogous statement holds for \( \phi(u_1), \phi(v_1) \) and \( \phi(w_2) \), where

\[
V(\phi(u_1) P \phi(v_1)) \cap V(\phi(u_1) P \phi(w_1)) = \phi(u_1) \text{ in } T_2. \tag{4.7}
\]

For \( \phi \) to be valid under the primary deformation model, all of the branchpoints in its common tree must behave like the example in Figure 4.4. Specifically, (4.6) and (4.7) must hold for any vertices \( u_1, v_1, \) and \( w_1 \) where \( v_1 \) and \( w_1 \) are children of \( u_1 \) in \( T_1[D(\phi)] \). Note that this requirement also extends naturally to branchpoints more than two children in the common tree, such as the trifurcation depicted in Figure 4.5.

Finally, it will be useful in practice to assume a loose upper bound on the number of times a branch of the common tree can be split by spurious branches. We set this bound to three. Thus, each edge in the common tree must correspond to a path traversing at most four edges in each of \( T_1 \) and \( T_2 \). The following definition summarizes the requirements on
valid matches under the primary deformation model.

**Definition 4.1** Consider trees $T_1$ and $T_2$. A function $\phi$ matching $r_1$ to $r_2$ is valid under the primary deformation model, denoted $\phi \in \Phi_P(T_1, T_2)$, if:

1. $\phi$ is an isomorphism between $T_1[D(\phi)]$ and $T_2[R(\phi)]$

2. $\forall u_1, v_1, w_1 \in D(\phi)$ with $(u_1, v_1), (u_1, w_1) \in E(T_1[D(\phi)])$
   
   (a) $V(u_1Pv_1) \cap V(u_1Pw_1) = u_1$ in $T_1$

   (b) $V(\phi(u_1)P\phi(v_1)) \cap V(\phi(u_1)P\phi(w_1)) = \phi(u_1)$ in $T_2$

3. $\forall(u_1, v_1) \in E(T_1[D(\phi)])$, $|E(u_1Pv_1)| \leq 4$ in $T_1$ and $|E(\phi(u_1)P\phi(v_1))| \leq 4$ in $T_2$

### 4.2.1.2 Secondary Deformation model

The primary deformation model is well-suited for matching anatomical trees that have been corrupted by spurious branches. Spurious branches are not, however, the only errors that corrupt our input trees. This section introduces a secondary deformation model that considers two additional errors. First, the branching order of corresponding subtrees may be reversed between $T_1$ and $T_2$. This situation is illustrated in Figure 4.6 where $T_1$ and $T_2$
disagree as to whether the rightmost subtree branches off before or after the leftmost subtree.

Second, a single branchpoint in $T_1$ or $T_2$ may correspond to multiple close branchpoints in the other tree. Figure 4.7 gives an example, in which the bifurcations at $u_1$ and $z_1$ in $T_1$ correspond to a trifurcation in $T_2$.

**Figure 4.6.** An example of a secondary deformation. Here, $T_1$ and $T_2$ disagree as to whether the rightmost or leftmost subtree branches off first. Because of the confusion, the vertex sets $\{u_1, z_1\}$ in $T_1$ and $\{\phi(u_1), z_2\}$ in $T_2$ are naturally treated as a trifurcation in the common tree.

**Figure 4.7.** A second example of a secondary deformation. Here, the two close bifurcations at $u_1$ and $z_1$ in $T_1$ correspond to a single trifurcation in $T_2$.

The new deformations illustrated in Figures 4.6 and 4.7 are naturally resolved by merging the blue highlighted branchpoints into a single “supernode” in the common tree. For example, the trifurcation in the common tree of Figure 4.6 encompasses the vertex pair $\{u_1, z_1\}$ in $T_1$ and the pair $\{\phi(u_1), z_2\}$ in $T_2$. The primary deformation model disallows
such matches and requires that each branchpoint in the common tree correspond to only a single vertex in both $T_1$ and $T_2$ as in the examples of Figures 4.4 and 4.5. We will therefore define an expanded set of valid matches under the secondary model, denoted $\Phi_S(T_1, T_2)$, which allows for the merging of branchpoints. Care must be taken, however, not to expand the set of valid matches too much by permitting unlimited branchpoint merging. Consider the example in Figure 4.8. Here, the match merges five vertices in $T_1$ and three vertices in $T_2$. The resulting supernode has six children in the common tree, even though both $u_1$ and $\phi(u_1)$ originally had only two. Such implausible situations can be eliminated by limiting the number of vertices merged into any single branchpoint to two. The following definition is necessary to formalize this idea.

**Figure 4.8.** A physically implausible match that is not considered to be valid under the secondary deformation model. The match merges five branchpoints in $T_1$ and three in $T_2$ to create a 6-furcation in the common tree. Vertex labels have been omitted to avoid clutter.

**Definition 4.2** Consider a tree $T$ and set of vertices $K \subseteq V(T)$ with $r \in K$. For each $u \in K$, define the supernode at $u$ in $T[K]$ to be

$$SN(T, u, K) = \{u\} \cup \{z \in V(T) \mid z \in V(uPv) \cap V(uPw) \text{ in } T \text{ for some } v, w \in C^u_{T[K]}\}. \quad (4.8)$$

Recall here that $C^u_{T[K]}$ represents the set of children of $u$ in $T[K]$. 

The supernodes associated with a match $\phi$ can be determined using Definition 4.2 as follows. Let $u_1$ be any matched vertex. Here, $SN(T_1, u_1, D(\phi))$ and $SN(T_1, \phi(u_1), R(\phi))$ represent the supernodes created by $\phi$ at $u_1$; i.e., the sets of vertices in $T_1$ and $T_2$ that are merged by $\phi$ into the $u_1, \phi(u_1)$ branchpoint in the common tree. In Figure 4.6 for example.

$$SN(T_1, u_1, D(\phi)) = \{u_1, z_1\} \text{ and } SN(T_1, \phi(u_1), R(\phi)) = \{\phi(u_1), z_2\}, \quad (4.9)$$

as $z_1$ lies along both $u_1Pv_1$ and $u_1Pw_1$ in $T_1$ and $z_2$ lies along both $\phi(u_1)P\phi(w_1)$ and $\phi(u_1)P\phi(x_1)$ in $T_2$. In Figure 4.9

$$SN(T_1, k_1, D(\phi)) = \{k_1, l_1\},$$

as $l_1$ lies along both $k_1Pm_1$ and $k_1Pn_1$ in $T_1$. Furthermore,

$$SN(T_2, b_2, R(\phi)) = \{b_2, f_2\} \text{ and } SN(T_2, a_2, R(\phi)) = \{a_2\}.$$ 

Intuitively, all the “normal” matched branchpoints in Figure 4.9 that are not shaded blue have trivial singleton supernodes and would be well-handled by the primary deformation model. The blue shaded regions with nontrivial supernodes necessitate the expanded set of valid matches provided by the secondary deformation model and summarized in the following definition.

**Definition 4.3** Consider trees $T_1$ and $T_2$. A function $\phi$ mapping $r_1$ to $r_2$ is valid under the secondary deformation model, denoted $\phi \in \Phi_S(T_1, T_2)$, if:

1. $\phi$ is an isomorphism between $T_1[\phi]$ and $T_2[\phi]$
2. \( \forall u_1 \in D(\phi), |SN(T_1, u_1, D(\phi))| \leq 2 \) and \( |SN(T_2, \phi(u_1), R(\phi))| \leq 2 \)

3. \( \forall (u_1, v_1) \in E(T_1[\phi]), |E(u_1Pv_1)| \leq 4 \) in \( T_1 \) and \( |E(\phi(u_1)P\phi(v_1))| \leq 4 \) in \( T_2 \)

In practice, it may be desirable to refine the restrictions in Definition 4.3. In some applications, it might be necessary to allow supernodes consisting of three or more vertices; e.g., if three close bifurcations must match to a single 4-furcation. Similarly, as anatomical trees are frequently embedded in a 3D Euclidean space and tree-extraction methods are unlikely to confuse distant branchpoints, it can be useful to require the vertices of a supernode to lie within a sphere of small radius. Some restriction of the supernodes is always necessary. Removing item 2 from Definition 4.3 entirely yields set of edit-distance mappings, and learning an optimal edit distance mapping is intractable for even the simplest cost functions [166].

![Figure 4.9](image-url)  
Figure 4.9. An example showing the extraction of a common tree from two trees corrupted by both primary and secondary deformations. The branches in red are present in either \( T_1 \) or \( T_2 \) but not in both. The branchpoints in green highlight areas corrupted by secondary deformations.
4.2.2 Similarity Measure Definition

Now that the set of valid matches between two anatomical trees has been defined via the secondary deformation model, we define a similarity measure for evaluating valid matches. The measure evaluates a match $\phi$ between trees $T_1$ and $T_2$ by comparing the vertex and edge attributes of corresponding branchpoints and branches via

$$S(T_1, T_2|\phi) = \sum_{u_1 \in V(T_1|\phi)} \sigma_v(u_1, \phi(u_1)) + \sum_{(u_1, v_1) \in E(T_1|\phi)} \sigma_e(u_1, v_1, \phi(u_1), \phi(v_1)). \quad (4.10)$$

Here, $\sigma_v$ is an affinity function comparing vertex attributes between two matched branchpoints $u_1$ and $\phi(u_1)$ and $\sigma_e$ is an affinity function comparing edge attributes between two matched branches $(u_1, v_1)$ and $(\phi(u_1), \phi(v_1))$ in the common tree. Many vertex and edge affinity functions are possible. We set

$$\sigma_v(u_1, u_2) = 0.1 \cdot \sigma_{\text{dist}}(u_1, u_2) \quad \text{and}$$
$$\sigma_e(u_1, v_1, u_2, v_2) = 0.45 \cdot \sigma_{\text{length}}(u_1, v_1, u_2, v_2) + 0.45 \cdot 1\sigma_{\text{angle}}(u_1, v_1, u_2, v_2), \quad (4.11)$$

where $\sigma_{\text{dist}}$ compares the relative 3D image locations of matched branchpoints, $\sigma_{\text{length}}$ compares the relative lengths of matched branches, and $\sigma_{\text{angle}}$ compares the running-directions of matched branches.

The distance affinity function $\sigma_{\text{dist}}$ is constructed as follows. Let $\rho(u)$ represent the estimated 3D image location of vertex $u$ and let $R$ and $t$ be the rotation matrix and translation vector for a rigid alignment between the image coordinate systems of $T_1$ and $T_2$. To match human airway trees, for example, $R$ and $t$ can be estimated by least-squares alignment of
the end vertices of the trachea and the left and right main bronchi as in [17]. Aligned vertex locations are compared via the affinity function $\sigma_{\text{dist}}$, which is defined as a saturated linear function via

$$
\sigma_{\text{dist}}(u_1, u_2) = \begin{cases} 
-1, & \text{if } \|\rho(u_1) - R(\rho(u_2) - t)\| > t_d \\
1 - \frac{2\|\rho(u_1) - R(\rho(u_2) - t)\|}{t_d}, & \text{otherwise}
\end{cases}
$$

(4.12)

where $t_d$ is a parameter. Because $R$ and $t$ are only approximately known, $t_d$ must be large relative to the expected distance between two matched branchpoints. We set $t_d = 75$ mm. Thus, (4.12) only weakly favors nearby branchpoints and is relatively insensitive to estimation errors in the rigid alignment parameters $R$ and $t$.

The edge length affinity function $\sigma_{\text{length}}$ is constructed as follows. Let $\lambda(u, v)$ represent the length of an edge $(u, v) \in E(T)$. Define the length ratio between two branches $(u_1, v_1)$ and $(u_2, v_2)$ to be

$$
L(u_1, v_1, u_2, v_2) = \max \left( \frac{\lambda(u_1, v_1)}{\lambda(u_2, v_2)}, \frac{\lambda(u_2, v_2)}{\lambda(u_1, v_1)} \right),
$$

(4.13)

and define the length measure to be

$$
\sigma_{\text{length}}(u_1, v_1, u_2, v_2) = \begin{cases} 
-1, & \text{if } L(u_1, v_1, u_2, v_2) > t_r \\
-\frac{2L(u_1, v_1, u_2, v_2)}{t_r - 1} + \frac{t_r + 1}{t_r - 1}, & \text{otherwise,}
\end{cases}
$$

(4.14)

where $t_r$ is a threshold. We use $t_r = 1.5$ in practice. Recall that an edge $(u_1, v_1)$ in the common tree may correspond to a path consisting of several edges in $T_1$ or $T_2$. In this case, the total branch length is defined in the obvious way by summing the lengths of its constituent edges.
Finally, $\sigma_{\text{angle}}$ is constructed as follows. Let

$$\cos_\angle(a, b) = \frac{a^T b}{\|a\|\|b\|}$$  \hfill (4.15)

be the cosine of the angle between two vectors. If $(u_1, v_1)$ and $(u_2, v_2)$ are two matched edges in the common tree, define the cosine of the angle between their aligned running directions to be

$$\cos_\angle(u_1, v_1, u_2, v_2) = \cos_\angle\left(\rho(v_1) - \rho(u_1), R(\rho(v_2) - \rho(u_2))\right).$$  \hfill (4.16)

The angle measure is given by

$$\sigma_{\text{angle}}(u_1, v_1, u_2, v_2) = \begin{cases} 
-1, & \text{if } \cos_\angle(u_1, v_1, u_2, v_2) < t_a \\
\frac{2\cos_\angle(u_1, v_1, u_2, v_2)}{1-t_a} + \frac{t_a+1}{t_a-1}, & \text{otherwise,}
\end{cases}$$  \hfill (4.17)

where $t_a$ is a threshold. Thus, $\sigma_{\text{angle}}$ favors matched branches that run nearly parallel in $T_1$ and $T_2$. We use $t_a = 0.5$ in practice.

### 4.2.3 Theory

We have now characterized the set of valid matches between two anatomical trees using the secondary deformation model and defined a similarity measure for evaluating valid matches. We must still produce an algorithm for efficiently locating an optimal valid match. This section provides the theoretical framework for such an algorithm, which will be developed in Section 4.2.4.

The development proceeds in three parts. Section 4.2.3.1 demonstrates that any valid
match under the secondary deformation model can be generated by concatenating a collection of simple "primitive" matches. Each primitive describes the connection between a vertex and its children in the common tree. The simple three- and four-vertex matches in Figures 4.4, 4.5, 4.6, and 4.7 are all examples of primitives. Section 4.2.3.2 shows that the similarity measure (4.10) can be computed for any valid match by summing a local contribution from each of its constituent primitives. Such measures are referred to as "locally computable." Finally, Section 4.2.3.3 derives a recursive definition for an optimal match between two trees. The recursion works for any set of valid matches generated from primitives and for any locally computable measure and therefore works for our problem. This recursive definition will be exploited in Section 4.2.4 to produce an efficient DP algorithm for anatomical-tree matching.

Most of the development in this section is purely technical and can be skipped in a casual reading. Only Definition 4.6, which formally defines the set of primitives used to construct valid matches under the secondary deformation model, and Corollary 4.11 which provides the recursion used by our DP algorithm, will be referenced outside of this section.

4.2.3.1 Local Deformation Models

This section derives a key property of $\Phi_S(T_1, T_2)$, the set of valid matches under the secondary deformation model. Specifically, it is demonstrated that each valid match $\phi \in \Phi_S(T_1, T_2)$ can be uniquely decomposed into a collection of simple primitives describing the connection between a matched vertex and its children in the common tree. Fig. 4.10 illustrates the decomposition of the match from Fig. 4.9. Conversely, any valid match can be constructed by concatenating primitives. These properties will be important in Section
which breaks the difficult problem of locating an optimal match into a number of simple subproblems of locating an optimal primitive.

Figure 4.10. The unique primitive decomposition of the match illustrated in Fig. 4.9. Each primitive matches of only a few vertices and describes the connection between a vertex and its children in the common tree.

To proceed, we must clarify what we mean when we say a match $\phi \in \Phi_S(T_1, T_2)$ is constructed from primitives.

**Definition 4.4** Consider trees $T_1, T_2 \in T$ and a pair of vertices $(u_1, u_2) \in V(T_1) \times V(T_2)$.

A function $\psi$ is a **primitive** between $T_1^{u_1}$ and $T_2^{u_2}$ if:

1. $\psi$ is one-to-one and maps $u_1$ to $u_2$

2. $\forall v_1, w_1 \in D(\psi) \setminus u_1$:

   (a) $v_1 \not\to w_1$ and $w_1 \not\to v_1$ in $T_1^{u_1}$

   (b) $\psi(v_1) \not\to \psi(w_1)$ and $\psi(w_1) \not\to \psi(v_1)$ in $T_2^{u_2}$
Intuitively, $\psi$ is a primitive if it defines a common tree with depth at most one. The matches illustrated in Figs. 4.4-4.8 all define a common tree of depth one and all satisfy Definition 4.4. Here, the notation $\psi$ is introduced to differentiate primitives, which typically match four or fewer vertices, from full matches $\phi$, which may match hundreds of vertices.

Definition 4.4 is broad and includes implausible matches such as the example in Fig. 4.8. To exclude nonsensical primitives, we define a special subset of valid primitives. The subset of valid primitives can change for each pair of vertices $u_1$ and $u_2$ depending, for example, on whether $u_1$ and $u_2$ are bifurcations, trifurcations, or leaf vertices in $T_1$ and $T_2$. The subset of valid primitives is therefore formally given by a set function $\Psi$, which we refer to as a local deformation model.

**Definition 4.5** Consider trees $T_1$ and $T_2$. A **local deformation model** is a set function $\Psi$ that returns, for any pair of potential matched vertices $(u_1, u_2) \in V(T_1) \times V(T_2)$, a subset of valid primitives between $T_{u_1}^{u_1}$ and $T_{u_2}^{u_2}$, denoted $\Psi(T_{u_1}^{u_1}, T_{u_2}^{u_2})$.

We illustrate Definition 4.5 by characterizing the valid primitives for the secondary deformation model.

**Definition 4.6** Consider two trees $T_1, T_2 \in \mathbb{T}$. For each pair of potential matched vertices $(u_1, u_2) \in V(T_1) \times V(T_2)$, define the set $\Psi_S(T_1^{u_1}, T_2^{u_2})$ with $\psi \in \Psi_S(T_1^{u_1}, T_2^{u_2})$ if:

1. $\psi$ is one-to-one and maps $u_1$ to $u_2$

2. $\forall v_1, w_1 \in D(\psi) \setminus u_1$:

   (a) $v_1 \not\sim w_1$ and $w_1 \not\sim v_1$ in $T_{u_1}^{u_1}$

   (b) $\psi(v_1) \not\sim \psi(w_1)$ and $\psi(w_1) \not\sim \psi(v_1)$ in $T_{u_2}^{u_2}$
3. \(|SN(T_1^{u_1}, u_1, D(\psi))| \leq 2\) and \(|SN(T_2^{u_2}, u_2, R(\psi))| \leq 2\)

4. \(\forall v_1 \in D(\psi) \setminus u_1, |E(u_1 P v_1)| \leq 4\) in \(T_1^{u_1}\) and \(|E(u_2 P \psi(v_1))| \leq 4\) in \(T_2^{u_2}\)

It is easy to verify that \(\Psi_S\) is indeed a local deformation model; i.e., that every element of the set \(\Psi_S(T_1^{u_1}, T_2^{u_2})\) is a primitive for arbitrary trees \(T_1^{u_1}\) and \(T_2^{u_2}\). We need only note that items 1 and 2 match the requirements of Definition 4.4. The remaining items are restrictions that forbid the primitive in Fig. 4.8, while allowing the primitives Figs. 4.4, 4.5, 4.6, and 4.7.

Given a local deformation model such as \(\Psi_S\), it is possible to determine whether a match \(\phi\) is constructed from valid primitives by performing a decomposition such as the one in Fig. 4.10 and independently verifying the validity of each of its constituent primitives.

**Definition 4.7** Consider two trees \(T_1, T_2 \in \mathbb{T}\) and a local deformation model \(\Psi\). The set of valid matches between \(T_1\) and \(T_2\) that is generated by \(\Psi\) is denoted \(\Phi(\Psi)(T_1, T_2)\), with \(\phi \in \Phi(\Psi)(T_1, T_2)\) if:

1. \(\phi\) matches \(r_1\) to \(r_2\)

2. \(\forall u_1 \in D(\phi)\) the function \(\psi_{u_1}^{\phi}\), defined by restricting the domain of \(\phi\) to \(u_1\) and its children in \(T_1[D(\phi)]\), is a valid primitive, denoted \(\psi_{\phi}^{u_1} \in \Psi(T_1^{u_1}, T_2^{\phi(u_1)})\).

In Fig. 4.10 for example,

\[
\psi_{\phi}^{a_1} = \{(a_1, a_2), (b_1, b_2), (k_1, m_2)\} \quad \text{and} \quad \psi_{\phi}^{b_1} = \{(b_1, b_2), (d_1, c_2), (g_1, g_2), (h_1, h_2)\}. \quad (4.18)
\]

Both satisfy Definition 4.6 and are therefore valid primitives under the secondary deformation model. In fact, all the primitives of the match \(\phi\) in Fig. 4.10 satisfy Definition 4.6.
meaning $\phi \in \Phi_{\Psi}(T_1, T_2)$. It is straightforward to demonstrate that this is true for any valid match under the secondary deformation model; i.e., that any valid $\phi \in \Phi_{\Psi}(T_1, T_2)$ is generated by concatenating primitives such as the ones in Figs. 4.4-4.7. We will first, however, require two technical results.

**Proposition 4.1** Consider $T \in \mathbb{T}$ and $u \in V(T)$. Let $T^u$ represent the bottom-up subtree of $T$ rooted at $u$ as defined in (4.5). Then:

1. For any $v, w \in V(T^u)$, $v \sim w$ in $T \iff v \sim w$ in $T^u$. Furthermore, the graph $vPw$ has the same vertex and edge sets in $T$ and $T^u$.

2. For any $v \in V(T^u)$, $(T^u)^v = T^v$

*Proof.* Follows directly from the definition of a bottom-up subtree in (4.5). □

In Figure 4.1, for example, we have $(T^u)^f = T^f$; i.e., the bottom-up subtree of $T^a$ rooted at $f$ is the same as the bottom-up subtree at $T$ rooted at $f$. The results of Proposition 4.1 will be used to prove many results in this section, including the following.

**Lemma 4.1** Consider two trees $T_1, T_2 \in \mathbb{T}$ and a function $\phi$ that maps $r_1$ to $r_2$ and is an isomorphism between $T_1[D(\phi)]$ and $T_2[R(\phi)]$. For any $u_1 \in D(\phi)$, define $\psi^{u_1}_\phi$ as in item 2 of Definition 4.7. Then

\[
SN(T_1, u_1, D(\phi)) = SN(T_1^{u_1}, u_1, D(\psi^{u_1}_\phi)) \text{ and } (4.19)
\]

\[
SN(T_2, \phi(u_1), R(\phi)) = SN(T_2^{\phi(u_1)}, \phi(u_1), R(\psi^{u_1}_\phi)). (4.20)
\]
Proof. Consider (4.19). By Definition 4.2,

\[
SN(T_1, u_1, D(\phi)) = \{u_1\} \cup \{x_1 \in V(T_1) | x_1 \in V(u_1 P v_1) \cap V(u_1 P w_1) \text{ in } T_1 \text{ for some } v_1, w_1 \in C_{T_1[D(\phi)]}^{u_1}\}
\]

and

\[
SN(T_1^{u_1}, u_1, D(\psi_{\phi}^{u_1})) = \{u_1\} \cup \{x_1 \in V(T_1^{u_1}) | x_1 \in V(u_1 P v_1) \cap V(u_1 P w_1) \text{ in } T_1^{u_1} \text{ for some } v_1, w_1 \in C_{T_1^{u_1}[D(\psi_{\phi}^{u_1})]}^{u_1}\}.
\]

Now, \(V(u_1 P v_1)\) and \(V(u_1 P w_1)\) are the same in \(T_1\) and \(T_1^{u_1}\) by Proposition 4.1. It therefore suffices to show \(C_{T_1[D(\phi)]}^{u_1} \subseteq C_{T_1^{u_1}[D(\psi_{\phi}^{u_1})]}^{u_1}\) to demonstrate (4.19); i.e., it is necessary to show that \(u_1\) has the same children in \(T_1[D(\phi)]\) and \(T_1^{u_1}[D(\psi_{\phi}^{u_1})]\). We begin by demonstrating \(C_{T_1[D(\phi)]}^{u_1} \subseteq C_{T_1^{u_1}[D(\psi_{\phi}^{u_1})]}^{u_1}\). Consider the following.

\[
[v_1 \text{ is a child of } u_1 \text{ in } T_1[D(\phi)]] \iff [u_1 \rightsquigarrow v_1 \text{ in } T_1 \text{ and } u_1 \rightsquigarrow z_1 \rightsquigarrow v_1 \text{ in } T_1 \Rightarrow z_1 \notin D(\phi)] \iff (4.21)
\]

\[
[u_1 \rightsquigarrow v_1 \text{ in } T_1^{u_1} \text{ and } u_1 \rightsquigarrow z_1 \rightsquigarrow v_1 \text{ in } T_1^{u_1} \Rightarrow z_1 \notin D(\psi_{\phi}^{u_1})] \Rightarrow (4.22)
\]

\[
[u_1 \rightsquigarrow v_1 \text{ in } T_1^{u_1} \text{ and } u_1 \rightsquigarrow z_1 \rightsquigarrow v_1 \text{ in } T_1^{u_1} \Rightarrow z_1 \notin D(\psi_{\phi}^{u_1})] \iff (4.23)
\]

\[
[v_1 \text{ is a child of } u_1 \text{ in } T_1^{u_1}[D(\psi_{\phi}^{u_1})]] (4.24)
\]

Here, (4.21) and (4.24) follow immediately from item 2 of Theorem 4.1. The implication in (4.22) follows from Proposition 4.1 and the implication in (4.23) follows because \(D(\psi_{\phi}^{u_1}) \subseteq D(\phi)\). Taken together, (4.21)-(4.24) demonstrate \(C_{T_1[D(\phi)]}^{u_1} \subseteq C_{T_1^{u_1}[D(\psi_{\phi}^{u_1})]}^{u_1}\).
Proving the reverse inclusion is more difficult. Note, however, that if $A$ and $B$ are finite sets with $A \subseteq B$ and $|B| \leq |A|$, then $A = B$. It will therefore suffice to show $|C_{T_{1}^{u_{1}}[D(\psi_{\phi}^{u_{1}})]}^{u_{1}}| \leq |C_{T_{1}[D(\phi)]}^{u_{1}}|$. Note that, by Theorem 4.1, $T_{1}^{u_{1}}[D(\psi_{\phi}^{u_{1}})]$ is a tree rooted at $u_{1}$ with $|D(\psi_{\phi}^{u_{1}})|$ vertices. Thus, $u_{1}$ can have at most $|D(\psi_{\phi}^{u_{1}})| - 1$ children in $T_{1}^{u_{1}}[D(\psi_{\phi}^{u_{1}})]$. Furthermore, as $D(\psi_{\phi}^{u_{1}})$ is defined to consist of $u_{1}$ and its children in $T_{1}[D(\phi)]$, we have $|D(\psi_{\phi}^{u_{1}})| = |C_{T_{1}^{u_{1}}[D(\phi)]}^{u_{1}}| + 1$. Thus, $|C_{T_{1}^{u_{1}}[D(\psi_{\phi}^{u_{1}})]}^{u_{1}}| \leq |C_{T_{1}^{u_{1}}[D(\phi)]}^{u_{1}}|$ as desired.

We have now shown $u_{1}$ has the same children in $T_{1}[D(\phi)]$ and $T_{1}^{u_{1}}[D(\psi_{\phi}^{u_{1}})]$. Thus, (4.19) holds. A similar line of reasoning can be followed to demonstrate (4.20). □

With this, we are ready for the main result of the section.

**Theorem 4.2** Consider two trees $T_{1}, T_{2} \in \mathbb{T}$. The set of valid matches $\Phi_{S}(T_{1}, T_{2})$ is generated by the local deformation model $\Psi_{S}$. In the notation of Definition 4.7

$$\Phi_{S}(T_{1}, T_{2}) = \Phi_{\Psi_{S}}(T_{1}, T_{2}).$$

**Proof.** We must show $\phi \in \Phi_{S}(T_{1}, T_{2}) \iff \phi \in \Phi_{\Psi_{S}}(T_{1}, T_{2})$.

($\Rightarrow$) Assume $\phi \in \Phi_{S}(T_{1}, T_{2})$; i.e., that $\phi$ satisfies Definition 4.3. We must show Definition 4.7 to hold with $\Psi = \Psi_{S}$. By Definition 4.3, any $\phi \in \Phi_{S}(T_{1}, T_{2})$ must map $r_{1}$ to $r_{2}$, so item 1 of Definition 4.7 is trivially satisfied. We turn to item 2. Choose an arbitrary matched vertex $u_{1} \in D(\phi)$. We must show $\psi_{\phi}^{u_{1}} \in \Psi_{S}(T_{1}^{u_{1}}, T_{2}^{\phi(u_{1})})$. We demonstrate the requirements of Definition 4.6 item-by-item.

1. By Definition 4.3, $\phi$ must define an isomorphism between $T_{1}[D(\phi)]$ and $T_{2}[R(\phi)]$. It must therefore be one-to-one. As $\psi_{\phi}^{u_{1}}$ is defined to be a restriction of $\phi$, it must be
one-to-one as well. Furthermore, it must agree with $\phi$ on all elements of its domain.
Specifically, it must map $u_1$ to $\phi(u_1)$.

2. Choose $v_1, w_1 \in D(\psi_{u_1}^{u_1}) \setminus u_1$ arbitrary.

   (a) Suppose, to elicit a contradiction, that $v_1 \rightsquigarrow w_1$ in $T_1^{u_1}$, hence in $T_1$. Then,
   $u_1 \rightsquigarrow v_1 \rightsquigarrow w_1$ in $T_1$ and $(u_1, w_1)$ cannot be an edge of $T_1[D(\phi)]$ by item 2 of
   Theorem 4.1. This implies $w_1 \notin D(\psi_{u_1}^{u_1}) \setminus u_1$, a contradiction. A symmetric
   argument applies shows $w_1 \not\rightsquigarrow v_1$ in $T_1^{u_1}$.

   (b) Because $\phi$ is an isomorphism between $T_1[D(\phi)]$ and $T_2[R(\phi)]$ and both $(u_1, v_1)$
   and $(u_1, w_1)$ are edges in $T_1[D(\phi)]$ by the definition of $\psi_{u_1}^{u_1}$, we know that both
   $(\phi(u_1), \phi(v_1))$ and $(\phi(u_1), \phi(w_1))$ are edges in $T_2[R(\phi)]$ as well. Now, if we assume
   $\psi_{\phi}^{u_1}(v_1) \rightsquigarrow \psi_{\phi}^{u_1}(w_1)$ in $T_2^{\phi(u_1)}$, we must have $\phi(v_1) \rightsquigarrow \phi(w_1)$ in $T_2^{\phi(u_1)}$, hence
   in $T_2$, because $\psi_{\phi}^{u_1}$ and $\phi$ must coincide on $D(\psi_{\phi}^{u_1})$. This, however, implies
   $(\phi(u_1), \phi(w_1))$ is not an edge in $T_2[R(\phi)]$, a contradiction. Thus $\psi_{\phi}^{u_1}(v_1) \not\rightsquigarrow
   \psi_{\phi}^{u_1}(w_1)$. A symmetric argument yields $\psi_{\phi}^{u_1}(w_1) \not\rightsquigarrow \psi_{\phi}^{u_1}(v_1)$.

3. Apply Lemma 4.1. If $|SN(T_1, u_1, D(\phi))| \leq 2$, then $|SN(T_1^{u_1}, u_1, D(\psi_{\phi}^{u_1}))| \leq 2$ must
   hold as well, for the two supernodes are the same. The same argument applies for
   $SN(T_2^{\phi(u_1)}, \phi(u_1), R(\psi_{\phi}^{u_1}))$.

4. Trivial.

   \((\iff)\) Assume $\phi \in \Phi_{\Psi,S}(T_1, T_2)$; i.e., that $\phi$ satisfies Definition 4.7 with $\Psi = \Psi_S$. We
   must show Definition 4.3. We proceed item-by-item.

   1. Definition 4.7 has no explicit requirement that $\phi$ be an isomorphism between $T_1[D(\phi)]$
and $T_2[R(\phi)]$. Theorem F.1 in Appendix F however, gives a general result that if $\Psi$ is any local deformation model and $\phi \in \Phi_\Psi(T_1, T_2)$ is any valid match generated by $\Psi$ then $\phi$ is indeed an isomorphism between $T_1[D(\phi)]$ and $T_2[R(\phi)]$. This result holds here, because $\Psi_S$ is a local deformation model and $\phi \in \Phi_{\Psi_S}(T_1, T_2)$ by hypothesis.

2. Choose $u_1 \in D(\phi)$ arbitrary. Because $\psi^{u_1}_\phi \in \Psi_S(T_1^{u_1}, T_2^{\phi(u_1)})$, we have

$$|SN(T_1^{u_1}, u_1, D(\psi^{u_1}_\phi))| \leq 2.$$ 

As $SN(T_1, u_1, D(\phi)) = SN(T_1^{u_1}, u_1, D(\psi^{u_1}_\phi))$ by Lemma 4.1, we must have

$$|SN(T_1, u_1, D(\phi))| \leq 2$$

as well. The same argument applies for $SN(T_2, \phi(u_1), R(\phi))$.

3. Trivial.

4.2.3.2 Locally-Computable Similarity Measures

Now that we have shown that each valid match $\phi \in \Phi_S(T_1, T_2)$ can be decomposed into a collection of valid primitives, we show that the primitive decomposition of $\phi$ can be used to simplify the computation of $S(T_1, T_2|\phi)$. Specifically, we show that $S(T_1, T_2|\phi)$ can be computed by summing an independent contribution from each primitive of $\phi$. This notion of additive decomposition is formalized in the following definition.

**Definition 4.8** Consider two trees $T_1, T_2 \in \mathbb{T}$, a local deformation model $\Psi$, and a similar-
ity measure $S(T_1, T_2|\phi)$. The measure $S$ is **locally computable** if there exists a function $S_L(T^{u_1}, T^{u_2}_2|\psi)$ defined for any $(u_1, u_2) \in V(T_1) \times V(T_2)$ and any primitive $\psi \in \Psi(T^{u_1}, T^{u_2}_2)$ such that

$$S(T_1, T_2|\phi) = \sum_{u_1 \in D(\phi)} S_L(T^{u_1}, T^{\phi(u_1)}_2|\psi_{\phi(u_1)})$$

holds for any $\phi \in \Phi_\Psi(T_1, T_2)$.

The following proposition demonstrates that the measure defined in (4.10) is locally computable.

**Proposition 4.2** The measure in (4.10) is locally computable for any $\sigma_v$ and $\sigma_e$ with

$$S_L(T^{u_1}_1, T^{u_2}_2|\psi) = \sigma_v(u_1, \psi(u_1)) + \sum_{v_1 \in D(\psi) \setminus u_1} \sigma_e(u_1, v_1, \psi(u_1), \psi(v_1)).$$

*Proof.* Consider the following chain of equations.

$$S(T_1, T_2|\phi) = \sum_{u_1 \in V(T_1[D(\phi)])} \sigma_v(u_1, \phi(u_1)) + \sum_{(u_1, v_1) \in E(T_1[D(\phi)])} \sigma_e(u_1, v_1, \phi(u_1), \phi(v_1))$$

$$= \sum_{u_1 \in D(\phi)} \sigma_v(u_1, \phi(u_1)) + \sum_{u_1 \in D(\phi)} \left[ \sum_{v_1 \in D(\psi_{\phi(u_1)}) \setminus u_1} \sigma_e(u_1, v_1, \phi(u_1), \phi(v_1)) \right]$$

$$= \sum_{u_1 \in D(\phi)} \left[ \sigma_v(u_1, \phi(u_1)) + \sum_{v_1 \in D(\psi_{\phi(u_1)}) \setminus u_1} \sigma_e(u_1, v_1, \phi(u_1), \phi(v_1)) \right]$$

$$= \sum_{u_1 \in D(\phi)} \left[ \sigma_v(u_1, \psi_{\phi(u_1)}(u_1)) + \sum_{v_1 \in D(\psi_{\phi(u_1)}) \setminus u_1} \sigma_e(u_1, v_1, \psi_{\phi(u_1)}(u_1), \psi_{\phi(u_1)}(v_1)) \right]$$

$$= \sum_{u_1 \in D(\phi)} S_L(T^{u_1}_1, T^{\phi(u_1)}_2|\psi_{\phi(u_1)})$$

(4.31)
Equation (4.27) is the definition of our measure. Equation (4.28) follows because \( V(T_1[D(\phi)]) \) is equal to \( D(\phi) \) by Theorem 4.1 and \((u_1, v_1) \in E(T_1[D(\phi)]) \Leftrightarrow v_1 \in D(\psi_{\phi}^{u_1}) \setminus u_1 \) by item 2 of Definition 4.7 which defines the domain of \( \psi_{\phi}^{u_1} \) to consist of \( u_1 \) and its children in \( T_1[D(\phi)] \).

Equation (4.30) holds because \( \psi_{\phi}^{u_1} \) is defined by restricting the domain of \( \phi \) and therefore \( \psi_{\phi}^{u_1} \) and \( \phi \) must return the same value for all \( v_1 \in D(\psi_{\phi}^{u_1}) \). Finally, (4.31) substitutes the definition of \( S_L \) from (4.26). Equation (4.31) completes the proof by demonstrating the requirement of Definition 4.8.  

4.2.3.3 Recursive Definition of an Optimal Match

We have now shown that \( \Phi_S(T_1, T_2) \) is generated by the local deformation model \( \Psi_S \) and that the similarity measure (4.10) is locally computable for any choice of \( \sigma_v \) and \( \sigma_e \). In this section, we show that the search for an optimal match between two input trees can be reduced to a easier search for an optimal primitive, provided optimal matches are known between all subtrees of the input trees. Theorem 4.3 provides the general result, which is specialized for our purposes in Corollary 4.1.

The following result will be useful throughout this section. It describes a commutative relationship between the bottom-up and induced subtrees of a tree.

**Proposition 4.3** Consider \( T \in \mathbb{T} \) and \( K \subseteq V(T) \) with \( r \in K \). For any \( u \in K \), let \( T^u[V(T^u) \cap K] \) be the tree obtained by deleting all vertices in \( V(T^u) \setminus (V(T^u) \cap K) \) from \( T^u \) and \( (T[K])^u \) be the bottom-up subtree of the induced subtree \( T[K] \) rooted at \( u \). Then \( T^u[V(T^u) \cap K] = (T[K])^u \).

**Proof.** It is necessary to demonstrate both \( V(T^u[V(T^u) \cap K]) = V((T[K])^u) \) and
Consider the following chain of equations.

\[
E(T^u[V(T^u) \cap K]) = E((T[K])^u).
\]

\[
V(T^u[V(T^u) \cap K]) = V(T^u) \cap K \tag{4.32}
\]

\[
= V(T^u) \cap V(T[K]) \tag{4.33}
\]

\[
= \{(u \cup \{v \in V(T) : u \leadsto v \text{ in } T\}) \cap V(T[K]) \tag{4.34}
\]

\[
= \{u \cup \{v \in V(T[K]) : u \leadsto v \text{ in } T\} \tag{4.35}
\]

\[
= \{u \cup \{v \in V(T[K]) : u \leadsto v \text{ in } T[K]\} \tag{4.36}
\]

\[
= V((T[K])^u) \tag{4.37}
\]

Here (4.32) is item 2 of Theorem 4.1 applied to \( T^u[V(T^u) \cap K] \). Equation (4.33) is item 2 of Theorem 4.1 applied to \( T[K] \). Equation (4.34) is the definition of a bottom-up subtree. Equation (4.35) follows because \( u \in K = V(T[K]) \) by hypothesis and (4.36) follows from item 3 of Theorem 4.1. Finally, (4.37) is the definition of a bottom-up subtree.

We now turn our attention to the edge sets.

\[
E(T^u[V(T^u) \cap K]) = \{(x, y) \mid x, y \in V(T^u) \cap K, \ x \leadsto y \text{ in } T^u, \text{ and}
\]

\[
x \leadsto z \leadsto y \text{ in } T^u \Rightarrow z \notin V(T^u) \cap K\} \tag{4.38}
\]

\[
= \{(x, y) \mid x, y \in V(T^u) \cap K, \ x \leadsto y \text{ in } T, \text{ and}
\]

\[
x \leadsto z \leadsto y \text{ in } T \Rightarrow z \notin V(T^u) \cap K\} \tag{4.39}
\]

\[
= \{(x, y) \mid x, y \in V(T^u) \cap K, \ x \leadsto y \text{ in } T, \text{ and}
\]

\[
x \leadsto z \leadsto y \text{ in } T \Rightarrow z \notin K\} \tag{4.40}
\]

\[
= \{(x, y) \mid x \in V(T^u) \cap K, \ x, y \in K, \ x \leadsto y \text{ in } T, \text{ and}
\]

\[
x \leadsto z \leadsto y \text{ in } T \Rightarrow z \notin K\} \tag{4.41}
\]
Here, (4.38) is item 2 of Theorem 4.1 applied to $T^u[V(T^u) \cap K]$. Equation (4.39) follows from item 1 of Proposition 4.1. For (4.40), note that $z \in V(T^u)$, hence $z \not\in V(T^u) \cap K \Leftrightarrow z \not\in K$. For (4.41), note that $x \preceq y \in T$ implies $y \in V(T^u) \Leftrightarrow x \in V(T^u)$. Equation (4.42) is item 2 of Theorem 4.1 applied to $T[K]$. Equation (4.43) follows from the first part of this proof, where it was demonstrated that $V(T^u) \cap K = V((T[K])^u)$ and (4.44) is again the definition of a bottom-up subtree.

The proof of Theorem 4.3 will use a standard “optimal substructure” argument [158]. Specifically, we will argue that if $\phi$ is an optimal match between $T_1$ and $T_2$ and $(u_1, \phi(u_1))$ is any pair of matched vertices, then $\phi$ must also provide an optimal match between the bottom-up subtrees $T^u_1$ and $T^\phi_{u_1}$. Define the restriction of $\phi$ to a bottom-up subtree as follows.

**Definition 4.9** Consider two trees $T_1, T_2 \in \mathbb{T}$ and a function $\phi$ with $D(\phi) \subseteq V(T_1)$ and $R(\phi) \subseteq V(T_2)$. For any $u_1 \in D(\phi)$, define $\phi^{u_1}$ to be the function $\phi$ restricted to $D(\phi) \cap V(T^u_1)$.

Proposition 4.3 provides a simple, but useful, technical fact about $\phi^{u_1}$.

**Proposition 4.4** Consider two trees $T_1, T_2 \in \mathbb{T}$ and a function $\phi$ with $D(\phi) \subseteq V(T_1)$ and $R(\phi) \subseteq V(T_2)$. Then, for any $u_1 \in D(\phi)$, $(T_1[D(\phi)])^{u_1} = T^{u_1}_1[D(\phi^{u_1})]$.

**Proof.** As $D(\phi^{u_1}) = D(\phi) \cap V(T^{u_1}_1)$ by definition, apply Proposition 4.3 with $T = T_1$.
and \( K = D(\phi) \). \( \square \)

A useful property of matches generated by a local deformation model is that if \( \phi \) is a valid match between \( T_1 \) and \( T_2 \), then \( \phi^{u_1} \) is also a valid match between \( T_1^{u_1} \) and \( T_2^{\phi(u_1)} \). Lemma 4.2 will prove this and other related results. We first need a simple proposition.

**Proposition 4.5** Consider a local deformation model \( \Psi \), two trees \( T_1, T_2 \in \mathbb{T} \), and a valid match \( \phi \in \Phi_\Psi(T_1, T_2) \). Let \( v_1 \) be an arbitrary vertex in \( D(\phi) \setminus r_1 \). Then \( v_1 \in D(\phi^{u_1}) \) for exactly one \( u_1 \in D(\psi^{r_1}) \setminus r_1 \).

**Proof.** Note that \([u_1 \in D(\psi^{r_1}) \setminus r_1] \Leftrightarrow [u_1 \text{ is a child of } r_1 \text{ in } T_1[D(\phi)]]\) by Definition 4.7. Furthermore, by Definition 4.9 we have \( D(\phi^{u_1}) = V(T_1^{u_1}) \cap D(\phi) \). In the proof of Proposition 4.3, we saw that \( V(T_1^{u_1}) \cap D(\phi) = V((T_1[D(\phi)])^{u_1}) \). Thus, \( D(\phi^{u_1}) = V((T_1[D(\phi)])^{u_1}) \).

The statement of this proposition is therefore equivalent to, “Given a tree \( T \in \mathbb{T} \) and a non-root vertex \( v \), then \( v \) belongs to \( V(T^u) \) for exactly one \( u \in C_T \).” This is a basic property of hierarchical trees. \( \square \)

**Lemma 4.2** Consider a local deformation model \( \Psi \), two trees \( T_1, T_2 \in \mathbb{T} \), a valid match \( \phi \in \Phi_\Psi(T_1, T_2) \), and a locally computable similarity measure \( S \). Then:

1. \( \forall u_1 \in D(\phi) \text{ and } \forall v_1 \in D(\phi^{u_1}), \psi^{v_1}_{\phi^{u_1}} = \psi^{v_1}_\phi \)

2. \( \forall u_1 \in D(\phi), \phi^{u_1} \in \Phi_\Psi(T_1^{u_1}, T_2^{\phi(u_1)}) \)

3. \( S(T_1, T_2|\phi) = S_L(T_1, T_2|\psi^{r_1}_\phi) + \sum_{u_1 \in D(\psi^{r_1}) \setminus \{r_1\}} S(T_1^{u_1}, T_2^{\phi(u_1)}|\phi^{u_1}) \)

**Proof.**
1. Both \( \psi_{\phi u_1}^{v_1} \) and \( \psi_{\phi}^{v_1} \) are defined to be restrictions of \( \phi \) (note that \( \psi_{\phi u_1}^{v_1} \) is a restriction of \( \phi u_1 \), itself a restriction of \( \phi \)). It is therefore sufficient to demonstrate \( D(\psi_{\phi u_1}^{v_1}) = D(\psi_{\phi}^{v_1}) \). The domain of \( \psi_{\phi u_1}^{v_1} \) consists of \( u_1 \) and its children in \( T_1[D(\phi)] \), while the domain of \( \psi_{\phi u_1}^{v_1} \) consists of \( u_1 \) and its children in \( T_u u_1[D(\phi u_1)] \). By Proposition 4.4,

\[
T_u u_1[D(\phi u_1)] = (T_1[D(\phi)]) u_1. \tag{4.45}
\]

Now, \( u_1 \) has the same children in \( T_1[D(\phi)] \) and \( (T_1[D(\phi)]) u_1 \). This, combined with (4.45), implies that \( u_1 \) has the same children in both \( T_1[D(\phi)] \) and \( T_u u_1[D(\phi u_1)] \). Thus,

\[
D(\psi_{\phi u_1}^{v_1}) = D(\psi_{\phi}^{v_1}) \quad \text{and} \quad \psi_{\phi u_1}^{v_1} = \psi_{\phi}^{v_1}.
\]

2. The restricted function \( \phi u_1 \) clearly matches the roots of \( T_u u_1 \) and \( T_2^{\phi(u_1)} \). We must still demonstrate that item 2 in Definition 4.7 holds; i.e., that

\[
\psi_{\phi u_1}^{v_1} \in \Psi((T_u u_1)^{v_1}, (T_2^{\phi(u_1)})^{\phi(u_1)(v_1)}) \quad \forall v_1 \in D(\phi u_1). \tag{4.46}
\]

Fix an arbitrary \( v_1 \in D(\phi u_1) \). We first note that \( \psi_{\phi u_1}^{v_1} = \psi_{\phi}^{v_1} \) by item 1. Furthermore, \( v_1 \in D(\phi u_1) \Rightarrow v_1 \in T_u u_1 \), hence \( (T_u u_1)^{v_1} = T_u u_1 \) by Proposition 4.1. Finally, \( \phi \in \Phi(\phi, \phi) \) by hypothesis. Thus \( u_1 \sim v_1 \) in \( T_1 \Leftrightarrow \phi(u_1) \sim \phi(v_1) \) in \( T_2 \) by Lemma E.1. As \( \phi(u_1) = \phi(v_1) \), we have \( \phi(u_1) \sim \phi(u_1)(v_1) \) in \( T_2 \). Another application of Proposition 4.1 yields \( (T_2^{\phi(u_1)})^{\phi(u_1)(v_1)} = T_2^{\phi(v_1)} \). Thus, the requirement in (4.46) is equivalent to

\[
\psi_{\phi}^{v_1} \in \Psi(T_u u_1, T_2^{\phi(v_1)}) \quad \forall v_1 \in D(\phi u_1). \tag{4.47}
\]

Now, \( \phi \in \Phi(\phi, \phi) \) by hypothesis. Hence, \( \psi_{\phi}^{v_1} \in \Psi(T_u u_1, T_2^{\phi(v_1)}) \) holds for all \( v_1 \in \).
$D(\phi)$. As $D(\phi) \supseteq D(\phi^{u_1})$, this completes the proof.

3. Consider the following chain of equations:

$$S(T_1, T_2|\phi) = \sum_{x_1 \in D(\phi)} S_L(T_1^{x_1}, T_2^{\phi(x_1)}|\psi_\phi^{x_1})$$  \hspace{1cm} (4.48)$$

$$= S_L(T_1, T_2|\psi_\phi^{r_1}) + \sum_{x_1 \in D(\phi) \setminus r_1} S_L(T_1^{x_1}, T_2^{\phi(x_1)}|\psi_\phi^{x_1})$$  \hspace{1cm} (4.49)$$

$$= S_L(T_1, T_2|\psi_\phi^{r_1}) + \sum_{u_1 \in D(\psi_\phi^{r_1}) \setminus r_1} \sum_{v_1 \in D(\phi^{u_1})} S_L(T_1^{u_1}, T_2^{\phi(v_1)}|\psi_\phi^{v_1})$$  \hspace{1cm} (4.50)$$

$$= S_L(T_1, T_2|\psi_\phi^{r_1}) + \sum_{u_1 \in D(\psi_\phi^{r_1}) \setminus r_1} \sum_{v_1 \in D(\phi^{u_1})} S_L((T_1^{u_1})^{v_1}, (T_2^{\phi(u_1)})^{\phi(v_1)}|\psi_\phi^{v_1})$$  \hspace{1cm} (4.51)$$

$$= S_L(T_1, T_2|\psi_\phi^{r_1}) + \sum_{u_1 \in D(\psi_\phi^{r_1}) \setminus \{r_1\}} S(T_1^{u_1}, T_2^{\phi(u_1)}|\phi^{u_1}).$$  \hspace{1cm} (4.52)$$

Equation (4.48) is simply the definition of a locally computable measure. Equation (4.49) holds because $\phi$ maps $r_1$ to $r_2$, $T_1^{r_1} = T_1$, and $T_2^{r_2} = T_2$. Equation (4.50) follows immediately from Proposition 4.5. Equation (4.51) repeats the logic from the proof of item 2. Finally, equation (4.52) is again just the definition of a locally computable measure applied once for each $u_1 \in D(\psi_\phi^{r_1})$.

Lemma 4.2 demonstrated that we could construct a valid match between bottom-up subtrees of $T_1$ and $T_2$ by “cutting” apart a valid match between $T_1$ and $T_2$. The following lemma provides the reverse functionality and shows how to construct a valid match between $T_1$ and $T_2$ by “pasting” together valid matches between its subtrees.
Lemma 4.3 Consider a local deformation model \( \Psi \), two trees \( T_1, T_2 \in T \), and a valid primitive \( \psi \in \Psi(T_1, T_2) \) with \( D(\psi) = \{ r_1, u_1^{(1)}, u_1^{(2)}, ..., u_1^{(M)} \} \). Suppose for each \( i \in \{1, 2, ..., M\} \) there exists a valid match \( \phi^{(i)} \in \Phi_\psi(T_1^{u_1^{(i)}}, T_2^{\psi(u_1^{(i)})}) \), and define \[
\phi \triangleq \psi \cup \left[ \bigcup_{i=1}^{M} \phi^{(i)} \right], \tag{4.53}
\]
where the union of functions is defined by treating the functions as sets of ordered pairs.

Then:

1. \( \phi \) is a function; i.e., it is single-valued
2. \( \phi^{u_1^{(i)}} = \phi^{(i)} \) for \( i \in \{1, 2, ..., M\} \)
3. \( \psi_{\phi}^{T_1} = \psi \)
4. \( \phi \in \Phi_\psi(T_1, T_2) \)

Proof.

1. As we are defining \( \phi \) to be the union of a set of ordered pairs, we must make sure it is single-valued. It suffices to show

\[ D(\psi) \cap D(\phi^{(i)}) \subseteq \{ u_1^{(i)} \} \ \forall i \quad \text{and} \quad D(\phi^{(i)}) \cap D(\phi^{(j)}) = \emptyset \ \forall i \neq j. \tag{4.54} \]

If (4.54) holds, the only possible confusion occurs on the \( u_1^{(i)} \). Of course, as \( \phi^{(i)} \in \Phi_\psi(T_1^{u_1^{(i)}}, T_2^{\psi(u_1^{(i)})}) \) by hypothesis, it must match \( u_1^{(i)} \) to \( \psi(u_1^{(i)}) \). We note that \( D(\phi^{(i)}) \subseteq V(T_1^{u_1^{(i)}}) \forall i \) and proceed demonstrate the stronger claims
\[ D(\psi) \cap V(T_{11}^{u_1}) \subseteq \{u_1^{(i)}\} \quad \forall i \quad \text{and} \quad V(T_{11}^{u_1}) \cap D(\phi^{(j)}) = \emptyset \quad \forall i \neq j. \quad (4.55) \]

(a) \( D(\psi) \cap V(T_{11}^{u_1}) \subseteq \{u_1^{(i)}\} \forall i : \) Clearly \( r_1 \notin V(T_{11}^{u_1}) \) as \( r_1 \rightsquigarrow u_1^{(i)} \Rightarrow u_1^{(i)} \not\rightsquigarrow r_1 \) in \( T_1 \) by antisymmetry. Now, suppose there is some vertex \( v_1 \in (D(\psi) \setminus r_1) \cap V(T_{11}^{u_1}) \) such that \( v_1 \neq u_1^{(i)} \). We then have \( u_1^{(i)} \rightsquigarrow v_1 \) in \( T_1 \). This, however, contradicts the hypothesis that \( \psi \) is a primitive, as \( u_1, v_1 \in D(\psi) \setminus r_1 \Rightarrow u_1 \not\rightsquigarrow v_1 \) in \( T_1 \) by item 2 of Definition 4.4.

(b) \( V(T_{11}^{u_1}) \cap D(\phi^{(j)}) = \emptyset \forall i \neq j : \) Clearly \( u_1^{(i)} \notin D(\phi^{(j)}) \), as this would imply \( D(\psi) \cap D(\phi^{(j)}) = \{u_1^{(i)}, u_1^{(j)}\} \), which we have just shown to be impossible. Suppose there is some other vertex \( x_1 \in V(T_{11}^{u_1}) \cap D(\phi^{(j)}) \). Then \( r_1 \rightsquigarrow u_1^{(j)} \rightsquigarrow x_1 \) and \( r_1 \rightsquigarrow u_1^{(i)} \rightsquigarrow x_1 \) in \( T_1 \). By path uniqueness, this implies either \( u_1^{(j)} \rightsquigarrow u_1^{(i)} \) or \( u_1^{(j)} \rightsquigarrow u_1^{(i)} \) in \( T_1 \), both of which are impossible as \( \psi \) is a primitive.

2. Choose \( i \) arbitrary. It suffices to show that \( D(\phi^{u_1^{(i)}}) = D(\phi^{(i)}) \).

\[
\begin{align*}
D(\phi^{u_1^{(i)}}) &= V(T_{11}^{u_1^{(i)}}) \cap D(\phi) \\
&= V(T_{11}^{u_1^{(i)}}) \cap [D(\psi) \cup \bigcup_{j=1}^{M} D(\phi^{(j)})] \\
&= [V(T_{11}^{u_1^{(i)}}) \cap D(\psi)] \cup \bigcup_{j=1}^{M} [V(T_{11}^{u_1^{(i)}}) \cap D(\phi^{(j)})] \\
&= \{u_1\} \cup [V(T_{11}^{u_1^{(i)}}) \cap D(\phi^{(i)})] \cup \bigcup_{i \neq j}^{M} [V(T_{11}^{u_1^{(i)}}) \cap D(\phi^{(j)})] \\
&= \{u_1\} \cup D(\phi^{(i)}) \cup \bigcup_{i \neq j}^{M} \emptyset \\
&= D(\phi^{(i)})
\end{align*}
\]
3. It suffices to show $D(\psi_{\phi}^{r_1}) = D(\psi)$. As $D(\psi_{\phi}^{r_1})$ consists of $r_1$ and its children in $T_1[D(\phi)]$, we must show that $(r_1, v_1) \in E(T_1[D(\phi)]) \iff v_1 = u_1^{(i)}$ for some $i$.

$(\Rightarrow)$ Suppose $(r_1, v_1) \in E(T_1[D(\phi)])$. Assume, to elicit a contradiction, that $v_1$ is not one of the $u_1^{(i)}$. As $v_1 \in V(T_1[D(\phi)]) = D(\phi)$, it must therefore be an element of $D(\phi^{(i)})$ for some $i$. Thus, there exists some $u_1^{(i)}$ satisfying $r_1 \sim u_1^{(i)} \sim v_1$ in $T_1$ and $u_1^{(i)} \in D(\phi)$. By item 2 of Theorem 4.1, this means $(r_1, v_1) \notin E(T_1[D(\phi)])$, a contradiction. Thus, $v_1$ is one of the $u_1^{(i)}$.

$(\Leftarrow)$ Choose $i$ arbitrary. Clearly $r_1 \sim u_1^{(i)}$, as $r_1$ is the root of $T_1$. Suppose $\exists w_1 \in V(T_1)$ such that $r_1 \sim w_1 \sim u_1^{(i)}$ in $T_1$. Now, $\psi$ is a valid primitive by hypothesis, so $u_1^{(j)} \not\sim u_1^{(i)}$ for $i \neq j$ by Definition 4.5. Thus, $w_1$ cannot be $u_1^{(j)}$ for $j \neq i$. Similarly, $w_1 \notin D(\phi_{u_1^{(j)}})$ for $j \neq i$ as $w_1 \in D(\phi_{u_1^{(j)}})$ implies $u_1^{(j)} \sim w_1 \sim u_1^{(i)}$, which is again impossible. Finally, $w_1 \notin D(\phi_{u_1^{(i)}})$ as $w_1 \in D(\phi_{u_1^{(i)}})$ implies $u_1^{(i)} \not\sim u_1^{(i)}$ by antisymmetry. Thus, $r_1 \not\sim w_1 \sim u_1^{(i)}$ in $T_1 \Rightarrow w_1 \notin D(\phi)$, hence $(r_1, u_1^{(i)}) \in E(T_1[D(\phi)])$.

4. Clearly, $\phi$ matches the roots of $T_1$ and $T_2$. We must show that $\forall v_1 \in D(\phi)$ the function $\psi_{\phi}^{r_1}$ is a primitive. By item 3, $\psi_{\phi}^{r_1} = \psi$, which is valid by hypothesis. Consider an arbitrary vertex $v_1 \in D(\phi) \setminus r_1$. By hypothesis, $v_1$ must belong to $D(\phi^{(i)})$ for at least one $i$ ($r_1$ is the only vertex for which this does not hold). This, combined with (4.54), implies that $v_1$ lies in $D(\phi^{(i)})$ for exactly one $i$. As each of the $\phi^{(i)}$ are valid matches, we know $\psi_{\phi^{(i)}}^{r_1} \in \Phi_{\psi}(T_1^{\phi^{(i)}}, T_2^{\phi^{(i)}})$ holds (here, we have implicitly used Proposition 4.1 and the fact that $\phi^{(i)}(v_1) = \phi(v_1)$). It therefore suffices to show that $\psi_{\phi}^{r_1} = \psi_{\phi^{(i)}}^{r_1}$. Moreover, we again need only demonstrate $D(\psi_{\phi}^{r_1}) = D(\psi_{\phi^{(i)}}^{r_1})$. Now, $D(\psi_{\phi}^{r_1})$ consists
of $v_1$ and its children in $T_1[D(\phi)]$, while $D(\psi^{v_1}\phi^{(i)})$ consists of $v_1$ and its children in $T_1^{v_1}[\phi^{(i)}]$. By item 2, $\phi^{(i)} = \phi^{u_1^{(i)}}$. Hence, $T_1^{v_1}[\phi^{(i)}] = T_1^{u_1^{(i)}}[\phi^{u_1^{(i)}}] = (T_1[D(\phi)])^{u_1^{(i)}}$ by Proposition 4.4. We need only note that the child sets of $v_1$ in $T_1[D(\phi)]$ and $(T_1[D(\phi)])^{u_1^{(i)}}$ are identical.

\[ \square \]

**Theorem 4.3** Consider a local deformation model $\Psi$ and two trees $T_1, T_2 \in T$. Let $S$ be a locally computable similarity measure. Then:

\[
S^*[T_1, T_2] = \max_{\psi \in \Psi(T_1, T_2)} \left\{ S_L(T_1, T_2|\psi) + \sum_{v_1 \in D(\psi) \setminus r_1} S^*[T_{v_1}^{v_1}, T_2^{\psi(\psi)}] \right\} \tag{4.62}
\]

and is achieved by

\[
\phi \triangleq \psi^* \cup \left[ \bigcup_{v_1 \in D(\psi^*) \setminus r_1} \phi^*[T_{v_1}^{v_1}, T_2^{\psi^*(\psi)}] \right], \tag{4.63}
\]

where $\psi^*$ is given by

\[
\psi^* \triangleq \arg \max_{\psi \in \Psi(T_1, T_2)} \left\{ S_L(T_1, T_2|\psi) + \sum_{v_1 \in D(\psi) \setminus r_1} S^*[T_{v_1}^{v_1}, T_2^{\psi^*(\psi)}] \right\} \tag{4.64}
\]

**Proof.** Lemmas 4.2 and 4.3 state that the $\phi$ defined in (4.63) is a valid match between $T_1$ and $T_2$ and that:

\[
S(T_1, T_2|\phi) = S_L(T_1, T_2|\psi^*) + \sum_{v_1 \in D(\psi^*) \setminus r_1} S^*[T_{v_1}^{v_1}, T_2^{\psi^*(\psi)}] \tag{4.65}
\]
Thus, the similarity given by the right hand side of (4.62) is achievable. We must show
that this is the maximum achievable similarity. Suppose, to elicit a contradiction, that this
is not true; i.e., that there exists a match $\tilde{\phi} \in \Phi_{S}(T_{1}^{u_{1}}, T_{2}^{u_{2}})$ such that $S(T_{1}^{u_{1}}, T_{2}^{u_{2}}|\tilde{\phi}) > S(T_{1}^{u_{1}}, T_{2}^{u_{2}}|\phi)$. One of two cases must therefore hold. Either $\psi_{\tilde{\phi}}^{r_{1}} = \psi^{*}$ or $\psi_{\tilde{\phi}}^{r_{1}} \neq \psi^{*}$. Both
cases lead to contradictions.

**Case 1:** ($\psi_{\tilde{\phi}}^{r_{1}} = \psi^{*}$) Expanding according to Lemma 4.2, subtracting, and canceling the
common $S_{L}(T_{1}, T_{2} | \psi^{*})$ term yields:

$$S(T_{1}, T_{2} | \tilde{\phi}) - S(T_{1}, T_{2} | \phi) = \sum_{v_{1} \in D(\psi^{*}) \backslash r_{1}} \left[ S(T_{1}^{v_{1}}, T_{2}^{\psi^{*} \tilde{\phi}}(v_{1}) | \tilde{\phi}(v_{1})) - S^{*}[T_{1}^{v_{1}}, T_{2}^{\psi^{*} \tilde{\phi}}(v_{1})] \right] \leq 0$$

where the inequality holds because $S^{*}[T_{1}^{v_{1}}, T_{2}^{\psi^{*} \tilde{\phi}}(v_{1})]$ is the maximum achievable similar-
ity between $T_{1}^{v_{1}}$ and $T_{2}^{\psi^{*} \tilde{\phi}}(v_{1})$ by definition. Thus, both $S(T_{1}, T_{2} | \tilde{\phi}) > S(T_{1}, T_{2} | \phi)$ and
$S(T_{1}, T_{2} | \tilde{\phi}) \leq S(T_{1}, T_{2} | \phi)$ must hold, a contradiction.

**Case 2:** ($\psi_{\tilde{\phi}}^{r_{1}} \neq \psi^{*}$) Construct the function $\phi'$ as follows:

$$\phi' \triangleq \psi_{\tilde{\phi}}^{r_{1}} \cup \left[ \bigcup_{v_{1} \in D(\psi_{\tilde{\phi}}^{r_{1}}) \backslash r_{1}} \phi^{*}[T_{1}^{v_{1}}, T_{2}^{\psi_{\tilde{\phi}}^{r_{1}}(v_{1})}] \right]$$

(4.66)

Again, $\phi'$ is a valid match by Lemma 4.3. Repeating the argument from Case 1 yields
$S(T_{1}, T_{2} | \phi') \geq S(T_{1}, T_{2} | \tilde{\phi})$. By assumption, $S(T_{1}, T_{2} | \tilde{\phi}) > S(T_{1}, T_{2} | \phi)$. Therefore,
$S(T_{1}, T_{2} | \phi') > S(T_{1}, T_{2} | \phi)$ holds. Thus,

$$S_{L}(T_{1}, T_{2} | \psi_{\tilde{\phi}}^{r_{1}}) + \sum_{v_{1} \in D(\psi_{\tilde{\phi}}^{r_{1}}) \backslash r_{1}} S^{*}[T_{1}^{v_{1}}, T_{2}^{\psi_{\tilde{\phi}}^{r_{1}}(v_{1})}] = S(T_{1}, T_{2} | \phi') >$$
\[
S(T_1, T_2 | \phi) = S_L(T_1, T_2 | \psi^*) + \sum_{v_1 \in D(\psi^*) \setminus v_1} S^*[T_1^{v_1}, T_2^{\psi^*(v_1)}]. \tag{4.67}
\]

This implies that \(\psi^*\) does not satisfy (4.64), contradicting the hypothesis of the theorem.

Now, because \(\Phi_S(T_1, T_2)\) is generated by the local deformation model \(\Psi_S\) and the similarity measure \(S\) is a locally computable measure, Theorem 4.3 can be used to simplify the tree matching problem. The following corollary specializes Theorem 4.3 for the secondary deformation model and the measure defined in Section 4.2.2.

**Corollary 4.1** Consider two input trees \(T_1, T_2 \in \mathbb{T}\). For any \((u_1, u_2) \in V(T_1) \times V(T_2)\), define \(S^*[T_1^{u_1}, T_2^{u_2}]\) to be the maximum achievable similarity between \(T_1^{u_1}\) and \(T_2^{u_2}\) under the secondary deformation model and the similarity measure in (4.10); i.e.,

\[
S^*[T_1^{u_1}, T_2^{u_2}] = \max_{\phi \in \Phi_S(T_1^{u_1}, T_2^{u_2})} \left\{ \sum_{u_1 \in V(T_1[D(\phi)])} \sigma_v(u_1, \phi(u_1)) + \sum_{(u_1, v_1) \in E(T_1[D(\phi)])} \sigma_e(u_1, v_1, \phi(u_1), \phi(v_1)) \right\}. \tag{4.68}
\]

Then

\[
S^*[T_1^{u_1}, T_2^{u_2}] = \sigma_v(u_1, u_2) + \max_{\psi \in \Psi_S(T_1^{u_1}, T_2^{u_2})} \left\{ \sum_{v_1 \in D(\psi) \setminus u_1} \left( \sigma_e(u_1, v_1, u_2, \psi(v_1)) + S^*[T_1^{v_1}, T_2^{\psi(v_1)}] \right) \right\}. \tag{4.69}
\]

Furthermore, (4.69) reduces to

\[
S^*[T_1^{u_1}, T_2^{u_2}] = \sigma_v(u_1, u_2) \tag{4.70}
\]
when either \( u_1 \) or \( u_2 \) is a leaf vertex in \( T_1 \) or \( T_2 \).

Proof. In Section 4.2.3.1, it was shown that \( \Phi_S(T_1, T_2) \) is generated by the local deformation model \( \Psi_S \) and in Section 4.2.3.2, it was shown that the measure (4.10) is locally computable. Therefore, Theorem 4.3 can be directly applied to the problem of maximizing (4.10) over valid matches \( \phi \in \Phi_S(T_1, T_2) \).

Consider the following chain of equations.

\[
S^*[T^{u_1}_1, T^{u_2}_2] = \max_{\phi \in \Phi_S(T^{u_1}_1, T^{u_2}_2)} \{ S(T^{u_1}_1, T^{u_2}_2 | \phi) \} \quad (4.71)
\]

\[
= \max_{\psi \in \Psi_S(T^{u_1}_1, T^{u_2}_2)} \left\{ S_L(T^{u_1}_1, T^{u_2}_2 | \psi) + \sum_{v_1 \in D(\psi) \setminus u_1} S^*[T^{v_1}_1, T_1^{\psi(v_1)}] \right\} \quad (4.72)
\]

\[
= \max_{\psi \in \Psi_S(T^{u_1}_1, T^{u_2}_2)} \left\{ S_L(T^{u_1}_1, T^{u_2}_2 | \psi) + \sum_{v_1 \in D(\psi) \setminus u_1} S^*[T_1^{v_1} T_1^{\psi(v_1)}] \right\} \quad (4.73)
\]

\[
= \max_{\psi \in \Psi_S(T^{u_1}_1, T^{u_2}_2)} \left\{ \sigma_v(u_1, u_2) + \sum_{v_1 \in D(\psi) \setminus u_1} \sigma_e(u_1, v_1, u_2, \psi(v_1)) \right. \]
\[
+ \sum_{v_1 \in D(\psi) \setminus u_1} S^*[T^{v_1}_1, T_1^{\psi(v_1)}] \right\} \quad (4.74)
\]

Here (4.72) is a direct application of Theorem 4.3. Equation (4.73) follows from Proposition 4.11 as \((T^{u_1}_1)^{v_1} = T^{v_1}_1 \) and \((T^{u_2}_1)^{\psi(v_1)} = T_1^{\psi(v_1)} \). Equation (4.74) follows by directly substituting the definition of \( S_L \) from (4.26). Finally, the desired result in (4.69) can be obtained by rearranging terms in (4.74) and noting that the \( \sigma_v(u_1, u_2) \) term is independent of \( \psi \).

To demonstrate (4.70), it suffices to note that if \( u_1 \) is a leaf in \( T_1 \), then \( T_1^{u_1} \) consists of only a single vertex. Hence, the only possible match between \( T_1^{u_1} \) and \( T_2^{u_2} \) is the trivial match \( \{(u_1, u_2)\} \). The similarity of this match is \( \sigma_v(u_1, u_2) \). A symmetric argument holds when \( u_2 \) is a leaf in \( T_2 \). \( \square \)
4.2.4 Dynamic Programming Algorithm

This section describes a bottom-up dynamic programming algorithm for the anatomical-tree matching problem. The development proceeds in three parts. Section 4.2.4.1 describes the main body of the algorithm. The main body of the algorithm repeatedly calls a subroutine for determining the optimal primitive between two bottom-up subtrees. Section 4.2.4.2 describes the subroutine. Finally, Section 4.2.4.3 analyzes the time and space requirements of both the subroutine and the algorithm as a whole.

4.2.4.1 Main Body of the DP Algorithm

The overall goal for the DP algorithm is to locate an optimal valid match under the secondary deformation model described in Definition 4.3 that maximizes the similarity measure $S$ defined in (4.10). The algorithm relies on Corollary 4.1 which reduces the search for an optimal match between two trees to a related search for an optimal primitive, provided that optimal matches between all bottom-up subtrees of the trees are already known. This simplifies the problem greatly as primitives, such as the examples in Figs. 4.4-4.7, typically match four or fewer vertices, while the full match returned by our algorithm may match hundreds of vertices. To ensure that Corollary 4.1 applies, the algorithm visits the vertices of $T_1$ and $T_2$ in a bottom-up order, starting with the leaves of the trees and ending with the roots.

Algorithm 4.1 gives the formal procedure. The table $S^*[T_{u1}^{u1}, T_{u2}^{u2}]$ in Algorithm 4.1 stores maximum achievable similarity between each pair of bottom-up subtrees $T_{u1}^{u1}$ and $T_{u2}^{u2}$, while $\psi^*[T_{u1}^{u1}, T_{u2}^{u2}]$ stores the optimal primitive connecting $u_1$ and $u_2$ to their children.
in the common tree. Per Corollary 4.1, the values of $S^*$ and $\psi^*$ are

$$S^*[T_{u_1}^{u_1}, T_{u_2}^{u_2}] = \sigma_e(u_1, u_2)$$

$$\psi^*[T_{u_1}^{u_1}, T_{u_2}^{u_2}] = \{(u_1, u_2)\}, \quad (4.75)$$

when either $u_1$ or $u_2$ is a leaf vertex, and

$$S^*[T_{u_1}^{u_1}, T_{u_2}^{u_2}] = \sigma_e(u_1, u_2) + \max_{\psi \in \Psi_S(T_{u_1}^{v_1}, T_{u_2}^{v_2})} \left\{ \sum_{v_1 \in D(\psi) \setminus u_1} (\sigma_e(u_1, v_1, u_2, \psi(v_1)) + S^*[T_{v_1}^{v_1}, T_{u_2}^{v_2}]) \right\}$$

$$\psi^*[T_{u_1}^{u_1}, T_{u_2}^{u_2}] = \arg\max_{\psi \in \Psi_S(T_{u_1}^{v_1}, T_{u_2}^{v_2})} \left\{ \sum_{v_1 \in D(\psi) \setminus u_1} (\sigma_e(u_1, v_1, u_2, \psi(v_1)) + S^*[T_{v_1}^{v_1}, T_{u_2}^{v_2}]) \right\}, \quad (4.76)$$

when both $u_1$ and $u_2$ are non-leaf vertices.

Lines 2 - 4 handle the base case that occurs when $u_1$ or $u_2$ is a leaf. Lines 6 - 14 loop through vertices in bottom-up order and update $S^*[T_{u_1}^{u_1}, T_{u_2}^{u_2}]$ according to (4.76). The optimization in (4.76) is performed by a subroutine, **LocateOptimalPrimitive**, which is described in the next section. The maximum achievable similarity between the full input trees $T_1$ and $T_2$ is computed in the final loop iteration. Finally, lines 16 - 19 reconstruct the optimal match from the stored primitives.

Fig. 4.11 illustrates the basic idea of the algorithm’s main loop. Here, the algorithm attempts to locate an optimal primitive between the bottom-up subtrees $T_{u_1}^{u_1}$ and $T_{u_2}^{u_2}$. The optimal primitive must match $u_1$ to $u_2$ and determine the children of $u_1$ and $u_2$ in the common tree. Now, because the subtrees are visited in bottom-up order, the maximum achievable similarity between each pair of subtrees below $u_1$ and $u_2$ is already known. Specifically, it is known that there is no good match between $T_{u_1}^{u_1}$ and any subtree of $T_{u_2}^{u_2}$, as
\(v_1\) belongs to a spurious branch. More importantly, it is known that the pairs \(T_1^{u_1}, T_2^{u_2}\) and \(T_1^{y_1}, T_2^{w_2}\) can both be matched with high similarity. In this example, the optimal primitive is

\[
\psi^* [T_1^{u_1}, T_2^{u_2}] = \{(u_1, u_2), (x_1, v_2), (y_1, w_2)\},
\]

which achieves similarity

\[
S^*[T_1^{u_1}, T_2^{u_2}] = \sigma_v(u_1, u_2) + \sigma_e(u_1, x_1, u_2, v_2) + \sigma_e(u_1, y_1, u_2, w_2) +
S^*[T_1^{x_1}, T_2^{v_2}] + S^*[T_1^{y_1}, T_2^{w_2}].
\]

The optimal match between \(T_1^{u_1}\) and \(T_2^{u_2}\) is given by the union of \(\psi^*[T_1^{u_1}, T_2^{u_2}]\) and the previously-computed optimal matches between the subtree pairs \(T_1^{x_1}, T_2^{v_2}\) and \(T_1^{y_1}, T_2^{w_2}\).

\[\text{Algorithm 4.1 } \phi^* = \text{LocateOptimalMatch} (T_1, T_2)\]

1: // Handle the base case when either \(u_1\) or \(u_2\) is a leaf vertex.
2: for all \((u_1, u_2)\) such that \(|V(T_1^{u_1})| = 1 \) or \(|V(T_2^{u_2})| = 1\) do
3: \(S^*[u_1, u_2] \leftarrow \sigma_{bp}(u_1, u_2)\) and \(\psi^*[u_1, u_2] \leftarrow \{(u_1, u_2)\}\)
4: end for
5: // Cycle through each pair of vertices \((u_1, u_2)\) in a depth-first order.
6: for \(i = 2\) to \(\text{depth}(T_1)\) do
7: \(\text{for } j = 2\) to \(\text{depth}(T_2)\) do
8: \(\text{for all } (u_1, u_2)\) such that \(\text{depth}(V(T_1^{u_1})) = i\) and \(\text{depth}(V(T_2^{u_2})) = j\) do
9: \(\text{// The subroutine in line 11 performs the optimization in (4.76) and}
10: \(\text{// is defined in Algorithm 4.2.}\)
11: \((S^*[T_1^{u_1}, T_2^{u_2}], \psi^*[T_1^{u_1}, T_2^{u_2}]) \leftarrow \text{LocateOptimalPrimitive}(T_1^{u_1}, T_2^{u_2})\)
12: end for
13: end for
14: end for
15: // Reconstruct the optimal match from the stored primitives.
16: \(\phi^* \leftarrow \{(r_1, r_2)\}\)
17: while \(\exists u_1 \in D(\phi^*)\) such that \(\psi^*[u_1, \phi^*(u_1)] \not\subseteq D(\phi^*)\) do
18: \(\phi^* \leftarrow \phi^* \cup \psi^*[u_1, \phi^*(u_1)]\)
19: end while
20: return \(\phi^*\)
Figure 4.11. An example illustrating the main loop of the proposed dynamic programming algorithm. (a) Subtrees are visited in bottom-up order. By the time we attempt to match $T_{u1}^{u1}$ and $T_{u2}^{u2}$, we therefore already know the maximum-similarity match between all pairs of subtrees below $u_1$ and $u_2$. (b) Knowledge of optimal matches between the subtrees of $T_{u1}^{u1}$ and $T_{u2}^{u2}$ makes it possible to avoid matching spurious branches, such as the $(v_1, w_1)$ branch in $T_1$.

4.2.4.2 Optimal Primitive Subroutine

This section describes the subroutine **LocateOptimalPrimitive** called in line 11 of Algorithm 4.1. The subroutine updates $S^*[T_{u1}^{u1}, T_{u2}^{u2}]$ and $\psi^*[T_{u1}^{u1}, T_{u2}^{u2}]$ per (4.76). Figure 4.12 gives an example used to illustrate the subroutine’s operation, which is fully detailed in Algorithm 4.2.

The subroutine begins by enumerating all possible supernodes $SN(T_{u1}^{u1}, u_1, D(\psi))$ and $SN(T_{u2}^{u2}, u_2, R(\psi))$ that can arise from a valid primitive $\psi \in \Psi_S(T_{u1}^{u1}, T_{u2}^{u2})$. From item 3 of Definition 4.6, it is apparent that each possible supernode can consist of at most two vertices. Thus, a possible supernode in $T_{u1}^{u1}$ consists of $u_1$ and at most one of its children.
The set of all possible supernodes in $T_{u_1}$ is therefore

$$PSN(T_{u_1}) = \left\{ \{u_1\}, \{u_1, c_1^{(1)}\}, \ldots \{u_1, c_1^{(\delta(u_1))}\} \right\},$$  \hspace{1cm} (4.79)$$

where $c_1^{(1)}, \ldots, c_1^{(\delta(u_1))}$ is an arbitrary ordering of the children of $u_1$ in $T_{u_1}$. In Figure 4.12(a), for example, $PSN(T_{u_1})$ consists of four elements as $\delta(u_1) = 3$. Define $PSN(T_{u_2})$ analogously.

**Algorithm 4.2** Subroutine `LocateOptimalPrimitive($T_{u_1}^{u_1}$, $T_{u_2}^{u_2}$)` determining the optimal primitive $\psi^*[T_{u_1}^{u_1}, T_{u_2}^{u_2}]$ and similarity $S^*[T_{u_1}^{u_1}, T_{u_2}^{u_2}]$ required in line 11 of Algorithm 4.1:

1: Construct $PSN(T_{u_1}^{u_1})$ and $PSN(T_{u_2}^{u_2})$ per (4.79)
2: // Set default values for $S^*$ and $\psi^*$ to correspond to the trivial match
3: $S^*[T_{u_1}^{u_1}, T_{u_2}^{u_2}] \leftarrow \sigma(u_1, u_2)$
4: $\psi^*[T_{u_1}^{u_1}, T_{u_2}^{u_2}] \leftarrow \{(u_1, u_2)\}$
5: // Loop through potential supernode pairs
6: for all $SN1 \in PSN(T_{u_1}^{u_1})$ do
7: for all $SN2 \in PSN(T_{u_2}^{u_2})$ do
8: Construct effective child sets $C_{eff}(SN1)$ and $C_{eff}(SN2)$ via (4.80)
9: $\delta_{eff}(SN1) \leftarrow |C_{eff}(SN1)|$ and $\delta_{eff}(SN2) \leftarrow |C_{eff}(SN2)|$
10: $N \leftarrow \delta_{eff}(SN1) + \delta_{eff}(SN1)$
11: Construct complete bipartite graph $K_{N,N}$ with edge weights $W_{ij}$ per (4.82)
12: // Run Hungarian algorithm
13: Let $M^*$ be a maximum-weight matching in $K_{N,N}$
14: Construct $\psi_{temp}$ from $M^*$ via (4.85)
15: Construct $S_{temp}$ from $M^*$ via (4.86)
16: // Test to see if similarity achieved with this pair of supernodes is best so far.
17: \textbf{if } $S_{temp} > S^*[T_{u_1}^{u_1}, T_{u_2}^{u_2}]$ \textbf{then}
18: \hspace{1cm} // If so, update optimal similarity and primitive.
19: $S^*[T_{u_1}^{u_1}, T_{u_2}^{u_2}] \leftarrow S_{temp}$
20: $\psi^*[T_{u_1}^{u_1}, T_{u_2}^{u_2}] \leftarrow \psi_{temp}$
21: \textbf{end if}
22: \textbf{end for}
23: \textbf{end for}
24: \textbf{return} $\psi^*[T_{u_1}^{u_1}, T_{u_2}^{u_2}]$ and $S^*[T_{u_1}^{u_1}, T_{u_2}^{u_2}]$

The subroutine next loops through each pair of possible supernodes $SN1 \in PSN(T_{u_1}^{u_1})$ and $SN2 \in PSN(T_{u_2}^{u_2})$, and determines the maximum achievable similarity for a primitive
that is consistent with the pair. The final primitive returned by the subroutine achieves
the maximum similarity over all pairs of possible supernodes.

To determine the maximum-similarity primitive for the pair $SN_1$ and $SN_2$, the subroutine first constructs “effective” child sets for each supernode. The effective child set of $SN_1$, for example, is

$$C_{eff}(SN_1) = \{ x_1 \mid x_1 \not\in SN_1 \text{ and } x_1 \in C_{T_1}^{y_1} \text{ for some } y_1 \in SN_1 \}. \quad (4.80)$$

The effective degree of a supernode is naturally defined by $\delta_{eff}(SN_1) = |C_{eff}(SN_1)|$. Define the effective child set and effective degree of $SN_2$ analogously. Figure 4.12(b) illustrates effective child sets in $T_1^{u_1}$ and $T_2^{u_2}$ with $SN_1 = \{ u_1, c_1^{(2)} \}$ and $SN_2 = \{ u_2, c_2^{(1)} \}$. In this case, $\delta_{eff}(SN_1) = 4$ and $\delta_{eff}(SN_2) = 3$. The effective children of $SN_1$ and $SN_2$ are assigned an arbitrary ordering and labeled $e_1^{(1)}, ..., e_1^{(4)}$ and $e_2^{(1)}, ..., e_2^{(3)}$.

Next, the subroutine constructs a weighted complete bipartite graph $K_{N,N}$, with

$$N = \delta_{eff}(SN_1) + \delta_{eff}(SN_2). \quad (4.81)$$

Figure 4.12(c) illustrates this graph for our example. Vertices $1, ..., \delta_{eff}(SN_1)$ on the left side of this graph each represent a bottom-up subtrees beneath an effective child of $SN_1$ and vertices $1, ..., \delta_{eff}(SN_2)$ on the right represent subtrees beneath the effective children of $SN_2$. The rest are “dummy” vertices. Edges incident upon at least one dummy vertex receive zero weight, while edges incident two non-dummy vertices are weighted according
\[ W_{ij} = \max \{ \sigma_{br}(u_1, v_1, u_2, v_2) + S^* [v_1, v_2] \mid (v_1, v_2) \in X_i \times Y_j \}, \quad (4.82) \]

where \( X_i \) and \( Y_j \) are the subsets of vertices in \( T_1^{e(i)} \) and \( T_2^{e(j)} \) satisfying item \(^4.6\) in Definition \(^4.6\). Specifically,

\[
X_i = \{ v_1 \in V(T_1^{e(i)}) \text{ such that } |E(u_1 P v_1)| \leq 4 \text{ in } T_1^{u_1} \} \quad \text{and} \quad (4.83)
\]

\[
Y_j = \{ v_2 \in V(T_2^{e(j)}) \text{ such that } |E(u_2 P v_2)| \leq 4 \text{ in } T_2^{u_2} \}. \quad (4.84)
\]

Next, the subroutine uses the Hungarian algorithm to locate a maximum-weight matching in \( K_{N,N} \) \(^{[167]}\). The returned matching \( M^* \) is a bijection between \( \{1, 2, ..., N\} \) and \( \{1, 2, ..., N\} \). The primitive associated with \( M^* \) is constructed via

\[
\psi_{\text{temp}} = \{(u_1, u_2)\} \cup \bigcup_{(i,j) \in M^*} \gamma_{ij} \quad (4.85)
\]

where \( \gamma_{ij} \) is the pair \( \{(v_1, v_2)\} \) maximizing \((4.82)\) if \( i \leq \delta_{\text{eff}}(SN_1) \) and \( j \leq \delta_{\text{eff}}(SN_2) \) and is otherwise empty. The similarity achieved by \( \psi_{\text{temp}} \) is given by

\[
S_{\text{temp}} = \sigma_{\psi}(u_1, u_2) + \sum_{(i,j) \in M^*} W_{ij}. \quad (4.86)
\]

If, for example, the maximum-weight bipartite matching in Figure \(^4.12\)(c) were given by

\[
M^* = \{(1, 2), (2, 4), (3, 1), (4, 5), (5, 6), (6, 7), (7, 3)\}, \quad (4.87)
\]

then the associated primitive \( \psi_{\text{temp}} \) would match \( u_1 \) to \( u_2 \), a vertex \( v_1 \in X_1 \subseteq V(T_1^{e(i)}) \) to a
vertex \( v_2 \in Y_2 \subseteq V(T_2^{(2)}) \), and a vertex \( w_1 \in X_3 \subseteq V(T_1^{(3)}) \) to a vertex \( w_2 \in Y_2 \subseteq V(T_2^{(2)}) \).

The associated similarity (ignoring zero weights) would be

\[
S_{\text{temp}} = \sigma_v(u_1, u_2) + W_{12} + W_{31} \\
= \sigma_v(u_1, u_2) + \sigma_e(u_1, v_1, u_2, v_2) + S^e[T_1^{u_1}, T_2^{v_2}] + \sigma_e(u_1, w_1, u_2, w_2) + S^e[T_1^{u_1}, T_2^{w_2}].
\]

(4.88)

**Figure 4.12.** An illustration of the reduction of \textbf{LocateOptimalPrimitive} to a maximum-weight matching in a complete bipartite graph. (a) Two example subtrees \( T_1^{u_1} \) and \( T_2^{u_2} \). Here, \( u_1 \) has three children and \( u_2 \) has two children. (b) Two possible supernodes highlighted in \( T_1^{u_1} \) and \( T_2^{u_2} \). The effective children of the supernodes are colored yellow. (c) The complete bipartite graph \( K_{7,7} \). Yellow vertices in the graph correspond to effective children of the illustrated supernodes. The rest are dummy vertices. Thick edges in the bipartite graph are weighted according to (4.82). The thin edges have zero weight.
4.2.4.3 Algorithm Analysis

The running time for Algorithm 4.1 is dominated the LocateOptimalPrimitive subroutine in line 11, which is called $O(|V(T_1)||V(T_2)|)$ times. The subroutine runs in

$$O(\delta(u_1)\delta(u_2)\delta(T_1)^4\delta(T_2)^4)$$

time, as demonstrated in Appendix G. Note that the running time of the subroutine increases rapidly with the degree of the input trees. Fortunately, however, branchpoints in anatomical trees typically have low degree. The vast majority represent bifurcations, and it is reasonable in many problems to assume all branchpoints have degree at most four. Thus LocateOptimalPrimitive generally requires only constant time for anatomical trees. The overall running time of Algorithm 4.1 is therefore $O(|V(T_1)||V(T_2)|)$. The results section details empirical running times, which are in good agreement with this analysis.

To analyze the space requirement, note that both $S^*$ and $\psi^*$ are stored in a table with $|V(T_1)||V(T_2)|$ entries. The optimal similarity $S^*[T_1^{u_1},T_2^{u_2}]$ is simply a number, but the optimal primitive $\psi^*[T_1^{u_1},T_2^{u_2}]$ is a function. Appendix G also demonstrates that $\psi[T_1^{u_1},T_2^{u_2}]$ matches at most $O(\min\{\delta(T_1),\delta(T_2)\})$ vertices. Thus, for anatomical trees with bounded degree, the space requirement for Algorithm 4.1 is also $O(|V(T_1)||V(T_2)|)$.

4.3 Results

This section presents results comparing automatically generated matches between pairs of human airway trees, pig airway trees, and mouse coronary arterial trees to hand-generated ground-truth matches. The section is organized as follows. Section 4.3.1 introduces a
computer-based tree-matching tool that implements the proposed matching algorithm and provides a means for obtaining ground-truth hand matches used to evaluate the automatic algorithm’s performance. Section 4.3.2 presents matching results obtained for pairs of human airway trees. Section 4.3.3 summarizes results obtained for other types of anatomical trees. Section 4.3.4 compares matches obtained using the proposed algorithm with those obtained using an existing method [17]. Finally, Section 4.3.5 analyzes the empirical running time of the proposed matching algorithm in comparison with the theoretical analysis performed in Section 4.2.4.3.

4.3.1 TreeMatch Tool and Standard for Comparison

The proposed automatic tree-matching algorithm has been implemented as part the TreeMatch Tool, a computerized system for visualizing and matching pairs of anatomical trees. The system is developed in C++ using OpenGL for visualization and is maintained in Microsoft Visual Studio 2003. The user loads two input trees into the system and views their medial-axis representations in side-by-side 3D renderings. Appendix II provides a complete user manual for the tree-matching system.

The TreeMatch tool also provides a means for validating the proposed automatic tree-matching algorithm by enabling the hand-matching of two input trees. Figure 4.13 illustrates the use of the TreeMatch Tool for this purpose. To ensure high-quality hand matches, the system enables simultaneous 3D rotation, panning, and zooming in both the left and right panes. Furthermore, the user can view the input trees superimposed on slices of the original 3D image data. This helps the user to resolve ambiguities caused by incomplete or incorrect tree extraction.
Once both automatic and hand matches have been obtained for a given pair of input trees, the accuracy of the automatically generated is measured as

\[
\text{accuracy} = \frac{2 \cdot \# \text{ of correct matches}}{\# \text{ of hand matches} + \# \text{ of automatic matches}}, \tag{4.89}
\]

where an automatically generated match is considered to be correct if agrees with one of the hand matches on both the source branchpoint in \( T_1 \) and the target branchpoint in \( T_2 \). The measure was chosen because it penalizes incorrect matches, missed matches, and extra false matches approximately equally. Thus, highly accurate automatic matches must identify both the correct number of matching branchpoints and the correct correspondences between the two input trees. The measure is also symmetric, as both an automatically generated match from \( T_1 \) to \( T_2 \) and its inverse from \( T_2 \) to \( T_1 \) will achieve the same accuracy.

### 4.3.2 Human Airway-Tree Matching

We first evaluate the accuracy of the proposed automatic algorithm in matching pairs of human airway trees extracted from MDCT chest scans of the same patient at both functional-residual and total lung capacities (FRC and TLC). Trees were automatically-extracted using the segmentation algorithm detailed in Chapter 3 of this thesis, the axial-analysis algorithm of Yu [54], and the airway quantification algorithm of Gibbs [18].

Table 4.1 summarizes results for six pairs of trees extracted from FRC and TLC scans of the same patient. The individual matches are illustrated in Figures 4.14-4.19. The overall accuracy achieved by the proposed algorithm is 96%, with nearly all errors involving leaf branches of the two trees. In Section 4.3.4, we show that these results represent a significant
Figure 4.13. Use of TreeMatch Tool to generate hand matches for evaluation of proposed automatic tree-matching algorithm. (a) A global tree view. The blue airways have already been matched and the green airways remain unmatched. The user is about to match the corresponding pair of yellow airways. (b) The user can choose to link or unlink the camera locations in the left and right panes and view the input trees superimposed on oblique slices through the original 3D image data. Incorporating the 3D image data helps the user make good matching decisions when the information provided by the extracted trees is incorrect or ambiguous.

improvement over the best existing method for human airway-tree matching.

Overall, the trees extracted from TLC scans have nearly twice as many branches as the trees extracted from FRC scans. This is to be expected, as fully-inflated airways typically have stronger image signatures than partially-inflated airways. Figure 4.20 gives an extreme
Table 4.1. Summary of matching results for six pairs of airway trees extracted from FRC and TLC scans of the same patient. The columns labeled “FRC size” and “TLC size” refer to the number of branches in the trees extracted from the FRC and TLC scans. Accuracy is computed via (4.89).

<table>
<thead>
<tr>
<th>Case</th>
<th>FRC size</th>
<th>TLC size</th>
<th>Num. Hand Matches</th>
<th>Num. Auto Matches</th>
<th>Num. Correct Matches</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>h002</td>
<td>136</td>
<td>230</td>
<td>126</td>
<td>127</td>
<td>122</td>
<td>96.4%</td>
</tr>
<tr>
<td>h008</td>
<td>114</td>
<td>162</td>
<td>108</td>
<td>108</td>
<td>105</td>
<td>97.2%</td>
</tr>
<tr>
<td>h018</td>
<td>70</td>
<td>136</td>
<td>67</td>
<td>67</td>
<td>63</td>
<td>94.0%</td>
</tr>
<tr>
<td>20349-3-28</td>
<td>116</td>
<td>575</td>
<td>110</td>
<td>111</td>
<td>101</td>
<td>91.4%</td>
</tr>
<tr>
<td>20349-3-36</td>
<td>304</td>
<td>408</td>
<td>276</td>
<td>279</td>
<td>270</td>
<td>97.3%</td>
</tr>
<tr>
<td>Emphysema</td>
<td>164</td>
<td>216</td>
<td>160</td>
<td>159</td>
<td>156</td>
<td>95.7%</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>904</strong></td>
<td><strong>1,727</strong></td>
<td><strong>847</strong></td>
<td><strong>851</strong></td>
<td><strong>817</strong></td>
<td><strong>96.2%</strong></td>
</tr>
</tbody>
</table>

example of this phenomenon for case 20349-3-28. Note that the TLC tree for this case contains nearly five times as many branches as the FRC tree. The extra branches visible in the TLC tree are colored yellow Figure 4.17.

The majority of the matches located by the proposed algorithm are well-handled using only the primary deformation model of Section 4.2.1.1, which assumes the two input trees are corrupted only by extra spurious branches. More complex secondary deformations, described in Section 4.2.1.2, occur in only 12 of the 851 total matches (1.4%) located by the proposed algorithm. Proper handling of secondary deformations is nonetheless vital for achieving highly-accurate automatic matches. Figure 4.21 provides one example of a secondary deformation occurring in case h002, in which the FRC and TLC trees disagree as to which of three well-matched subtrees branches off first. The correct match is valid under the secondary deformation model, but no valid match under the primary deformation model correctly matches all three subtrees.

We have also applied the proposed algorithm to the problem of matching two human airway trees extracted from the same MDCT chest scan using different tree-extraction methods. The automatically generated matches were compared with ground-truth hand
Figure 4.14. Automatic FRC/TLC match for case h002. Matched branches are depicted in blue for the first tree and black for the second. Unmatched branches are depicted in red and yellow. Corresponding branchpoints are connected by thin green lines.

Figure 4.15. Automatic FRC/TLC match for case h008. Matched branches are depicted in blue for the first tree and black for the second. Unmatched branches are depicted in red and yellow. Corresponding branchpoints are connected by thin green lines.
Figure 4.16. Automatic FRC/TLC match for case h018. Matched branches are depicted in blue for the first tree and black for the second. Unmatched branches are depicted in red and yellow. Corresponding branchpoints are connected by thin green lines.

Figure 4.17. Automatic FRC/TLC match for case 20349-3-28. Matched branches are depicted in blue for the first tree and black for the second. Unmatched branches are depicted in red and yellow. Corresponding branchpoints are connected by thin green lines.
**Figure 4.18.** Automatic FRC/TLC match for case 20349-3-36. Matched branches are depicted in blue for the first tree and black for the second. Unmatched branches are depicted in red and yellow. Corresponding branchpoints are connected by thin green lines.

**Figure 4.19.** Automatic FRC/TLC match for case emphysema. Matched branches are depicted in blue for the first tree and black for the second. Unmatched branches are depicted in red and yellow. Corresponding branchpoints are connected by thin green lines.
matches on two cases, 21405-3a and 20349-3-29. For both cases, the segmentation for $T_1$ was obtained using the automatic algorithm presented in Chapter 3 of this thesis and the segmentation for $T_2$ was obtained using an adaptive region-growing algorithm [1, 9, 11]. The same axial-analysis algorithm was used for both trees [54]. The accuracy of the automatically generated matches was 97.7% for case 21405-3a and 96.2% for case 20349-3-29. The automatically generated matches are illustrated in Figures 4.22-4.23. Here, the automatic match serves to highlight (in yellow) the extra branches obtained using the proposed automatic segmentation algorithm. The match also highlights (in red) several areas of leakage in the left lower lobe of case 20349-3-29.

Figure 4.20. Oblique MDCT cross-sections of corresponding airways as depicted in a functional residual capacity (a) and total lung capacity (b) scan. The same airways appear dramatically larger in the TLC scan, and the tree extracted from the TLC scan contains nearly five times as many branches as the one extracted from the FRC scan. The images were required using the TreeMatch tool introduced in Section 4.3.1. Pixels in each slice are isotropically sampled at a rate of 0.5mm and windowed to HU ∈ [−1000, −200]. The illustrated case is 20349-3-28.
Figure 4.21. (a) A close-up view of the right upper lobe from the FRC/TLC match for case h002. Note that in the FRC tree (blue), the subtree rooted at $x_1$ “branches off” before the subtrees rooted at $v_1$ and $w_1$, while in the TLC tree (black/yellow) the subtree rooted at $\phi(v_1)$ branches off first. (b) A simplified depiction of this situation. The highlighted branchpoints are treated as a trifurcation in the common tree.
Figure 4.22. Automatic match between two airway trees extracted from the same scan (case 21405-3a) using different methods. The first was segmented using the automatic algorithm described in Chapter 3 of this thesis. The second tree was segmented using an adaptive region-growing approach [1,9,11]. The same axial-analysis algorithm was used for both trees [54]. Matched branches are depicted in blue for the first tree and black for the second. Unmatched branches are depicted in red and yellow. Corresponding branchpoints are connected by thin green lines.

Figure 4.23. Automatic match between two airway trees extracted from the same scan (case 20349-3-29) using different methods. The first was segmented using the automatic algorithm described in Chapter 3 of this thesis. The second tree was segmented using an adaptive region-growing approach [1,9,11]. The same axial-analysis algorithm was used for both trees [54]. Matched branches are depicted in blue for the first tree and black for the second. Unmatched branches are depicted in red and yellow. Corresponding branchpoints are connected by thin green lines.
Figure 4.24. Automatic match between two pig airway trees. The trees were extracted from scans taken before and after a bronchoscopic procedure [24]. Matched branches are depicted in blue for the first tree and black for the second. Unmatched branches are depicted in red and yellow. Corresponding branchpoints are connected by thin green lines.

4.3.3 Other Matching Results

Although human airway trees extracted from MDCT chest scans are a primary focus of this thesis, the proposed graph-theoretic tree-matching framework is applicable for many different types of anatomical trees. For instance, we have successfully applied the same algorithm used to match pairs of human airway trees to automatically match two pig airway trees extracted from CT scans acquired before and after a bronchoscopic surgical procedure [24]. The automatically generated match achieved an accuracy of 99.2% when compared to a ground-truth hand matching and is illustrated in Figure 4.24.

We have also applied the proposed algorithm to match pairs of mouse coronary-arterial trees extracted from the same micro-CT image by different segmentation and axial-analysis
methods [51,168]. One algorithm parameter change was necessary for this task, as a mouse heart is drastically smaller than either a human or pig airway tree. Specifically, we reduced the threshold $t_d$ in the vertex distance component of the similarity measure (4.12) from $75\,\text{mm}$ to $40\,\mu\text{m}$. The remaining components of the measure depend upon scale-invariant quantities such as branching angles and the length ratio of two branches and were left unchanged. In this case, the automatically generated match achieved an accuracy of 94.3%. The match is illustrated in Figure 4.25.

**Figure 4.25.** The trees were extracted from the same micro-CT scan (h61) using different segmentation and axial-analysis methods [51,168]. Matched branches are depicted in blue for the first tree and black for the second. Unmatched branches are depicted in red and yellow. Corresponding branchpoints are connected by thin green lines.
4.3.4 Comparison With Existing Work

We have compared the proposed automatic matching algorithm with an existing method for the six pairs of FRC/TLC airway trees summarized in Table 4.1. The existing method represents the current state-of-the-art in airway-tree matching and was implemented as described in published work by Tschirren et al. [17,169]. Algorithm parameters were tuned on a case-by-case basis to obtain the best possible results for each of the six cases. Table 4.2 summarizes the results. The overall accuracy achieved by the existing method is only 56.9%, compared with the 96.2% accuracy of the proposed method reported in Table 4.1.

<table>
<thead>
<tr>
<th>case</th>
<th>FRC size</th>
<th>TLC size</th>
<th>Num. Hand Matches</th>
<th>Num. Auto Matches</th>
<th>Num. Correct Matches</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>h002</td>
<td>136</td>
<td>230</td>
<td>126</td>
<td>63</td>
<td>58</td>
<td>61.4%</td>
</tr>
<tr>
<td>h008</td>
<td>114</td>
<td>162</td>
<td>108</td>
<td>62</td>
<td>51</td>
<td>60.0%</td>
</tr>
<tr>
<td>h018</td>
<td>70</td>
<td>136</td>
<td>67</td>
<td>28</td>
<td>26</td>
<td>54.7%</td>
</tr>
<tr>
<td>20349-3-28</td>
<td>116</td>
<td>575</td>
<td>110</td>
<td>55</td>
<td>45</td>
<td>54.5%</td>
</tr>
<tr>
<td>20349-3-36</td>
<td>304</td>
<td>408</td>
<td>276</td>
<td>119</td>
<td>107</td>
<td>54.2%</td>
</tr>
<tr>
<td>emphysema</td>
<td>164</td>
<td>216</td>
<td>160</td>
<td>80</td>
<td>70</td>
<td>58.3%</td>
</tr>
<tr>
<td>total:</td>
<td>904</td>
<td>1,727</td>
<td>847</td>
<td>407</td>
<td>357</td>
<td>56.9%</td>
</tr>
</tbody>
</table>

Table 4.2. Summary of matching results obtained for the six cases from Table 4.1 using the method of Tschirren et al. [17,169]. The columns labeled “FRC size” and “TLC size” refer to the number of branches in the trees extracted from the FRC and TLC scans. Accuracy is computed via (4.89).

It is likely that the published works used to implement the existing method omitted minor details that could have improved the method’s performance. Thus, the results in Table 4.2 should be considered as a lower bound on the accuracy of the existing method. We note, however, that the results we have obtained are in agreement with previously-published results for the method [17]. Both our experiments and the previously-published results demonstrate that the existing method matches very few branchpoints. The authors state that the method produces so few matches because it was tuned to favor specificity.
over sensitivity and report the method’s accuracy using only the specificity measure

\[
\text{specificity} = \frac{\# \text{ of correct matches}}{\# \text{ of automatic matches}}. \tag{4.90}
\]

By this measure, the existing method appears to perform much better, and achieves a specificity of 87.8% compared with the 96.0% specificity achieved by the proposed method.

We believe that the accuracy measure in (4.89), which measures both the specificity and sensitivity of an automatic algorithm, provides a better overall indication of method performance. Figure [4.26] gives an illustrative example of a case where the existing method achieves high specificity, but produces an inaccurate and unacceptable result. The existing method operates by first locating and matching “major” branchpoints between $T_1$ and $T_2$, then matching the subtrees beneath each pair of matched major branches. This strategy frequently results in sub-optimal matches. In Figure [4.26] for example, an error in the major branchpoint matching leads to numerous missed matches, as the subtrees beneath the incorrectly-matched major branchpoints do not correspond. The resulting match is incomplete and inaccurate. The specificity measure in (4.90), however, only penalizes the match for a single incorrect correspondence, and ignores the missing matches completely.

### 4.3.5 Empirical Running Time Analysis

Section 4.2.4.3 analyzed the proposed automatic tree-matching algorithm and suggested a running time proportional to $O(|V(T_1)| \cdot |V(T_2)|)$ for anatomical trees of bounded degree, where $|V(T_1)|$ and $|V(T_2)|$ represent the number of branchpoints in $T_1$ and $T_2$. Table 4.3 presents empirical running times measured for the ten cases described in the section.
Figure 4.26. Visual comparison of matches obtained by an existing method (a) and the proposed method (b). The existing matching method was implemented as described in published work by Tschirren et al. [17, 169]. The trees were extracted using the method of Kiraly et al. [51]. Matched branches are depicted in blue and black, while unmatched branches are depicted in red and yellow for FRC and TLC, respectively. Corresponding branchpoints are connected by a green line. Here, the existing method is trapped at a locally optimal solution in the left lung (highlighted by the dotted oval) where a single incorrect correspondence leads to several missed correspondences. The proposed method finds the globally optimal solution with respect to our chosen similarity measure.

The running times were measured on a dual-core 2.6GHz PC with 4GB RAM running the Microsoft Windows XP operating system.

We provide experimental validation for the asymptotic estimate by plotting the measured running times against $|V(T_1)| \cdot |V(T_2)|$ for all ten cases. The result, illustrated in Figure 4.27, is very nearly linear, with an $R^2$ value of 0.994. Thus, the asymptotic $O(|V(T_1)| \cdot |V(T_2)|)$ estimate is reasonable for all matching cases considered to date.
Table 4.3. Tree-matching algorithm running times for ten cases measured on a dual-core 2.6GHz PC with 4GB RAM running Windows XP.

| case             | $|V(T_1)|$ | $|V(T_2)|$ | Algorithm run time (sec) |
|------------------|--------|--------|------------------------|
| mouse arterial-tree | 64     | 124    | 0.3                    |
| h018             | 70     | 136    | 0.8                    |
| h008             | 114    | 162    | 1.8                    |
| pig airway-tree  | 138    | 172    | 2.7                    |
| 20349-3-29       | 160    | 200    | 2.7                    |
| h002             | 136    | 230    | 2.9                    |
| 21405-3a         | 138    | 224    | 2.9                    |
| emphysema        | 164    | 216    | 3.5                    |
| 20349-3-28       | 116    | 575    | 6.2                    |
| 20349-3-36       | 304    | 408    | 11.5                   |

Figure 4.27. Plot of proposed algorithm run time versus problem size for the ten cases detailed in Table 4.3. The $R^2$ value for the best-fit line is 0.994.
Chapter 5

Human Airway-Tree Labeling

This chapter describes an application of the robust tree-matching methodology of Chapter 4 to the problem of accurately labeling human airway-tree anatomy. The chapter is organized in three parts. First, Section 5.1 describes an interactive labeling tool that enables the collection of a database of hand-labeled airway trees. Such a database is important for both the development and validation of an automatic labeling algorithm. Next, Section 5.2 describes an automatic tree-labeling algorithm developed using the DP tree-matching framework of Chapter 4. Finally, Section 5.3 presents experimental results obtained using both the interactive labeling tool and the automatic labeling algorithm.

5.1 Interactive Labeling Tool

The development of a robust and automatic tree-labeling algorithm requires a large database of hand-labeled trees covering the range of human anatomy. This section describes a computer-based tool facilitating the collection of hand-labeled data. The software takes as input an MDCT chest scan and an unlabeled centerline representation of the airway tree...
depicted in the scan.

The labeling tool first presents the user with a 3D rendering of the tree. Initially, the branches are unlabeled and are represented by uniformly colored lines. Figure 5.1 gives an example of this view. Keyboard and mouse controls allow the user to render the tree from an arbitrary 3D camera position. A small stick-figure “compass” in the bottom-left corner of the screen helps orient the user. An auxiliary view presents Netter’s well-known map of the airway tree [141]. The map provides a 2D view of a “typical” airway tree. The user labels the tree by clicking individual branches in the rendering and selecting the appropriate label from a hierarchical list. As the tree is labeled, the colors of labeled branches change to match the colors of the anatomical map and the labeled branch names are displayed as text on the screen. The tool contains standard undo/redo functionality.

Figure 5.2 illustrates the labeling process. Our goal is to identify airway-tree anatomy to the segmental level. Thus, between 30 and 40 unique labels are assigned to each tree. Note that the user need not tediously label every branch of the input tree. Once a segmental bronchus has been assigned, its descendants are automatically assigned the same segmental label. In Figure 5.2, for example, all remaining unlabeled branches will be assigned to the apical segment once the user labels the segmental bronchus. Once all labels have been assigned, the tool outputs an augmented centerline representation including the label of each branch. Appendix I provides a user manual for the interactive labeling tool.
5.2 Automatic Labeling Algorithm

The overall strategy of the proposed tree-labeling algorithm is to match an initially unlabeled input tree to a labeled model tree and transfer labels between corresponding branches between the two trees. This strategy is naive, as a single model tree cannot capture the full range of anatomical variations known to occur in humans, but can be effective for many cases when a robust tree-matching method is used.

The model tree is taken from case 21405-3a. This tree was chosen because it exhibits the standard anatomy depicted in Figure 5.1 and contains subsuperior basal segments in both the left and right lower lobes. Subsuperior basal segments occur in approximately half of the population and must be included in the model to be automatically recognized in unlabeled input trees [144, 146]. All branches below the segmental level were pruned from the model tree. There are two reasons for this. First, this reduces the size of the model to 43 branches from nearly 300 branches. More importantly, airway-tree branching patterns exhibit a large amount of variation beyond the segmental level [141, 144]. Branches located
Figure 5.2. An example illustrating the label assignment process in the tree-labeling tool. The user selects an unlabeled branch and chooses the correct label from a hierarchical list. Once a label has been chosen, the color of the labeled branch changes to match the corresponding color in Netter's 2D anatomical map [141].

beyond the segmental level are correspondingly difficult to match and can interfere with the main problem of matching segmental bronchi [17]. Figure 5.3 depicts the pruned model tree.

Denote the initially unlabeled input tree by $T$ and the labeled model tree by $M$. The labeling algorithm labels $T$ in three steps, as illustrated in Figure 5.4:

1. The major bronchi (trachea, left main bronchus, and right main bronchus) are labeled using a method inspired by the work of Tschirren et al. [17]. The labeled major bronchi are used to align $T$ and $M$ in a common coordinate system (Section 5.2.1, Figure 5.4a).

2. The lobar bronchi (RUL, RML, RLL, LUL, and LLL) are labeled using the method of Mori et al. [28]. One segmental bronchus ($RB^6$) is also labeled during this step.
Figure 5.3. The labeled model tree used in the proposed automatic labeling algorithm. The tree is from case 21405-3a. All branches below the segmental level have been removed. The subsuperior basal segments are colored bright blue in both the left- and right- lower lobes. Remaining branch colors correspond to Netter’s 2D tree label map [141].

(Section 5.2.2, Figure 5.4b).

3. The labeled lobar bronchi are used to identify subtrees of $T$ corresponding to each of the five lung lobes and rigidly align each subtree with its corresponding lobar subtree in $M$. The segmental bronchi are labeled by matching corresponding lobar subtrees using the framework of Chapter 4 (Section 5.2.3, Figure 5.4c).

5.2.1 Major Bronchi Labeling

The proposed algorithm labels the major bronchi by first locating the end vertex of the trachea ($endT$), then locating the end of the left and right main bronchi ($endLMB$ and $endRMB$). The vertex $endT$ is the point at which the airway tree bifurcates into subtrees spanning the right and left lungs. Thus, the end of the trachea is naturally characterized as the branchpoint with the largest pair of child subtrees. To locate the trachea, the proposed
Figure 5.4. Progression of the proposed automatic tree-labeling algorithm. (a) The algorithm first labels the major bronchi (trachea, RMB, and LMB). (b) Next, the algorithm labels the lobar bronchi. One segmental bronchus ($RB^6$) is also labeled during this step. (c) Finally, the algorithm labels the segmental bronchi by posing five independent subtree-matching problems. Here, each of the five highlighted lobar subtrees is independently matched to its corresponding lobar subtree in the model. Segmental branch colors correspond to Netter’s 2D tree map [141]. The label for $RB^6$ (purple branch) is omitted as it is assigned in the lobar labeling step. The left and right subsuperior basal segments ($RB^*$ and $LB^*$) are represented with dotted lines as they are missing in approximately half of all patients.
algorithm assigns an importance score $I$ to each non-leaf vertex $u \in V(T)$. The importance of $u$ is the geometric mean of the number of branchpoints in its two largest child subtrees. Formally,

$$I(u) = \max \left\{ \sqrt{|V(T_x)| \cdot |V(T_y)|} \mid x, y \in C_u \right\}. \quad (5.1)$$

The end vertex of the trachea is taken to be the vertex maximizing $I$; i.e.,

$$endT = \arg \max \{I(u) \mid u \in V(T)\}. \quad (5.2)$$

The vertices $endRMB$ and $endLMB$ are determined as follows. First, let $a$ be the descendent of $endT$ maximizing $(5.1)$. Next, let $b$ be the descendent of $endT$ maximizing $(5.1)$ subject to the constraint $\text{lca}(a, b) = endT$. Thus, $a$ and $b$ are the two most important vertices belonging to different child subtrees of $endT$. The algorithm sets $endRMB = a$ and $endLMB = b$ if the $x$-coordinate of vertex $a$’s image location is greater than the $x$-coordinate of vertex $b$’s image location. Otherwise, the algorithm sets $endRMB = b$ and $endLMB = a$. This simple strategy of comparing image coordinates to distinguish between the left and right main bronchi works because patients lie in a standard position when receiving a chest scan.

Once the major bronchi have been identified, the input tree $T$ is rigidly transformed into a canonical coordinate system in two steps. First, $T$ is translated so that $endT$ lies at the origin. Second, $T$ is rotated so that the plane containing $endT$, $endRMB$, and $endLMB$ is normal to the $y$-axis, the line bisecting the angle connecting $endRMB$ to $endT$ to $endLMB$ points along the positive $z$-axis, and the $x$-coordinate of $endRMB$ is negative.

This simple method for determining the ends of the major bronchi is surprisingly reli-
able. The method accurately locates the major bronchi even in the presence of aberrant anatomical segments, as illustrated in Figure 5.5(b)-(c). The method will fail, however, if either the left or right lung is largely incomplete or missing entirely. In such a case, branch labels will need to be corrected interactively using the tree-labeling tool of Section 5.1.

Figure 5.5. Major bronchi labeling examples. The proposed major bronchi labeling algorithm correctly locates the end vertices of the trachea ($endT$), right main bronchus ($endRMB$), and left main bronchus ($endLMB$) in cases with both normal anatomy (a) and cases exhibiting aberrant segments (b) and (c). The illustrated cases are (a) 20349-3-7, (b) 20349-3-3, and (c) h002. Colors in the illustrated trees correspond to ground-truth segmental labels.
5.2.2 Lobar Bronchi Labeling

Now that the major bronchi have been labeled and $T$ has been rigidly transformed into a canonical coordinate system, the algorithm attempts to label the lobar bronchi. Specifically, this section describes the method for locating the ends of the intermediate bronchus ($endBI$), right upper lobe bronchus ($endRUL$), right middle lobe bronchus ($endRMB$), right superior basal segment ($endRB^6$), left upper lobe bronchus ($endLUL$), and left lower lobe bronchus ($endLUL$). The method is adapted from published work by Mori et al. [28], and proceeds in three steps:

1. Label $endBI$ and $endRUL$ given $endT$ and $endRMB$.

2. Label $endLUL$ and $endLLL$ given $endT$ and $endLMB$.

3. Label $endRML$, $endRLL$ and $endRB^6$ given $endRMB$ and $endBI$.

For concreteness, we describe the method for performing the labeling in step[1] The methods for steps[2] and [3] are similar.

Let $n$ represent the 3D unit vector pointing along the line connecting $endT$ to $endRMB$ in $T$ and let $c_1$ and $c_2$ represent unit vectors pointing from $endRMB$ to its children $x$ and $y$ in $T$. Similarly, let $n_M$ represent unit vector along the $endT$-$endRMB$ line in $M$ and let $c_{RUL}$ and $c_{BI}$ point along the $endRMB$ – $endRUL$ and $endRMB$ – $endBI$ lines in $M$. Figure[5.6](a) summarizes the notation for this step.
Figure 5.6. A schematic example illustrating the lobar bronchi labeling method. (a) The main bronchi (endT, endRMB, endLMB) of the input tree T are already known. This step will assign the labels endRMB and endBI by comparing the input tree to the model tree M. (b) The input tree is rigidly rotated by $R_{n,n_M}$ so that the right main bronchi of the two trees are aligned. The labels endRUL and endBI are assigned in $T$ to maximize the sum of inner products between corresponding labeled branches in the rotated input and model trees. Here, endRUL = $y$ and endBI = $x$ as (5.5) does not hold.

Define $R_{n,n_M}$ to be the 3D rotation matrix aligning $n$ with $n_M$ in the plane normal to $n \times n_M$. Specifically,

$$R_{n,n_M} = \left( \begin{array}{ccc} r & u & \frac{n \times n_M}{\|n \times n_M\|} \end{array} \right) \left( \begin{array}{ccc} \cos(\phi) & -\sin(\phi) & 0 \\ \sin(\phi) & \cos(\phi) & 0 \\ 0 & 0 & 1 \end{array} \right) \left( \begin{array}{c} r^T \\ u^T \\ \frac{n \times n_M}{\|n \times n_M\|}^T \end{array} \right) ,$$

(5.3)
where \( \mathbf{r} \) and \( \mathbf{u} \) are 3D unit vectors chosen so that \( \mathbf{r}, \mathbf{u}, \) and \( \frac{\mathbf{n} \times \mathbf{n}_M}{\|\mathbf{n} \times \mathbf{n}_M\|} \) form a right-handed orthonormal frame and

\[
\phi = \arctan\left( \frac{\mathbf{u}^T \mathbf{n}}{\mathbf{r}^T \mathbf{n}} \right) - \arctan\left( \frac{\mathbf{u}^T \mathbf{n}_M}{\mathbf{r}^T \mathbf{n}_M} \right)
\]  

(5.4)

is the angle of rotation around \( \mathbf{n} \times \mathbf{n}_M \) that aligns \( \mathbf{n} \) and \( \mathbf{n}_M \). Thus, \( R_{\mathbf{n}, \mathbf{n}_M} \) aligns \( \mathbf{n} \) and \( \mathbf{n}_M \) without “twisting” about \( \mathbf{n} \). The labels \textit{endRUL} and \textit{endBI} are assigned to maximize the sum of inner products between corresponding labeled branches in the rotated input and model trees. Formally, \textit{endRUL} = \( x \) and \textit{endBI} = \( y \) if

\[
c_{\text{RUL}}^T \left( R_{\mathbf{n}, \mathbf{n}_M} \mathbf{c}_1 \right) + c_{\text{BI}}^T \left( R_{\mathbf{n}, \mathbf{n}_M} \mathbf{c}_2 \right) > c_{\text{RUL}}^T \left( R_{\mathbf{n}, \mathbf{n}_M} \mathbf{c}_2 \right) + c_{\text{BI}}^T \left( R_{\mathbf{n}, \mathbf{n}_M} \mathbf{c}_1 \right).
\]  

(5.5)

Otherwise, \textit{endRUL} = \( y \) and \textit{endBI} = \( x \). Figure 5.6(b) provides an illustrative example.

The lobar labeling method well-tolerates differences in branching angles between \( T \) and \( M \) as the rigid rotation \( R_{\mathbf{n}, \mathbf{n}_M} \) provides a local alignment between the two trees and the algorithm only decides between two or three labels at a time. In Figure 5.6, for example, the method assigns correct lobar labels even through the running directions of corresponding bronchi (\( \mathbf{c}_1 \) versus \( c_{\text{RMB}} \) for the right upper lobe bronchus and \( \mathbf{c}_2 \) versus \( c_{\text{BI}} \) for the intermediate bronchus) are quite different in \( T \) and \( M \). The method does not, however, consider the possibility of false branches or aberrant segments incident upon the lobar bronchi as our database of hand-labeled trees does not yet include examples of corrupted lobar bronchi. Any errors introduced by corrupted lobar bronchi will again need to be corrected interactively using the proposed tree-labeling tool.
5.2.3 Segmental Bronchi Labeling

Once the lobar bronchi have been identified, the algorithm labels segmental bronchi by independently matching each of the five lobar subtrees of $T$ to their corresponding subtrees in $M$. The lobar subtrees are defined to be the bottom-up subtrees rooted at the vertices labeled $endRUL, endRML, endRLL, endLUL$, and $endLLL$.

The method proceeds as follows. Let $I$ be a unit normal vector representing the running direction of a lobar bronchus in $T$ and let $I_M$ be the running direction of the corresponding lobar bronchus in $M$. The lobar subtree of $T$ is aligned with the corresponding lobar subtree of $M$ via a rigid rotation by $R_{I,I_M}$, as defined in \((5.3)\). The lobar subtrees are then matched under the secondary deformation model defined in Section 4.2.1.2. The optimal match maximizes a similarity measure given by the sum of inner-products between normalized running directions of corresponding branches in the rotated input tree and the model. No branch length measure is used as the lengths of segmental bronchi can vary significantly between airway trees. Similarly, no relative branchpoint location measure is used, as corresponding branchpoints are frequently located far apart even after $T$ and $M$ are rigidly aligned. Finally, labels for the segmental bronchi are transferred from $M$ to the corresponding branches in $T$.

Use of the secondary deformation model for matching lobar subtrees helps ameliorate some of the difficulties associated with a single-tree model of human airway tree anatomy by providing the labeling algorithm with a tolerance for mild anatomical variations. For example, the secondary deformation model naturally handles the four common branching-pattern variations occurring in the right upper lobe, even though the model tree can exhibit
only one of the four patterns. Figure 5.7 illustrates the four variants. The branching pattern of the model tree is the trifurcation illustrated in Figure 5.7(a). The patterns in (b)-(d) can be matched to (a), however, using the secondary primitive illustrated in Figure 4.7, which combines two close bifurcations in one tree to match a trifurcation in the other.

Similarly, the proposed matching framework also naturally handles common variations involving subsuperior basal segments in the right or left lungs. Recall that case 21405-3a was chosen for the model tree in part because it had subsuperior basal segments in both lungs. Figure 5.4(c) illustrates the branching pattern of the model, in which the right subsuperior basal segment ($RB^*$) branches off just after $RB^7$ and the left subsuperior basal segment ($LB^*$) branches off just after $LB^6$. If either $RB^*$ or $LB^*$ is missing from $T$, the corresponding segment in $M$ is simply treated as a “spurious” branch, remains unmatched, and does not assign its label to any branch in $T$. Similarly, variations in which $RB^*$ branches off before $RB^7$ or after $RB^8$ are naturally handled by the primitive illustrated in Figure 4.6, which reverses the order of two well-matched branches $T$ and $M$.

The proposed matching framework does not, however, suffice to handle all anatomical variations. Figure 5.8 gives an example of a variation in the right lower lobe producing two lobar subtrees that cannot be properly matched under the secondary deformation model. Here, $RB^8$ branches from an unusual location in the input tree. Correctly matching all of the segmental bronchi would entail treating the end of the right lower lobe bronchus as a 4-furcation with child branches $RB^7$, $RB^8$, $RB^*$, and the bronchus leading to $RB^9$ and $RB^{10}$ in both $T$ and $M$. Even then, the algorithm would be unlikely to select such a match as the atypical branching pattern exhibited by the input tree is typically accompanied by unusual branching angles for both $RB^7$ and $RB^8$ [144]. The inability to recognize such
Figure 5.7. Four common branching patterns in the right upper lobe. A schematic diagram in the upper-left corner of each figure offers a simplified depiction of the illustrated branching pattern. (a) All three segmental bronchi branch independently from the upper lobe bronchus. (b)-(d) One segmental bronchus branches independently from the upper lobe bronchus, while the other two share a common trunk. For example, RB2 and RB3 share a common trunk in (c). The common trunk receives the label RB$^{2+3}$ by convention.
Figure 5.8. A branching pattern variation in the right lower lobe that is not handled by the proposed automatic labeling algorithm.

variations is therefore an inherent limitation in a single-tree model of human anatomy. The incorporation of multiple labeled trees into the anatomical model is therefore an obvious avenue for future work and is discussed in Chapter 7.

5.3 Results

This section describes two sets of labeling results. First, Section 5.3.1 describes an initial feasibility study of the usefulness of the proposed interactive tree-labeling tool. Section 5.3.2 then presents results obtained using the proposed automatic labeling algorithm.

5.3.1 Interactive labeling

We have conducted a pilot study in conjunction with physicians at Penn State’s Hershey Medical Center to evaluate the feasibility of generating a collection of hand-labeled example trees. In the study, three physicians used an early prototype of the tree-labeling tool.
described in Section 5.1 to label a small set of example trees. Each physician performed three different trials in labeling trees extracted from four MDCT scans (36 trials total). The trials were spaced at least one week apart. The physicians labeled the main, lobar, and segmental bronchi for each case.

Physicians spent an average of 11 ± 5 minutes (mean ± stdev) to completely label each tree. To evaluate the repeatability of the hand-labeled results, we have tabulated both intraobserver and interobserver agreement. Intraobserver agreement was assessed by comparing labels assigned to the same case by the same physician across two separate trials. Interobserver agreement was assessed by comparing labels assigned to the same case by two different physicians. A total of 36 intraobserver and 108 interobserver labeling comparisons were performed. The physicians agreed on nearly all the main and lobar bronchi of the four cases. Table 5.1 summarizes results for the segmental bronchi. The overall intraobserver agreement was 75% and the interobserver agreement was 67% with the largest number of errors occurring in the right upper and lower lobes.

<table>
<thead>
<tr>
<th>Lobe</th>
<th>Agreement</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>RUL</td>
<td>72 out of 108</td>
<td>67%</td>
</tr>
<tr>
<td>RML</td>
<td>56 out of 70</td>
<td>80%</td>
</tr>
<tr>
<td>RLL</td>
<td>101 out of 186</td>
<td>54%</td>
</tr>
<tr>
<td>LUL</td>
<td>135 out of 142</td>
<td>95%</td>
</tr>
<tr>
<td>LLL</td>
<td>122 out of 146</td>
<td>84%</td>
</tr>
<tr>
<td>Total</td>
<td>486 out of 652</td>
<td>75%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Lobe</th>
<th>Agreement</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>RUL</td>
<td>219 out of 324</td>
<td>68%</td>
</tr>
<tr>
<td>RML</td>
<td>120 out of 210</td>
<td>57%</td>
</tr>
<tr>
<td>RLL</td>
<td>219 out of 558</td>
<td>39%</td>
</tr>
<tr>
<td>LUL</td>
<td>405 out of 426</td>
<td>95%</td>
</tr>
<tr>
<td>LLL</td>
<td>347 out of 438</td>
<td>79%</td>
</tr>
<tr>
<td>Total</td>
<td>1310 out of 1956</td>
<td>67%</td>
</tr>
</tbody>
</table>

Table 5.1. Agreement results for segmental bronchi in an initial feasibility study of the proposed tree-labeling tool. Trial involved three physicians each labeling four trees three different times. The table therefore aggregates a total of 36 intraobserver and 108 interobserver comparisons. The four cases considered in the study were h006, 21405-17, 20349-3-6, and 20349-3-15.

The relatively low interobserver agreement percentage highlights the difficulty of the
labeling problem, even for trained experts. The results in Table 5.1, however, paint an overly-pessimistic picture for several reasons. First, these results include labelings created during the physicians’ first-ever experience with an early prototype version of the labeling tool. This early version of the tool included only straight-line approximations for each airway and did not incorporate the raw MDCT image data. The physicians were therefore unaware of the size or shape of the airways they were labeling. More importantly, the physicians spent only 10-15 minutes on each case. Much of this time was spent assigning labels to “obvious” branches such as the trachea, left and right main bronchi, and lobar bronchi. Thus, relatively little time was spent assigning labels to the more difficult segmental bronchi.

The results of the initial feasibility study suggested several practical changes to both the labeling tool and our data-collection protocol. First, the labeling tool was updated to enable the user to view the airway centerlines superimposed on an arbitrary oblique slice through the MDCT image data. Figure 5.9 gives an example of this view, which provides the user with additional information, such as airway size and shape, that is not apparent from the centerline representation. Also, to reduce the burden of labeling on the physician, we now assign easy labels, such as the main bronchi, lobar bronchi, and segmental bronchi exhibiting normal branching patterns in our lab. This enables the physician to concentrate only on difficult segmental labels, especially those in the left and right lower lobes.

The improved tree-labeling tool and data-collection protocol has been used to completely label the main, lobar, and segmental bronchi of 12 human airway trees. In addition, the labeling software has been used to label the main and lobar bronchi in more than 20 cases as part of a computerized system for extracting and visualizing lymph nodes in the central
chest [170]. The labeled airways serve as important anatomical landmarks to aid the system in locating established lymph-node “stations” [171]. Knowledge of the station locations facilitates the rapid location, extraction, and classification of the central chest lymph nodes.

Figure 5.9. A view of the tree-labeling tool with the labeled airways superimposed on an oblique slice through the MDCT image data.

5.3.2 Automatic labeling

We have tested the proposed automatic labeling algorithm on the 12 hand-labeled example trees we have collected to date. One tree (case 21405-3a) was reserved for the model, as described in Section 5.2, and the remaining 11 were labeled by proposed algorithm. The algorithm correctly labeled the main and lobar bronchi in all 11 test cases. The method also assigned correct labels to 88% of the segmental bronchi. Table 5.2 provides a summary of the results broken out by lung lobe.

The automatically generated labelings were completely correct in 5 of the 11 test set cases. Figure 5.10 gives an example of a correctly-labeled case. Most labeling errors occurred
Table 5.2. Accuracy results for automatic labeling of segmental bronchi.

<table>
<thead>
<tr>
<th>Lobe</th>
<th>Proportion of Correct Labels</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>RUL</td>
<td>31 out of 33</td>
<td>94%</td>
</tr>
<tr>
<td>RML</td>
<td>22 out of 22</td>
<td>100%</td>
</tr>
<tr>
<td>RLL</td>
<td>49 out of 62</td>
<td>79%</td>
</tr>
<tr>
<td>LUL</td>
<td>42 out of 44</td>
<td>95%</td>
</tr>
<tr>
<td>LLL</td>
<td>40 out of 48</td>
<td>83%</td>
</tr>
<tr>
<td>Total</td>
<td>184 out of 209</td>
<td>88%</td>
</tr>
</tbody>
</table>

in the left and right lower lobes of cases in which the branching pattern of the test set tree differed significantly from that of the model. Figure 5.11 gives an example of one such case, in which the branching patterns of the input and model trees correspond to the schematic diagram from Figure 5.8. Correctly handling such anatomical variations will require the collection of a significantly larger training set of ground-truth labeled trees and the incorporation of the variations observed in the training set into the model used by the automatic algorithm.

The automatic algorithm was implemented in Matlab and tested on a dual-core 2.6GHz PC with 4GB RAM running Windows XP. Automatic labeling was performed in under five seconds for each of the test set cases. It is expected that a C++ implementation of the algorithm would operate very quickly as the lobar subtree-matching step, which consumes the vast bulk of the processing time of the current Matlab implementation, has already been shown to be efficiently solvable in C++. Recall that the model tree contains only 43 branches. Furthermore, no lobar subtree of the model contains more than 10 branches (RLL) and several of the lobar subtrees contain only 2 or 3 branches (RML,RUL). Thus, each lobar subtree-matching problem is significantly smaller than any of the tree-matching problems considered in Section 4.3. This efficiency will be important for future improvements to
the labeling algorithm. In Chapter 7, for example, we describe an improved algorithm which matches the lobar subtrees of the unlabeled input tree to a number of labeled lobar subtrees covering the range of human anatomy. Such an algorithm can be practical only if the lobar-subtree matching problem can be solved quickly.

Figure 5.10. Example of a correct automatically-labeled tree. All main, lobar, and segmental bronchi were correctly assigned. The illustrated case is 20349-3-6.

Figure 5.11. Example of errors in an automatically-labeled result. Both (a) and (b) provide closeup views of the right lower lobe from case 20349-3-16. The ground-truth labeling determined with physician assistance is depicted in (a). The branching pattern is an anatomical variation illustrated schematically in Figure 5.8(a). The automatically generated labeling is depicted in (b). The algorithm incorrectly assigns labels that agree with the standard anatomy present in the model tree. The branching pattern of the model tree is depicted schematically in Figure 5.8(b).
Image-Guided Peripheral Bronchoscopy Pilot Study

Successful bronchoscopic biopsy of suspect lung tissue enables the early diagnosis and assessment of lung cancer while avoiding the increased risk of complications associated with more invasive procedures. Bronchoscopic biopsy is difficult, however, as the target tissue is frequently located beyond the visible airway walls. Biopsy of peripheral sites is even more problematic, as the bronchoscope must first be correctly navigated through many airway generations before reaching an appropriate diagnostic site. The reported yields of peripheral bronchoscopic procedures are correspondingly low [29,31].

Several methods have been proposed to aid the bronchoscopist in successfully navigating to and sampling a target site. Fluoroscopy and CT-fluoroscopy, for example, provide the bronchoscopist with intraoperative radiological images [33,65]. Endobronchial ultrasound (EBUS) and electromagnetic (EM) systems provide guidance information obtained using probes inserted through the bronchoscope’s working channel [66,69,70,73,76–79].
An alternate approach, which provides Virtual Bronchoscopic (VB) guidance derived solely from a patient’s multi-detector CT (MDCT) chest scan, has shown promise towards improving peripheral bronchoscopy performance without requiring significant additional hardware [24,36,39,78,81,82,88–90]. In particular, works by Asano et al. and Shinagawa et al. have demonstrated the potential for VB guidance to improve diagnostic yields and shorten overall procedure times for peripheral bronchoscopy [36,39,88,89].

We are developing a VB guidance system that incorporates several novel elements designed to improve both the utility and the practicality of VB guidance for peripheral bronchoscopy. First, our system applies automatic image-processing methods to define the 3D airway model used for VB image generation and to determine an appropriate endobronchial route to the target site. Using our system, a trained technician can plan a complete peripheral bronchoscopic procedure with physician interaction limited to indicating target diagnostic sites in the MDCT data. This contrasts with previous VB-guidance systems, which placed significant pre-operative burdens of the physician [88, 89]. The system also contains an automated mechanism for generating compact, web-based pre-bronchoscopic reports for helping the bronchoscopist to understand what to expect during the subsequent image-guided procedure [21, 94]. Finally, the system employs a method for automatically synchronizing the VB view with the live bronchoscopic video feed during the procedure, thus enabling precise intraoperative determination of the bronchoscope’s position and orientation in the airways [96].

The proposed navigation system has already been shown to provide reliable guidance for the central chest lymph nodes in humans and to facilitate highly-accurate bronchoscopic localization of peripheral lesions in a rigid plastic phantom [24,35]. The aim of the current
The study is to evaluate the safety and efficacy of the proposed navigation system for peripheral bronchoscopic guidance in humans.

6.1 Methods

6.1.1 Subjects

The study was performed at the Pennsylvania State University’s Hershey Medical Center and included 15 patients enrolled between October 2007 and April 2008. Patients were selected for the study if they presented with peripheral lesions requiring the use of an ultrathin bronchoscope or central chest lesions for which standard bronchoscopy was presumed to be difficult and guidance was likely to be useful. The study was approved by the university’s Institutional Review Board and informed consent was obtained from all patients.

6.1.2 VB Guidance System

Each guided procedure consisted of both an offline procedure-planning stage and a live bronchoscopic-guidance stage.

6.1.2.1 Procedure Planning

Each patient received a thin-slice MDCT scan with slice thickness 0.75 mm, slice spacing 0.50 mm, and in-plane resolution between 0.50 mm and 0.86 mm. The bronchoscopist indicated target sites by pointing to transverse slices of the scan viewed in the hospital’s picture archiving and communication system. The MDCT scan was then input to our system, which extracted the 3D airway tree, computed the endoluminal surfaces of the
airways, defined a 3D endobronchial route leading to the diagnostic site, and produced an interactive pre-bronchoscopic report for previewing the procedure. Figure 6.1 illustrates the procedure-planning process, which is almost-fully automatic [57].

### 6.1.2.2 Live Image-guided Procedure

In the bronchoscopy suite, a technician interfaced a computer running the guidance system software with the bronchoscope. The system provided the bronchoscopist with continually-updated displays depicting the live bronchoscopic video feed, the current global location and orientation of the bronchoscope within the patient’s airways, and the VB rendering corresponding to the current bronchoscope position (Figure 6.2). Supplemental displays provided the bronchoscopist with a history of previously-visited locations. The preplanned endobronchial route was indicated on all displays by a blue line leading the bronchoscopist to the diagnostic site.

Guidance to each diagnostic site proceeded as follows. First, the technician navigated the VB view to an agreed-upon location along the preplanned endobronchial route, typically the point at which the route entered a lobar bronchus. The bronchoscope was then navigated to a location similar to the VB view and the VB rendering adjusted to match the view of the bronchoscope. Once the VB and bronchoscopic video views had been synchronized, the blue line indicating the preplanned route unambiguously indicated the next bronchus along the route. The bronchoscopist directed the technician to advance the VB view to the next branching point. This process was repeated as the bronchoscope was advanced through the airways. Towards the end of the preplanned route, the system presented the bronchoscopist with additional visual cues, including an arrow indicating the optimal diagnostic site, a
Figure 6.1. The offline procedure-planning stage of the VB guidance system. (a) The physician indicates the target lesion directly in a patient’s MDCT scan. Here the target lesion is a cavitated nodule in the right upper lobe (diameter: 2 cm). (b) A patient-specific 3D airway model is extracted from the MDCT data. The green object in the right upper lobe is the target lesion. (c) An optimal 3D route to the lesion (blue line) is determined automatically. Here, the illustrated route ends in a 9th generation bronchus. (d) A pre-bronchoscopic report provides pictures of each bifurcation along the route and an interactive movie of the entire route to preview the procedure.
Figure 6.2. A live screen capture of the VB guidance system during a peripheral bronchoscopic procedure (case 20349-3-28). (a) A global 3D airway-tree rendering. The target lesion is colored green. The white arrow indicates a small cylinder representing the current global position and orientation of the bronchoscope tip. (b) The live bronchoscopic video feed. (c) The corresponding VB rendering at the current bronchoscope location. (d),(e) “Frozen” views of the bronchoscopic video and VB rendering taken from the last branchpoint along the route.

3D rendering of the extraluminal target lesion, and quantitative information indicating the distance from the current position of bronchoscope to the lesion (Figure 6.3).

6.1.3 Procedure Details

Procedures were performed by one of two staff bronchoscopists using either an ultrathin Olympus BF Type XP160F (outer diameter: 2.8 mm; channel diameter: 1.2 mm) bronchoscope or a Pentax Type EB-1570K (outer diameter: 5.5 mm; channel diameter: 2.0 mm) bronchoscope. The larger Pentax bronchoscope was required for procedures involving transbronchial needle aspiration (TBNA) of target lesions, as no needle was available for the
ultrathin bronchoscope. Additional diagnostic samples were obtained by cytologic brushing and bronchoalveolar lavage (BAL). Each procedure was recorded for post-operative analysis. Bronchoscopies were performed with the patient either under conscious sedation and anesthetized with topical lidocaine or under general anesthetic at the bronchoscopist’s discretion. Radiographic fluoroscopy was available during several procedures, but was used only to corroborate the final position of the bronchoscope after image-based guidance.

6.1.4 Data Analysis

Metrics assessing the performance of both the offline procedure-planning and live-guidance aspects of the system were considered. First, the efficacy of the system in determining an appropriate endobronchial route for each target diagnostic site was measured by both the minimum distance from the automatically-determined biopsy site to the target lesion and the maximum airway generation at which informative VB renderings were available along the route. The utility of the system in providing intraoperative guidance was measured by
the maximum depth to which the bronchoscope could be inserted by direct vision along the route and the proportion of target sites for which a sample was successfully obtained along the preplanned route. The location of the bronchoscope throughout the procedure was determined by retrospective comparison of the recorded bronchoscopic video feed with the reconstructed VB airway model and the patient’s pre-operative MDCT scan.

6.2 Results

The VB guidance system provided route planning and live bronchoscopic guidance for a total of 31 diagnostic sites of various sizes, types, and locations. Table 6.1 summarizes the diagnostic sites considered in the study and Table 6.2 presents guidance results for each site. The preplanned routes generated by the system traversed an average of 8.6 ± 2.9 bronchial generations (mean ± standard deviation, median: 8, range: 4-13). The route length increases to 9.5 ± 2.3 bronchial generations (median: 8.5, range: 7-13) if only the 26 peripheral routes planned for ultrathin bronchoscopy are considered. Informative VB images were available along the entire extent of each preplanned route. The routes frequently led directly to the target diagnostic site, advancing to within 5 mm for 26 sites (84%) and to within 20 mm for all 31 sites.

The bronchoscopist was able to maneuver through an average of 7.0 ± 2.1 (median: 7, range: 3-13) bronchial generations. The bronchoscope traversed the entire preplanned route for 12 sites (39%), including routes of 12 and 13 bronchial generations. Figures 6.4-6.5 provide comparisons between VB images produced by the navigation system and live bronchoscopic video frames along the 13-generation route. The target diagnostic site, a
### Table 6.1. Summary of target diagnostic sites. Point targets were airway sites selected as favorable locations for bronchoalveolar lavage or visual bronchial tissue inspection.

<table>
<thead>
<tr>
<th>Diagnostic site type</th>
<th>No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nodule (diameter &lt; 3 cm)</td>
<td>8</td>
</tr>
<tr>
<td>Mass (diameter ≥ 3 cm)</td>
<td>3</td>
</tr>
<tr>
<td>Ground-glass opacity</td>
<td>3</td>
</tr>
<tr>
<td>Infiltrate</td>
<td>3</td>
</tr>
<tr>
<td>Lymph node</td>
<td>2</td>
</tr>
<tr>
<td>Point target</td>
<td>6</td>
</tr>
<tr>
<td>Other</td>
<td>6</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>31</strong></td>
</tr>
</tbody>
</table>

14-mm diameter nodule located in RB<sup>8</sup>, was clearly seen to be impinging upon the carina at the end of the 13<sup>th</sup> generation bronchus. Visual confirmation of an 8<sup>th</sup> generation lesion in RB<sup>1</sup> was also possible in a second case, as illustrated in Figure 6.6.

Figure 6.4. Guidance results for the right lower lobe nodule in case 20349-3-24. The view on the left depicts a global exterior rendering of the airway-tree surfaces. The target nodule is colored red and the preplanned route is indicated by the blue line. The views on the right compare bronchoscopic video frames captured during the procedure with corresponding VB renderings generated by the guidance system. The 3<sup>rd</sup> and 5<sup>th</sup> bronchial generations are depicted here. Figure 6.5 follows the remainder of the route.

Figure 6.5 follows the remainder of the route.

Table 6.3 presents a comparison of the peripheral diagnostic sites considered during the
Figure 6.5. Video frames and the corresponding virtual bronchoscopic views along the remainder of the route depicted in Figure 6.4. The ROI can be seen impinging on the lower airway in the generation 13 video frame.

current study with those from a recent VB-guidance study of Shinagawa et al. [90]. The comparison indicates a statistically-significant increase in both the overall depth to which informative VB renderings were available for guidance (p < 0.001, Student’s t-test) and the actual depth of bronchoscope insertion (p < 0.001, Student’s t-test). Neither the current study nor the previous study were randomized, however, and a selection bias is possible. This issue is discussed further in the next section.
Figure 6.6. Guidance results for the right upper lobe lesion in case 20349-3-42. (a) An oblique (non-orthogonal) cross-section through the MDCT data. Each pixel in the cross-section is 0.5 mm × 0.5 mm. The arrows indicate the location of the target lesion and the bronchoscope for the video frames depicted below. (b) Corresponding video view and VB rendering with the bronchoscope one generation short of the target lesion. (c) Corresponding video view and VB rendering with the bronchoscope in the involved bronchus. The lesion is clearly visible as airway inflammation in the bottom orifice. (d) A fused video view depicting both the target lesion (green) and the preplanned route (blue) alongside the corresponding VB rendering. The video frame and renderings in (c) and (d) are the same.
<table>
<thead>
<tr>
<th>Case</th>
<th>Site loc.</th>
<th>Site type</th>
<th>Site diam. (mm)</th>
<th>Min. site distance (mm)</th>
<th>VB image depth (bronch. gen.)</th>
<th>Bronchoscope insertion depth (bronch. gen.)</th>
<th>Guidance successful (y/n)</th>
</tr>
</thead>
<tbody>
<tr>
<td>20349-3-24</td>
<td>RLL</td>
<td>N</td>
<td>14</td>
<td>1</td>
<td>13</td>
<td>13</td>
<td>y</td>
</tr>
<tr>
<td>20349-3-25</td>
<td>LUL</td>
<td>N</td>
<td>11</td>
<td>1</td>
<td>8</td>
<td>8</td>
<td>y</td>
</tr>
<tr>
<td>20349-3-26</td>
<td>LUL</td>
<td>N</td>
<td>6</td>
<td>14</td>
<td>8</td>
<td>7</td>
<td>y</td>
</tr>
<tr>
<td></td>
<td>RUL</td>
<td>M</td>
<td>30</td>
<td>2</td>
<td>8</td>
<td>7</td>
<td>y</td>
</tr>
<tr>
<td>20349-3-28</td>
<td>RUL</td>
<td>N</td>
<td>21</td>
<td>1</td>
<td>9</td>
<td>8</td>
<td>y</td>
</tr>
<tr>
<td></td>
<td>LUL</td>
<td>M</td>
<td>37</td>
<td>1</td>
<td>12</td>
<td>12</td>
<td>y</td>
</tr>
<tr>
<td>20349-3-29</td>
<td>LUL-1</td>
<td>M</td>
<td>42</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>y</td>
</tr>
<tr>
<td></td>
<td>LUL-2</td>
<td>N</td>
<td>10</td>
<td>11</td>
<td>13</td>
<td>13</td>
<td>y</td>
</tr>
<tr>
<td>20349-3-30</td>
<td>RLL</td>
<td>N</td>
<td>24</td>
<td>3</td>
<td>11</td>
<td>10</td>
<td>y</td>
</tr>
<tr>
<td>20349-3-33</td>
<td>RUL</td>
<td>I</td>
<td>49</td>
<td>0</td>
<td>13</td>
<td>6</td>
<td>y</td>
</tr>
<tr>
<td></td>
<td>RLL</td>
<td>I</td>
<td>41</td>
<td>0</td>
<td>10</td>
<td>4</td>
<td>n</td>
</tr>
<tr>
<td></td>
<td>LLL</td>
<td>I</td>
<td>30</td>
<td>0</td>
<td>12</td>
<td>8</td>
<td>y</td>
</tr>
<tr>
<td>20349-3-34</td>
<td>RML</td>
<td>PT</td>
<td>n/a</td>
<td>0</td>
<td>7</td>
<td>7</td>
<td>y</td>
</tr>
<tr>
<td></td>
<td>RUL</td>
<td>GGO</td>
<td>26</td>
<td>0</td>
<td>9</td>
<td>8</td>
<td>n</td>
</tr>
<tr>
<td>20349-3-35</td>
<td>RML</td>
<td>PT</td>
<td>n/a</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td>y</td>
</tr>
<tr>
<td></td>
<td>RUL</td>
<td>PT</td>
<td>n/a</td>
<td>0</td>
<td>7</td>
<td>6</td>
<td>y</td>
</tr>
<tr>
<td></td>
<td>LUL</td>
<td>PT</td>
<td>n/a</td>
<td>2</td>
<td>7</td>
<td>6</td>
<td>n</td>
</tr>
<tr>
<td>20349-3-37</td>
<td>RUL-1</td>
<td>N</td>
<td>20</td>
<td>3</td>
<td>7</td>
<td>5</td>
<td>y</td>
</tr>
<tr>
<td></td>
<td>RUL-2</td>
<td>LN</td>
<td>20</td>
<td>6</td>
<td>4</td>
<td>3</td>
<td>y</td>
</tr>
<tr>
<td>20349-3-38</td>
<td>RUL-1</td>
<td>O</td>
<td>40</td>
<td>0</td>
<td>8</td>
<td>6</td>
<td>y</td>
</tr>
<tr>
<td></td>
<td>RUL-2</td>
<td>O</td>
<td>15</td>
<td>6</td>
<td>8</td>
<td>6</td>
<td>y</td>
</tr>
<tr>
<td></td>
<td>LUL</td>
<td>O</td>
<td>16</td>
<td>4</td>
<td>13</td>
<td>10</td>
<td>y</td>
</tr>
<tr>
<td>20349-3-39</td>
<td>RLL</td>
<td>LN</td>
<td>15</td>
<td>2</td>
<td>5</td>
<td>5</td>
<td>y</td>
</tr>
<tr>
<td></td>
<td>LLL</td>
<td>PT</td>
<td>n/a</td>
<td>5</td>
<td>7</td>
<td>7</td>
<td>y</td>
</tr>
<tr>
<td>20349-3-40</td>
<td>RML-1</td>
<td>O</td>
<td>13</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td>y</td>
</tr>
<tr>
<td></td>
<td>RML-2</td>
<td>N</td>
<td>18</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>y</td>
</tr>
<tr>
<td>20349-3-41</td>
<td>RLL-1</td>
<td>GGO</td>
<td>24</td>
<td>4</td>
<td>13</td>
<td>11</td>
<td>y</td>
</tr>
<tr>
<td></td>
<td>RLL-2</td>
<td>GGO</td>
<td>9</td>
<td>6</td>
<td>12</td>
<td>8</td>
<td>y</td>
</tr>
<tr>
<td></td>
<td>RUL</td>
<td>PT</td>
<td>n/a</td>
<td>2</td>
<td>8</td>
<td>5</td>
<td>n</td>
</tr>
<tr>
<td>20349-3-42</td>
<td>RUL-1</td>
<td>O</td>
<td>10</td>
<td>2</td>
<td>8</td>
<td>8</td>
<td>y</td>
</tr>
<tr>
<td></td>
<td>RUL-2</td>
<td>O</td>
<td>30</td>
<td>0</td>
<td>9</td>
<td>9</td>
<td>y</td>
</tr>
</tbody>
</table>

Table 6.2. List of target diagnostic sites for the current study. Cases are numbered according to institutional review board protocol 20349-3. Site types are defined as follows: N = nodule (diameter < 3 cm); M = mass (diameter ≥ 3 cm); LN = lymph node; GGO = ground-glass opacity; I = infiltrate; PT = point target; O = other. The diameter of a site is defined to be its longest principal axis. The diameter of point targets is undefined as such sites corresponded to target locations rather than physical structures in the MDCT scan. Min. site distance is defined to be the minimum distance between the preplanned endobronchial route and a location on the surface of the target diagnostic site. VB image depth is defined to be the maximum bronchial generation to which VB guidance was available along the preplanned route. Guidance to a diagnostic site is deemed to be successful if a sample is obtained along the preplanned route.

The guidance success rate for the proposed system remained high despite the increased depth of the target peripheral sites. Samples were obtained along the preplanned route...


**Depth of VB Image Construction**

<table>
<thead>
<tr>
<th>Lobe</th>
<th>Current study</th>
<th>Previous study [90]</th>
<th>Significance</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>No.</td>
<td>Depth</td>
<td>No.</td>
</tr>
<tr>
<td>RUL</td>
<td>10</td>
<td>8.5±1.7</td>
<td>20</td>
</tr>
<tr>
<td>RML</td>
<td>2</td>
<td>7.0±0.0</td>
<td>7</td>
</tr>
<tr>
<td>RLL</td>
<td>5</td>
<td>11.8±1.3</td>
<td>23</td>
</tr>
<tr>
<td>LUL</td>
<td>7</td>
<td>10.0±2.6</td>
<td>21</td>
</tr>
<tr>
<td>LLL</td>
<td>2</td>
<td>9.5±3.5</td>
<td>14</td>
</tr>
<tr>
<td>Total</td>
<td>26</td>
<td>9.5±2.3</td>
<td>85</td>
</tr>
</tbody>
</table>

**Depth of Bronchoscope Insertion**

<table>
<thead>
<tr>
<th>Lobe</th>
<th>Current study</th>
<th>Previous study [90]</th>
<th>Significance</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>No.</td>
<td>Depth</td>
<td>No.</td>
</tr>
<tr>
<td>RUL</td>
<td>10</td>
<td>6.6±1.3</td>
<td>20</td>
</tr>
<tr>
<td>RML</td>
<td>2</td>
<td>7.0±0.0</td>
<td>7</td>
</tr>
<tr>
<td>RLL</td>
<td>5</td>
<td>9.2±3.4</td>
<td>23</td>
</tr>
<tr>
<td>LUL</td>
<td>7</td>
<td>8.1±2.2</td>
<td>21</td>
</tr>
<tr>
<td>LLL</td>
<td>2</td>
<td>7.5±0.7</td>
<td>14</td>
</tr>
<tr>
<td>Total</td>
<td>26</td>
<td>7.6±2.2</td>
<td>85</td>
</tr>
</tbody>
</table>

Table 6.3. Comparison of peripheral sites considered in the current study versus those from a previously-reported study [90]. The results from the current study include only the 26 peripheral sites for which ultrathin bronchoscopic procedures were planned. Depths are reported in terms of bronchial generations as mean ± standard deviation. Statistical significance was determined using the two-tailed version of Student’s t-test.

for 27 of the 31 total diagnostic sites (87%). The bronchoscope was successfully navigated to within two generations of the end of the preplanned route in two of the remaining four sites. One navigation failure (case 20349-3-35, LUL) was unrelated to the guidance system. Here, an appropriate sample site near the end of the preplanned route had been reached under system guidance, but patient coughing and dynamic airway collapse forced the bronchoscopist to retreat several generations. The bronchoscopist attempted to return to the site from memory, but sampled from an incorrect location. The remaining three failures were caused by a combination of difficulty in maneuvering the bronchoscope and
lack of communication between the bronchoscopist and guidance technician. These issues are addressed in the discussion section.

The average procedure length during the study was 38:47 ± 16:09 minutes (median: 37:00, range: 17:20 - 69:24). This is significantly longer than has been reported in other peripheral VB guidance studies. For example, Shinagawa et al. report an average procedure time of 24:30 ± 6:12 minutes (\( p < 0.001 \), Student’s t-test) [39]. However, previous studies have typically considered only a single diagnostic site per procedure. The current study considered an average of more than two sites per procedure. Table 6.4 breaks down the total procedure time for each case into time spent guiding to a target site and time required to sample the site. Overall, the average guidance time for each target site was 8:16 ± 7:70 minutes (median: 5:50, range: 0:59-32:35).

Several factors unrelated to the proposed guidance system contributed to the overall guidance time. We report anecdotal results from two cases with abnormally high guidance and total procedure times.

The longest guidance time for a target site in the study involved the left upper lobe nodule in case 20349-3-25. The nodule was located only eight bronchial generations deep into the airways, but required more than 30 minutes of guidance to reach. The majority of the guidance time involved a single maneuver, which required the bronchoscopist to insert the bronchoscope into a small orifice at an unusually sharp branching angle. The orifice, illustrated in Figure 6.7, led to aberrant segment incident upon the lingular bronchus and occurred only four generations into the preplanned route. As indicated in Figure 6.7(c), the route was clear to the bronchoscopist, but the maneuver required many attempts and much time. Furthermore, the patient’s airways exhibited significant bleeding throughout
<table>
<thead>
<tr>
<th>Case</th>
<th>Site loc.</th>
<th>Guidance time (min:sec)</th>
<th>Sample types</th>
<th>Sample time (min:sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>20349-3-24</td>
<td>LUL</td>
<td>5:38</td>
<td>BAL, brush</td>
<td>2:47</td>
</tr>
<tr>
<td>20349-3-25</td>
<td>RUL</td>
<td>2:21</td>
<td>brush</td>
<td>1:05</td>
</tr>
<tr>
<td>20349-3-26</td>
<td>LUL</td>
<td>10:47</td>
<td>BAL, brush</td>
<td>53:28</td>
</tr>
<tr>
<td>20349-3-28</td>
<td>RUL</td>
<td>6:37</td>
<td>BAL</td>
<td>9:57</td>
</tr>
<tr>
<td>20349-3-29</td>
<td>LUL-1</td>
<td>10:47</td>
<td>BAL, brush, forceps</td>
<td>53:28</td>
</tr>
<tr>
<td></td>
<td>LUL-2</td>
<td>2:33</td>
<td>BAL</td>
<td>0:38</td>
</tr>
<tr>
<td>20349-3-30</td>
<td>RUL</td>
<td>5:38</td>
<td>brush</td>
<td>10:17</td>
</tr>
<tr>
<td>20349-3-31</td>
<td>LUL</td>
<td>13:29</td>
<td>BAL</td>
<td>4:54</td>
</tr>
<tr>
<td>20349-3-32</td>
<td>RML</td>
<td>3:10</td>
<td>visual</td>
<td>0:35</td>
</tr>
<tr>
<td>20349-3-35</td>
<td>RUL</td>
<td>1:07</td>
<td>visual</td>
<td>0:35</td>
</tr>
<tr>
<td>20349-3-36</td>
<td>LUL</td>
<td>5:02</td>
<td>BAL</td>
<td>1:05</td>
</tr>
<tr>
<td>20349-3-37</td>
<td>RUL-1</td>
<td>10:21</td>
<td>forceps, BAL</td>
<td>18:37</td>
</tr>
<tr>
<td></td>
<td>RUL-2</td>
<td>14:34</td>
<td>TBNA</td>
<td>6:56</td>
</tr>
<tr>
<td>20349-3-38</td>
<td>RUL-1</td>
<td>12:32</td>
<td>brush, BAL</td>
<td>5:34</td>
</tr>
<tr>
<td></td>
<td>RUL-2</td>
<td>3:45</td>
<td>BAL</td>
<td>3:43</td>
</tr>
<tr>
<td></td>
<td>LUL</td>
<td>5:30</td>
<td>brush, BAL</td>
<td>4:04</td>
</tr>
<tr>
<td>20349-3-39</td>
<td>RLL</td>
<td>4:57</td>
<td>TBNA</td>
<td>6:02</td>
</tr>
<tr>
<td></td>
<td>LLL</td>
<td>0:59</td>
<td>visual</td>
<td>0:26</td>
</tr>
<tr>
<td>20349-3-40</td>
<td>RML-1</td>
<td>1:04</td>
<td>BAL</td>
<td>3:30</td>
</tr>
<tr>
<td></td>
<td>RML-2</td>
<td>3:41</td>
<td>TBNA</td>
<td>0:51</td>
</tr>
<tr>
<td>20349-3-41</td>
<td>RLL-1</td>
<td>2:00</td>
<td>BAL</td>
<td>2:39</td>
</tr>
<tr>
<td></td>
<td>RLL-2</td>
<td>8:10</td>
<td>BAL</td>
<td>5:26</td>
</tr>
<tr>
<td></td>
<td>RUL</td>
<td>5:50</td>
<td>BAL</td>
<td>2:44</td>
</tr>
<tr>
<td>20349-3-42</td>
<td>RUL-1</td>
<td>10:56</td>
<td>brush, BAL</td>
<td>4:42</td>
</tr>
<tr>
<td></td>
<td>RUL-2</td>
<td>4:52</td>
<td>brush, BAL</td>
<td>5:15</td>
</tr>
<tr>
<td>Average:</td>
<td>8:16 ± 7:10</td>
<td></td>
<td>7:03 ± 9:31</td>
<td></td>
</tr>
</tbody>
</table>

Table 6.4. Summary of guidance times, sample types, and sample times for the current study. Guidance time was defined as the length of the interval between the moment the technician and bronchoscopist meet at the first branching point of the pre-planned route to the moment the first diagnostic sample begins. For TBNA, sampling was defined to begin when the needle pierced the bronchial wall. For BAL and cytological brushings, sampling was defined to begin when the brush or saline wash first appeared in the bronchoscope’s field of view. Sample time included the entire duration of the sampling process as well as any subsequent clearing of the airways.

Case 20349-3-29 represented the longest procedure at 69 minutes. The case contained a small peripheral nodule located 13 generations deep into the tree, but was selected for the
study primarily because it exhibited a focal mass that occurred very near the pulmonary artery (PA). As TBNA was indicated for this case, the task of the guidance system was to aid the physician in aspirating the mass without puncturing the PA. Figure 6.8 illustrates the diagnostic sites for the case (green and blue) as well as the portion of the PA (red) bordering the target mass. Guidance to the predetermined sample location required less than a minute. The remainder of the guidance time was spent maneuvering the bronchoscope into a position from which the target mass could be sampled safely. Each sample location was verified using the final site visualization illustrated in Figure 6.3. As both a target site (the mass) and an obstacle (the PA) were present, the system alternated between displaying the target (in green) and obstacle (in red) until only target tissue was located in the bronchoscope’s field of view. Physically maneuvering the bronchoscope into such a position proved to be difficult and time-consuming, but the bronchoscopist eventually aspirated the mass without puncturing the PA.

6.3 Discussion

The technological advancements of thin-slice MDCT scanners and ultrathin videobronchoscopes have combined to make VB guidance, previously feasible only for central-chest sites, a promising option for peripheral bronchoscopy. Thin-slice scanners enable the detection and visualization of small airways, while ultrathin bronchoscopes allow bronchoscopists to visit peripheral sites under direct vision. Indeed, several encouraging results have already been obtained for VB-guided peripheral bronchoscopy [35, 36, 39, 88, 89].

Peripheral VB guidance is appealing, in large part, for its elegant simplicity. VB guid-
Figure 6.7. A difficult maneuver in case 20349-3-25. The bronchoscopist was required to make a sharp turn through a small orifice to approach the left upper lobe nodule in this case. The nodule was located in an aberrant segment incident upon the lingular bronchus. (a) A global rendering of the airway tree. The blue line represents the preplanned route and the target nodule is rendered in green. The arrow indicates the location of the difficult turn. (b) A close-up view of the tree from (a). The arrow again indicates the location of the difficult turn. (b) A bronchoscopic video frame and corresponding VB rendering depicting the turn. As indicated by the blue line, the bronchoscopist must enter the top bronchus.

ance information can be derived solely from the same pre-operative MDCT scan used to identify target diagnostic sites without the need for complex or expensive navigational devices. The simplicity of VB guidance is especially apparent in the bronchoscopy suite, as VB guidance requires minimal setup and is immediately available at the start of the procedure. The full promise of peripheral VB guidance, however, has yet to be realized. The successful
Figure 6.8. A global airway rendering for the case with the longest overall procedure time in the study (case 20349-3-29). The green region represents a target mass in the left upper lobe. The red region represents a section of the pulmonary artery (PA) that bordered the mass. The blue object at the top of the figure (in the patient’s left upper lobe) is a peripheral nodule. The procedure was lengthy because the bronchoscopist needed to take carefully aspirate the mass to avoid contacting the PA.

early studies of Asano-Shinagawa, for example, required significant pre-operative physician effort to select an appropriate route to the target site and generate informative VB views along the route [36, 39, 88]. Intraoperatively, the system required similar intervention to keep the real bronchoscopic video and VB rendered views aligned. Our system aims to build upon these pioneering efforts and apply medical-image analysis and machine-vision techniques to help peripheral VB guidance achieve its full potential.

The system relies heavily on automatic procedure-planning methods for extracting patient-specific 3D airway models and defining appropriate endobronchial routes from the MDCT data without requiring physician interaction [21, 40, 42, 172]. Such automated methods make it possible to extract and visualize the entire airway tree, thus enabling intraop-
erative 3D visualization of the bronchoscope’s global location within the airways as seen in Figure 6.2. There are, however, dangers associated with complete reliance on automatic computerized medical-image analysis. The system therefore incorporates a simple graphical user interface (GUI), which enables a technician to quickly verify and, if necessary, correct, the automatically-planned route. Details of the entire procedure-planning process have already been presented [21, 57, 172]. Furthermore, we are aware of the danger of divorcing the physician entirely from the procedure-planning process and have therefore developed an automated reporting mechanism enabling web-based interactive previews of the entire preplanned procedure [94].

The current study has demonstrated the overall effectiveness of the proposed procedure-planning methodology for peripheral bronchoscopy. In fact, the study has shown that VB guidance information can be made available an average of more than two full bronchial generations deeper than has been previously reported [90]. This increase is partly attributable to selection bias, as the study preferentially enrolled patients presenting with distant peripheral lesions that would challenge the limits of VB guidance. The increase is still important, however, as it represents a new upper-bound on the potential depth to which VB guidance can be effectively employed. Furthermore, our procedure-planning methodologies may prove useful for other guidance techniques, such as CT-fluoroscopy and electromagnetic probe navigation [65, 70, 73].

The current study has also demonstrated the efficacy of the live-guidance component of our system in leading the bronchoscopist down a preplanned endobronchial route. The system’s automated method for synchronizing the VB view with the live bronchoscopic video feed was tested successfully in several patients. This represents the first live peripheral
human trial of such a method. The visualization and synchronization capabilities of the system were most useful at the end of the planned endobronchial routes, where the system provided the bronchoscopist with unambiguous graphical and distance information. These visualization capabilities proved to be quite flexible, enabling visualization of both a target diagnostic site and a vascular obstacle in two cases. Finally, the live-guidance system is easy-to-use. A total of five different technicians operated the system over the fifteen cases considered in this study. Two of the technicians had minimal experience operating the system before their first live procedure.

Extensive retrospective analysis of the procedure video has suggested that the four guidance failures encountered during the study were both readily detectable and potentially preventable. The guidance failures each occurred at bronchial branchpoints requiring the bronchoscopist to preform a difficult blind maneuver. In these cases, the bronchoscopist clearly knew the correct branch to enter, but could not pass through the bronchial opening without backing up and approaching the branchpoint with momentum. Direct vision was obscured during the maneuver, however, and the bronchoscopist entered an incorrect opening.

It is typically easy to discern such events, as the bronchoscopic video is no longer in good agreement with the model. In the failure cases, however, the model was similar enough to the video view to be confusing and the physician requested the technician to align the VB and video views. Here, the technician, having no tactile feedback from the bronchoscope, had no choice but to trust the bronchoscopist’s judgement and adjust the VB view accordingly. As the bronchoscope was advanced deeper into the airways, however, it became clear that the bronchoscopic view and VB model could no longer be synchronized. The technician,
reluctant to correct the experienced bronchoscopist, would continue to adjust the VB view to best match the bronchoscopic video and the bronchoscopist, seeing the adjusted views, would remain unaware of the lack of synchrony and sample in an incorrect location.

An obvious remedy to such guidance failures would be to remove the technician entirely from the bronchoscopic suite and give the physician full control over the guidance system. The physician would then be immediately aware of any lack of synchrony between the observed video and the VB model. This is an ongoing development focus for our system.

Overall, this study has demonstrated that image-based VB guidance of peripheral bronchoscopy is both safe and feasible in humans, even when significant aspects of preoperative procedure planning and intraoperative bronchoscopic guidance are controlled by automatic computerized methods. Thus, the study has helped to pave the way for practical and widespread application of peripheral VB guidance. Future randomized outcome studies will help to clarify the practical benefit of both our system and VB guidance in general.
This thesis has presented robust methods for both human airway-tree segmentation and anatomical-tree matching. This chapter summarizes the novel contributions and practical impact of this work and discusses potential directions for future research.

The automatic and interactive segmentation methods described in Chapter 3 enable the reliable extraction of a full global airway tree and ensure the accuracy of critical routes through the airways that are necessary for image-based peripheral bronchoscopic guidance. In particular, ground-truth comparisons have demonstrated that the proposed automatic algorithm extracts substantially more peripheral airways than existing algorithms while producing very few false-positive branches. Furthermore, the interactive methods enable the easy extraction of any visible airway in an MDCT scan, no matter how weak its image signature.

The automatic and interactive methods have been combined in an integrated segmentation system. Segmentations produced by this system have enabled, for the first time, human trials of the Virtual Navigator system for peripheral bronchoscopic guidance. The
results of these trials, presented in Chapter [6] demonstrated the safety and feasibility of VB-guided peripheral bronchoscopy, even when significant aspects of pre-operative procedure planning and intraoperative bronchoscopic guidance are controlled by automatic computer algorithms. Furthermore, the trials represented the most challenging test to date of a VB guidance system, considering target diagnostic sites an average of two full bronchial generations deeper into the airway tree than had been previously studied. The success of these trials was due, in large part, to the representational accuracy of patient-specific 3D airway models derived from segmented airways produced by the proposed system.

Human trials have provided direct validation for the proposed segmentation system by comparing segmented airways directly to a patient’s physical anatomy. The system has also been carefully validated on a large number of real MDCT chest scans on human patients for which peripheral bronchoscopy was not performed. The system has demonstrated robust performance on these images, producing good results for scans of both healthy and diseased airways obtained from several different sources under a number of different protocols. Further validation is always necessary, however, as human data exhibit a remarkable amount of variability. Future cases will likely suggest the need both for alterations to the automatic segmentation algorithm and for new interactive tools and techniques.

The efficiency with which the proposed system produces segmentations for image-guided peripheral bronchoscopy has proven to be vital for clinical application. The automatic algorithm runs on current hardware in less than three minutes on average. The interactive refinement of a peripheral bronchoscopic route can be performed by a trained technician in just a few minutes more. The system therefore requires no physician interaction.

While the system is already efficient enough for clinical use, future research should focus
on improving the speed and usability of the interactive toolkit. Most of the technician’s interactive processing time involves scanning the proposed endobronchial route for small terminal branches missed by the automatic algorithm. An interesting research problem would therefore be the development of an algorithm for “suggesting” the locations of missing branches along a route. The technician’s task could therefore be limited to accepting or rejecting each proposed branch location. Such an algorithm could be advantageously tuned to achieve high sensitivity at the expense of specificity, as the technician would be sure to reject obvious false branches. The algorithm would also need only search the small portion of the image volume that is directly adjacent to the proposed route, and could therefore employ more sophisticated (and computationally expensive) processing methods than would be feasible for a global segmentation algorithm.

This thesis has also introduced a number of novel techniques for 3D medical-image segmentation. In particular, the automatic airway-segmentation algorithm provides a unified approach to 3D tree segmentation by integrating low-level image cues into anatomically-relevant branch-level information, connecting and constraining branches using an analytic 3D surface model, and postponing important branch/no-branch decisions until global optimization using information from the entire volume can be performed.

In the future, the techniques this thesis has used to segment the airway tree could be profitably applied to tree-segmentation problems involving the lung, heart, or liver vasculature. As in the airway-segmentation problem, such applications involve the detection of slowly-varying tubular branches. Many of the techniques introduced in Chapter 3 could be applied with little modification. The airway section filter, for example, which searches for low-intensity air voxels surrounded by high-intensity wall voxels, could easily become a
“vessel section” filter searching for high-intensity vessel voxels surrounded by air.

Chapter 4 introduced a novel DP tree-matching framework. The framework facilitated the development of a robust anatomical tree-matching algorithm, capable of tolerating common topological errors introduced during tree extraction and efficiently locating a globally-optimal match with respect to a rich geometric cost function. The proposed matching algorithm has been shown to automatically generate matches that agree well with ground-truth hand matches produced by a human observer. In particular, the proposed algorithm was shown to increase an important measure of matching accuracy to more than 96% from the 57% achieved by the best existing method for the same human airway-tree data.

The tree-matching framework developed for this thesis could potentially be useful in more general graph-matching problems. In the anatomic domain, such work could potentially be useful in handling branching structures with graph-theoretic cycles, such as the vasculature of the brain. It is highly unlikely that more general graph structures could be accommodated without sacrificing optimality, as even simple matching problems, such as determining whether two graphs are isomorphic, are not known to be efficiently solvable when the graphs are not constrained to be trees. The optimal DP framework developed in this framework could, however, find use as a subroutine of a more complex algorithm.

Chapter 5 considered the problem of labeling human airway-tree anatomy, and described both a tool for collecting a database of hand-labeled example trees and an automatic labeling algorithm. The automatic algorithm efficiently labeled an arbitrary input tree by matching it to labeled model tree, and tolerated a number of common anatomical variations by employing the robust matching framework of Chapter 4. The most straightforward method for accurately labeling more severe anatomical variations is to extend the lobar-
subtree matching algorithm of Section 5.2.3 by matching each unlabeled lobar subtree to several different labeled lobar subtrees covering the expected range of human anatomy then assigning labels based on the maximum-similarity match. The main barrier to implementing this approach is data collection. Many hand-labeled airway trees would be required to develop and validate such an algorithm, as examples of each variation would be needed in both the model and the validation set.

Finally, this thesis concludes with a list of current and expected publications by the author. The publications are organized by topic:

1. Human airway-tree segmentation - [172,173]

2. Image-guided bronchoscopy - [38,57,94,95,174–176]

3. Anatomical-tree matching and labeling - [177–179]

4. Statistical pattern recognition - [180,181]
Appendix A

Interpolated Surface Definition

This appendix describes a method for constructing a smooth interpolated surface between two airway sections described by 3D ellipses per (3.11). Represent the two ellipses by the quadruples $E_i = (n_i, P_i(n_i), c_i, M_i), i = 1, 2$, where $n_i$ is a 3D unit normal vector, $P_i(n_i)$ is a $2 \times 3$ orthonormal projection matrix for the plane normal to $n_i$, $c_i$ is the 3D location of the ellipse center, and $M_i$ is a non-singular $2 \times 2$ matrix. We assume that $\det(M_i)$ is positive; i.e., that (3.11) traces $E_i$ in a counter-clockwise direction when viewed along $-n_i$.

This assumption involves no loss of generality, however, as replacing $M_i$ with

$$M_i \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \quad (A.1)$$

reverses the sign of $\det(M_i)$ without altering the resulting ellipse.

In general, $n_1$ and $n_2$ do not define parallel planes. Let $a + \lambda \cdot b$, with $\|b\| = 1$ and $\lambda \in (-\infty, \infty)$, trace the line of intersection between the two planes. The interpolated

---

1Replacing $M_i$ with (A.1) is equivalent to a reparameterization of (3.11), in which $\theta$ is replaced by $-\theta$ and $E_i$ is traced out in the reverse direction.
surface is defined in a cylindrical coordinate system centered about this line of intersection, with coordinates \((ρ, h, φ)\) as in Fig. A.1(a). The origin of the cylindrical coordinate system is \(a\) and the frame is oriented so \(\hat{h}\) points along \(b\), \(\hat{ρ}\) points along \((\bar{c}_1 - a) - b \cdot b^T(\bar{c}_1 - a)\), and the \(φ\) coordinate of \(\bar{c}_1\) is zero.

Both \(E_1\) and \(E_2\) lie in \(ρ−h\) planes with constant \(φ\) coordinate in this coordinate system. Let \(φ_1 = 0\) and \(φ_2\) be the coordinates of these planes. There exist \(2 \times 3\) projection matrices \(P^ρ/ h_1\) and \(P^ρ/ h_2\) such that \(E_1\) and \(E_2\) can be described in the cylindrical coordinate system by

\[
\begin{pmatrix}
ρ \\
h \\
φ
\end{pmatrix} = \begin{pmatrix}
P^ρ/ h_i \left[ P_i (n_i)^T \bar{M}_i \theta + \bar{c}_i - a \right] \\
φ_i
\end{pmatrix}, \quad i = 1, 2. \tag{A.2}
\]

Thus, \(E_1\) and \(E_2\) are defined by a constant \(φ\) coordinate and a 2D ellipse in the \(ρ−h\) plane traced out by \(\tilde{M}_i \theta + \tilde{c}_i\) with \(\tilde{M}_i = P^ρ/ h_i P_i (n_i)^T \bar{M}_i\) and \(\tilde{c}_i = P^ρ/ h_i (\bar{c}_i - a)\).

The interpolated surface is constructed by sweeping the plane containing \(E_1\) towards the plane containing \(E_2\) around the \(\hat{h}\) axis while smoothly morphing \(E_1\) into \(E_2\) in the \(ρ−h\) plane. For each \(α \in [0, 1]\), the interpolated surface is characterized by the 2D ellipse given by

\[
\begin{pmatrix}
ρ \\
h \\
φ
\end{pmatrix} = \begin{pmatrix}
[ (1 - α)\tilde{M}_1 + α\tilde{M}_2 R^* ] \theta + [ (1 - α)\tilde{c}_1 + α\tilde{c}_2 ] \\
(1 - α)φ_1 + αφ_2
\end{pmatrix}, \tag{A.3}
\]

where \(R^*\) is the \(2 \times 2\) rotation matrix satisfying

\[
R^* = \arg \min_R \int_{\theta=0}^{2π} \left\| (\tilde{M}_1 \theta + \tilde{c}_1) - (\tilde{M}_2 R \theta + \tilde{c}_2) \right\|^2. \tag{A.4}
\]
Note that both $\tilde{M}_2$ and $\tilde{M}_2R$ trace out the same ellipse for any rotation matrix $R$. Specifically, the transformation $\theta \mapsto R\theta$ simply reparameterizes the ellipse, replacing $\theta$ with $\theta + \psi$ for some rotation angle $\psi$. Thus, (A.4) determines an optimal “alignment” between $E_1$ and $E_2$ that minimizes the integrated squared distance between associated points on the two ellipses. The final interpolated surface is obtained by converting (A.3) back into the image’s Cartesian coordinate system.

![Diagram](image-link)  

**Figure A.1.** Constructing a smooth interpolated surface between two airway sections. (a) The cylindrical coordinate system used for the construction. (b) The ellipses $E_1$ and $E_2$ lie in constant $\phi$ planes. The interpolated surface is constructed by sweeping the plane containing $E_1$ towards the plane containing $E_2$ while simultaneously morphing $E_1$ into $E_2$. It is useful to picture a door swinging on a hinge.
The main result of this appendix is Theorem B.1, which asserts two key properties about the subtree $t^*$ returned by Algorithm 3.1. Two lemmas are required. Recall that a relaxed subtree of a tree $T$ on $V$ vertices is a vector $f \in [0,1]^V$ satisfying $f_0 = 1$ and $f_k \leq f_{P[k]}$ for all $k > 0$, where $P[k]$ represents the parent of $k$ in $T$. Similarly, a non-relaxed subtree (or simply a subtree) $t$ is a binary vector satisfying $t_0 = 1$ and $t_k = 1 \Rightarrow t_{P[k]} = 1$.

**Lemma B.1** Consider a tree $T$ on $V$ vertices and a relaxed subtree $f$ of $T$. Then $f$ can be written as a weighted sum of non-relaxed subtrees of $T$:

$$f = \sum_{n=1}^{N} w^{(n)} t^{(n)},$$

where $N \leq V$ and the $w^{(n)}$ are positive weights summing to one. Furthermore, the cost and benefit of $f$ per (3.16) and (3.17) can be written as

$$B(f) = \sum_{n=1}^{N} w^{(n)} B(t^{(n)}) \quad \text{and} \quad C(f) = \sum_{n=1}^{N} w^{(n)} C(t^{(n)}).$$

(B.2)
Proof. Recursively define \( g^{(n)} = (g_1^{(n)}, g_2^{(n)}, \ldots, g_{V-1}^{(n)}) \), \( t^{(n)} \in \{0, 1\}^V \), and \( w^{(n)} \in [0, 1] \) as follows:

\[
g^{(1)}_k = f_k \\
\begin{cases} 
1, & \text{if } g^{(1)}_k > 0 \\
0, & \text{otherwise}
\end{cases} \\
\begin{cases} 
w^{(1)} = \min_k \{g^{(1)}_k : t^{(1)}_k = 1\} \\
0, & \text{otherwise}
\end{cases}
\]

\[
g^{(n)}_k = g^{(n-1)}_k - w^{(n-1)} t^{(n-1)}_k \\
\begin{cases} 
1, & \text{if } g^{(n)}_k > 0 \\
0, & \text{otherwise}
\end{cases} \\
\begin{cases} 
w^{(n)} = \min_k \{g^{(n)}_k : t^{(n)}_k = 1\} \\
0, & \text{otherwise}
\end{cases}
\]

Let \( N = \max\{n \text{ such that } w^{(n)} > 0\} \). We have \( N \leq V \) as each \( g^{(n)} \) contains at least one more zero entry than \( g^{(n-1)} \). The \( t^{(n)} \) are non-relaxed subtrees by construction. Equation (B.1) is verified by expanding \( f \) via

\[
f = g^{(1)} = g^{(2)} + w^{(1)} t^{(1)} = g^{(3)} + w^{(2)} t^{(2)} + w^{(1)} t^{(1)} = \ldots = g^{(N+1)} + w^{(N)} t^{(N)} + \ldots + w^{(1)} t^{(1)},
\]

and noting that \( g^{(N+1)} \) is the zero vector. The \( w^{(n)} \) are positive by construction and must sum to one because \( f_0 = 1 \) by the definition of a relaxed subtree. Finally, (B.2) follows from (B.1) as both \( B(f) \) and \( C(f) \) are linear functions.

Fig. B.1 gives an illustrative example of Lemma B.1.

Lemma B.2 Consider a tree \( T \) with non-negative costs and benefits \( \{c_k\} \) and \( \{b_k\} \) and a fixed constant \( r > 0 \). Let \( t^* \) be the output of Algorithm 3.1 i.e.,

\[
t^* = \arg\max\{B(t) - rC(t) \text{ such that } t \text{ is a non-relaxed subtree of } T\}. \tag{B.3}
\]
Figure B.1. Illustration of Lemma B.1 for the relaxed subtree $f = (1 \ 0.75 \ 0.6 \ 0.3 \ 0.3)^T$. Here, $f$ is equal to the weighted sum of five non-relaxed subtrees. For example, in the notation of the lemma, $w^{(2)} = 0.1$ and $t^{(2)} = (1 \ 1 \ 1 \ 0 \ 1)^T$.

Then there is no relaxed subtree $f$ of $T$ satisfying $B(f) - rC(f) > B(t^*) - rC(t^*)$.

Proof. Suppose such an $f$ existed. Then

$$B(f) - rC(f) > B(t^*) - rC(t^*) \Rightarrow \sum_{n=1}^{N} w^{(n)} [B(t^{(n)}) - rC(t^{(n)})] > B(t^*) - rC(t^*) \Rightarrow$$

$$\max_{n=1}^{N} \left\{ B(t^{(n)}) - rC(t^{(n)}) \right\} > B(t^*) - rC(t^*). \tag{B.4}$$

Inequality (B.4) follows immediately from Lemma B.1. Inequality (B.5) also follows from Lemma B.1, as the $w^{(n)}$ are positive and sum to one. But (B.5) contradicts our assumption that $t^*$ maximizes $B(t) - rC(t)$ over non-relaxed subtrees, as at least one of the $t^{(n)}$ must be a non-relaxed subtree achieving a higher value. □

Theorem B.1 Consider a tree $T$ with non-negative costs and benefits $\{c_k\}$ and $\{b_k\}$ and a fixed constant $r > 0$. Let $t^*$ be the output of Algorithm 3.1. Then:

1. The pair $(C(t^*), B(t^*))$ lies on both the TKP and RTKP curves defined in (3.18) and (3.20).
2. The slope of the RTKP curve, where defined, is bounded from below by \( r \) for all \( \gamma < C(t^*) \) and from above by \( r \) for all \( \gamma > C(t^*) \).

Proof.

1. As the RTKP curve upper-bounds the TKP curve and \( t^* \) is a non-relaxed subtree of \( T \), it suffices to show that \((C(t^*), B(t^*))\) lies on the RTKP curve. Suppose, to elicit a contradiction, that \((C(t^*), B(t^*))\) does not lie on the RTKP curve. Then there exists a relaxed subtree \( f \) of \( T \) satisfying both \( C(f) \leq C(t^*) \) and \( B(f) > B(t^*) \). As \( r > 0 \), this implies \( B(f) - rC(f) > B(t^*) - rC(t^*) \), which contradicts Lemma [B.2].

2. The RTKP curve represents the solution to a parametric linear programming problem and is therefore piecewise linear with non-increasing slope \([161,162]\). Consider the left and right derivatives of \( \beta_R(\gamma) \), which exist for all \( \gamma \) and are non-increasing. It suffices to show that \( r \) provides a lower bound for the left derivative of \( \beta_R \) at \( C(t^*) \) and an upper bound for the right derivative of \( \beta_R \) at \( C(t^*) \) by demonstrating that no point on the RTKP curve can lie above the line passing through the point \((C(t^*), B(t^*))\) with slope \( r \).

Suppose a relaxed subtree \( f \) of \( T \) achieving such a point \((C(f), B(f))\) exists. If \( C(f) < C(t^*) \) the slope of the line connecting \((C(f), B(f))\) and \((C(t^*), B(t^*))\) must be less than \( r \), but

\[
\frac{B(t^*) - B(f)}{C(t^*) - C(f)} < r \Rightarrow B(t^*) - rC(t^*) < B(f) - rC(f), \tag{B.6}
\]

which contradicts Lemma [B.2]. A symmetric argument applies when \( C(f) > C(t^*) \).
Additional Segmentation Results

This appendix provides full visual results for the gold-standard comparisons summarized in Table 3.2. The proposed algorithm produces the most accurate segmentation in each case. The second-best segmentations vary from case to case, however. The morphological algorithm is second for case 21405-3a, the hybrid algorithm is second for case 20349-3-39, and the adaptive region-growing algorithm is second for case h002-tlc.
Figure C.1. Segmentation results used for gold-standard comparison (case 21405-3a).
Figure C.2. Segmentation results used for gold-standard comparison (case 20349-3-39).
Figure C.3. Segmentation results used for gold-standard comparison (case h002-tlc).
User Manual for the SegTool

This appendix describes the usage of the SegTool, an integrated airway-tree segmentation system that combines a robust and efficient automatic algorithm with a versatile interactive segmentation toolkit. The tool is a stand-alone Microsoft Foundation Class (MFC)-based dialog application. Figure D.1 depicts the Microsoft Windows icon used to invoke the tool. The tool takes as input a VN case study consisting of a pointer to a 3D MDCT chest scan, the location of a root site in the proximal trachea of the airway tree depicted in the scan, and an optional collection of physician-indicated ROIs. Once an appropriate VN case study has been loaded, the user has two options:

1. Generate an automatic airway-tree segmentation

2. Interactively clean and extend a previously-generated segmentation

These options correspond to the automatic segmentation algorithm and interactive segmentation toolkits described in Sections 3.1 and 3.2 of this thesis.
The manual is organized as follows. Section D.1 describes the steps necessary to generate a global airway-tree segmentation using the automatic algorithm. Section D.2 describes the steps necessary to edit the automatically-segmented result. The manual will follow a “cookbook” format and follow the generation of a complete segmentation suitable for image-based guidance to a peripheral ROI for a single case (20349-3-25).

Figure D.1. Windows icon for the SegTool.

D.1 Automatic Algorithm

At system startup, only two buttons are enabled in the SegTool. The first, illustrated in Figure D.2 invokes a file dialog prompting the user to load a VN case study. The second, highlighted by the life-preserver icon illustrated in Figure D.3 brings up a help dialog. The help dialog is context-sensitive, and informs the user of his available options within the system at any time. Figure D.4 gives an example of the help dialog at system startup.

Figure D.2. SegTool button for loading a VN case study.

Figure D.3. Help icon.

Once the user has loaded a case study, the “Auto Seg.” button in the upper-right of the view is enabled. Pressing this button begins the automatic segmentation algorithm,
which begins with the generation of a conservative segmentation by adaptive region-growing. Because the region-growing algorithm may fail in a small fraction of cases, the system allows the user to preview the conservative segmentation and, if necessary, adjust the region-growing threshold. Figure D.5 gives an illustrative example of this process.

Once the user has verified and, if necessary, corrected the conservative segmentation, the
algorithm generates an approximate lung mask (the set of voxels inside the lungs). From the lung mask, the algorithm extracts a 3D bounding box for the lung volume. Voxel
outside the bounding box will not be processed in subsequent steps of the algorithm. The goal here is efficiency. The bounding box can significantly reduce the number of voxels for the algorithm to process, especially on whole-body scans where the lungs occupy only a small fraction of the image volume. Here again, the automatically generated bounding box is almost always acceptable, but the system allows the user to refine the result if necessary.

Figure D.6 illustrates the bounding-box refinement process. The user is presented with transverse (top-left), coronal (top-right), and sagittal (lower-right) orthogonal viewing slices with the appropriate cross-section of the bounding box superimposed in red. The user can change the position of the orthogonal slices by left-clicking in the appropriate pane, then scrolling the mouse wheel or pressing the page-up or page-down key. The bounding box can be resized by right-clicking and dragging. The goal is for the bounding box to enclose the entire lung volume with little wasted space. Care must be taken, however, as a bounding box that looks good in one slice may be too small in another. It is preferable to be conservative and leave some extra space on all sides.

Once the lung volume bounding box has been approved, the automatic algorithm runs its course. This typically requires 1-3 minutes on modern (2008) hardware. Once the algorithm has completed, the user is presented with a voxel-level rendering of the result, as illustrated in Figure D.7. The automatic result is accompanied by a “Verify automatic segmentation” dialog. This dialog contains the first set of interactive tree-editing tools available to the user, and is discussed in Section D.2.2. To accept the automatically generated result without editing, the user should click the “accept and save” button. This brings up a standard Windows file dialog. The default extension for SegTool files is .seg. The .seg file provides basic information about the image and voxel dimensions and a list of voxels belonging to
Figure D.6. Verifying the lung volume bounding-box in the SegTool. (a) The user is presented with the automatically generated lung volume bounding box. The automatically generated result is nearly-always good enough and can be accepted by pressing “OK” (b) The user can scroll through orthogonal slices using the mouse wheel and adjust the bounding box by right-clicking and dragging. Here, the user has scrolled the top-left transverse slice towards the top of the lungs and tightened-up the bounding box. (c) Tightening the bounding box to a single slice is generally a bad idea, however, as this image indicates. Here, the bounding box from (b) is far too tight for a transverse slice further down in the lungs. It is best to leave the automatically generated bounding box “as is” unless it obviously too large or omits part of the lung volume.

the segmentation in a plain-text format, as illustrated in Figure D.8.
Figure D.7. The automatic segmentation algorithm output in the SegTool. (a) The automatic segmentation. (b) The “Verify automatic segmentation” dialog that accompanies the automatically-segmented result. The controls in this dialog actually belong to SegTool’s interactive toolkit and are discussed in Section D.2.2. To accept the automatic result, the user should click the button labeled “Accept and Save.”
D.2 Interactive Toolkit

The user manual for the SegTool’s interactive tree-editing toolkit is organized as follows. First, Section D.2.1 describes controls for visualizing the MDCT image data and physician-indicated ROIs. These controls are common to all interactive tools. Then, Sections D.2.2 through D.2.5 describe the three basic tree-editing modes containing the interactive tools.

D.2.1 Oblique Slice and ROI Dialog Controls

All interactive tree editing modes enable user interaction with the segmented airways in conjunction with oblique slices through the raw MDCT image data and the physician-indicated ROIs associated with the VN case study. Application controls for the oblique slice and ROIs can be summoned by clicking on the buttons labeled “Slice” and “ROI” in the upper-right portion of the main window. Figure D.9 illustrates the icons associated
with these buttons.

Pressing the “Slice” button summons the “Oblique slice display parameters” dialog illustrated in Figure D.10(a). The user can turn on the oblique slice data by checking the “Draw oblique slice” box, control the slice size using the slider, and center the view about an arbitrary image voxel by typing in its x,y,z coordinates and pressing “Jump.” Pressing the “HU Window” summons the dialog in Figure D.10(b). With this dialog, the user can choose an arbitrary HU windowing function, or chose from a list of pre-set window. Figure D.11 illustrates the effects of the various slice controls.

![Slice ROI](image)

**Figure D.9.** Oblique slice and ROI icons in the SegTool.

Pressing the “ROI” button summons the “ROI display dialog” illustrated in Figure D.12. This dialog contains controls enabling the user to determine which of the ROIs associated with the case study are displayed and which are hidden from view. This option is especially useful for cases in which major vasculature such as the aorta or pulmonary artery have been defined. Such ROIs can be useful in many tools, but they also take up a lot of room. The “Locate ROI” button centers the view on the center of mass of the currently highlighted ROI.

### D.2.2 Verify Automatic Segmentation Editing Mode

The *Verify Automatic Segmentation* editing mode is the user’s first opportunity to add or remove branches from an automatically-segmented tree. It also provides insight into the operation of the automatic algorithm. The mode is initialized with the “Classic” voxel-based
Figure D.10. The oblique slice and HU window dialogs.

To edit the tree, the user must uncheck the “Classic” view option to bring up the surface-based representation used internally by the automatic segmentation algorithm. This representation is illustrated in Figure D.13. Here, the user can view the conservative segmentation and all of the branch segments that were located by the algorithm. Branch segments are differentially colored by whether they touch the conservative segmentation (green), belong to the automatic segmentation but do not touch the conservative segmentation (yellow), can be connected to the automatic segmentation but were omitted by the
Figure D.11. Effects of the SegTool oblique slice controls. (a) The default oblique slice appearance (b) Increased slice size (c) HU window set to “mediastinal”
The ROI display dialog.

The user has three main editing options. First, the user can adjust the global cost/benefit ratio threshold used by the graph-partitioning algorithm. Raising the threshold removes branches from the tree and lowering the threshold adds branches to the tree. Second, the user can select an automatically segmented (yellow) branch, and remove it and all its descendants from the tree. This option is illustrated in Figure D.14. Finally, the user can add connected branches that were erroneously omitted by the graph-partitioning algorithm. Figure D.15 illustrates this process. Note that the oblique slice is very helpful in determining whether or not a particular branch really belongs in the tree. Once the user has made any desired changes to the automatic segmentation, the mode can be exited by pressing the “Accept and save” button.
Figure D.13. Visualization options in the Verify Automatic Segmentation editing mode. (a) The voxels of the conservative segmentation are drawn. Tree branches touching the conservative segmentation are colored green. Tree branches not touching the conservative segmentation are colored yellow. Interpolated connection surfaces are colored orange. (b) The conservative segmentation has been reduced to its approximate centerline structure to remove visual clutter. (c) Branch segments that can be connected to the tree but were rejected by the graph-partitioning algorithm are colored blue. Branch segments that are far from the tree and could not be connected are colored red.
Figure D.14. Subtree deletion in the *Verify Automatic Segmentation* editing mode. (a) A perfectly good subtree. (b) The user highlights the subtree by right-clicking. (c) The subtree is deleted with a single button press.
Figure D.15. Branch addition in the Verify Automatic Segmentation editing mode (a) The blue branch was erroneously omitted by the graph partitioning algorithm. It is a short leaf branch separated from the rest of the tree by a strong blockage and therefore has very high cost, relatively low benefit, and spans no subtree of supporting branches. (b) The user highlights the branch by right-clicking. The branch’s connection to the rest of the tree is also highlighted. (c) The subtree is added with a single button press.
D.2.3 Visualize Segmentation Mode

Once the user has either generated and accepted an automatic segmentation, edited an automatically generated segmentation, or loaded a previously-generated segmentation, the system enters the Visualize Segmentation mode. This mode is read-only (it does not contain any editing tools) but it enables the user to view the current segmented airways in conjunction with the oblique slice and ROI data. The user can enter one of the “true” editing modes at any time by selecting from the drop-down menu labeled “Current tree-editing mode” at the top of the main application window.

D.2.4 Clean Segmentation Editing Mode

The Clean Segmentation editing mode lets the user quickly and easily delete voxels from a tree. The user simply clicks on a location in the tree to highlight that location and all the voxels located beneath that location in the tree. The highlighted voxels are removed with a single button press. The editing mode also contains an “undo” function, as illustrated in Figure D.16. The mode can also be canceled at any time and the tree will be restored to its original state. The “Accept + Exit” button commits the changes made by the user and returns the SegTool back to the Visualize Segmentation mode.

D.2.5 Livewire Editing Mode

The Livewire editing mode is, by far, the most important interactive mode of the SegTool. This mode enables the user to interactively define single-voxel thick branches that may be missed by the automatic algorithm. This section describes the use of the livewire tool in ensuring that the segmented airways capture all branches necessary for guidance to a
Figure D.16. An illustration of the Clean Tree editing mode. (a) The user clicks and highlights a subtree for deletion. (b) The subtree is deleted by pressing the “Delete highlighted branches” button or the “delete” key on the keyboard. (c) Erroneously-deleted voxels can be recovered by pressing the “Undelete branches” button.

Proper use of the tool requires a preplanned endobronchial route generated by the peripheral ROI.
PathPlanner tool of the VN suite. Once an appropriate route has been defined, the user can jump to the preplanned biopsy site using the “Jump to image location” function in the Oblique slice dialog, as described in Section D.2.1. Figure D.17 illustrates the result of this operation for the nodule in case 20349-3-25. The center of the view jumps to the preplanned biopsy site, just beyond the surface of the nodule.

![Image of the nodule and route](image.jpg)

**Figure D.17.** End of the preplanned endobronchial route for the nodule in case 20349-3-25.

The user then simply traces the route back to the trachea, searching for any missed branches. In this case, there is one missed branch located just short of the preplanned biopsy site. The branch is visible in Figure D.18(a). To add the branch, the user centers the view on the segmentation voxel that is nearest to the missing branch by holding the “control” key and right-clicking. This voxel serves as the first seed point for the livewire algorithm. The user then presses the “Start LW” button on the Livewire dialog and hovers the mouse cursor over a location in the airway. A red line “snaps” to the airway. Once the red line is in an acceptable location, the user presses the right mouse button to “freeze” the livewire. This turns the red line to blue and sets the end of the blue line as the next...
seed point. The user is now free to rotate the view to capture the next section of airway if possible. Once the branch has been completely added, the user simply clicks the “Stop livewire” button to add the branch to the segmentation.

Figure D.18. Adding a missing branch in the Livewire editing mode. (a) The user locates the missing branch and selects a seed voxel in the segmentation. (b) The user selects “Start Livewire” and moves the cursor to a location inside the missing airway. The livewire algorithm determines the best path for connecting the cursor to the seed voxel (red voxels). (c) The user freezes the livewire (blue voxels) by right-clicking. The end of the frozen livewire becomes the new seed point. (d) Pressing the “Stop livewire” button adds all frozen voxels to the segmentation (now yellow) and kills the livewire.

One important option in the Livewire dialog bears further discussion. Note the “Use HU weights for Livewire” checkbox in the bottom of the dialog. The box is checked by
default. This option tells the SegTool to run a shortest-paths algorithm for livewire that avoids passing through bright voxels that are likely to belong to airway wall. This typically helps contain the livewire within the airways and speed peripheral airway definition. The differential voxel weighting is disadvantageous, however, when the peripheral airway is blocked-off. Figure D.19 gives an example where disabling this option makes it easier to define a peripheral airway.

Once the user has traced back to the trachea, it is clear that the route is clean, complete, and ready for image-based guidance. Any changes can be committed to the segmentation by pressing “Accept + Exit.” This returns the system to the Visualize Segmentation mode and lets the user save the final airway tree.
Figure D.19. The effect of the “Use HU weights for Livewire” option. (a) When the option is checked, the SegTool attempts to avoid passing through bright areas of the image. This makes it unnecessarily difficult to define blocked or stenosed airways as the livewire attempts to go around the blockage instead of through it. (b) Unchecking the option makes it possible to draw a straight line through the blockage.
Proof of Theorem 4.1

This appendix provides a proof for Theorem 4.1 from Section 4.1 which gives several basic properties about induced subtrees of a tree $T$. Recall from the hypothesis of the theorem that $K$ is a subset of $V(T)$ containing $r = \text{root}(T)$.

We begin by considering items 1 and 2. Let $N \triangleq |V(T) \setminus K|$ be the number of vertices to be deleted and let $x^{(1)}, x^{(2)}, ..., x^{(N)}$ be an arbitrary permutation of these vertices. Define the trees $T^{(i)}, i \in \{0, 1, ..., N\}$, inductively with $T^{(0)} = T$ and $T^{(i)}$ the tree obtained by deleting $x^{(i)}$ from $T^{(i-1)}$. The $T^{(i)}$ can be recursively defined using the definition of the vertex deletion operation as

\begin{equation}
V(T^{(i)}) = V(T^{(i-1)}) \setminus \{x^{(i)}\}, \quad \text{and}
\end{equation}

\begin{equation}
E(T^{(i)}) = \{(u, v) \in E(T^{(i-1)}) | u, v \in V(T^{(i)})\} \cup \{(u, x^{(i)}), (x^{(i)}, v) \in E(T^{(i-1)})\}
\end{equation}

for $1 \leq i \leq N$. To show that $T[K]$ is well-defined, it suffices to demonstrate that $T^{(N)}$ is
independent of the order in which the vertices in \( V(T) \setminus K \) are deleted. It is clear from \( (E.1) \) that \( V(T^{(N)}) = K \) for any permutation of \( V(T) \setminus K \). Hence \( V(T[K]) = K \) as claimed and \( V(T[K]) \) is well-defined. Define the set

\[
S^{(i)} \triangleq \left\{ (u, v) \mid u, v \in V(T^{(i)}), u \sim v \text{ in } T, \text{ and } u \sim w \sim v \text{ in } T \Rightarrow w \notin V(T^{(i)}) \right\} \quad (E.3)
\]

We claim \( E(T^{(i)}) = S^{(i)} \) for \( 0 \leq i \leq N \) and proceed by induction. The claim is trivially true for \( i = 0 \) as \( V(T^{(0)}) = V(T) \) and \( E(T^{(0)}) = E(T) \). For \( 0 < i \leq N \) we inductively assume that the claim holds for \( i - 1 \) and prove that it holds for \( i \) via the two steps below:

**Step 1:** Show \( E(T^{(i)}) \subseteq S^{(i)} \)

Choose \((u, v) \in E(T^{(i)})\) arbitrary. If \((u, v) \in E(T^{(i-1)})\) as well, then \((u, v) \in S^{(i-1)}\) by our inductive hypothesis. Thus \( u \sim v \) in \( T \) and \( u \sim w \sim v \) in \( T \Rightarrow w \notin V(T^{(i-1)}) \). Also, \( V(T^{(i-1)}) \supset V(T^{(i)}) \), so \( u \sim w \sim v \) in \( T \Rightarrow w \notin V(T^{(i)}) \) and \((u, v) \in S^{(i)}\) as well.

If, on the other hand, \((u, x^{(i)}), (x^{(i)}, v) \in E(T^{(i-1)})\), we have \((u, x^{(i)}), (x^{(i)}, v) \in S^{(i-1)}\) by our inductive hypothesis. Thus, \( u \sim x^{(i)} \) and \( x^{(i)} \sim v \) in \( T \), implying \( u \sim v \) in \( T \). Furthermore, if \( u \sim w \sim v \) in \( T \), with \( w \neq x^{(i)} \), then either \( u \sim w \sim x^{(i)} \) or \( x^{(i)} \sim w \sim v \) in \( T \), implying \( w \notin V(T^{(i-1)}) \supset V(T^{(i)}) \). Thus, \((u, v) \in S^{(i)}\) and \( E(T^{(i)}) \subseteq S^{(i)} \).

**Step 2:** Show \( E(T^{(i)}) \supseteq S^{(i)} \)

Choose \((u, v) \in S^{(i)}\) arbitrary. First, suppose \( u \sim x^{(i)} \sim v \) in \( T \). We have \( u, x^{(i)}, v \in V(T^{(i-1)}) \), \( u \sim x^{(i)} \) in \( T \), and \( x^{(i)} \sim v \) in \( T \). Furthermore, if \( w \neq x^{(i)} \) and either \( u \sim w \sim x^{(i)} \) or \( x^{(i)} \sim w \sim v \) holds in \( T \), we must have \( u \sim w \sim v \) in \( T \), hence \( w \notin V(T^{(i)}) \) as \((u, v) \in S^{(i)}\). But \( V(T^{(i-1)}) = V(T^{(i)}) \cup \{x^{(i)}\} \), so \( w \notin V(T^{(i-1)}) \) as well. Thus, \((u, x^{(i)}), (x^{(i)}, v) \in S^{(i-1)}\) implying \((u, x^{(i)}), (x^{(i)}, v) \in E(T^{(i-1)})\) by our inductive
hypothesis. This implies \((u, v) \in E(T^{(i)})\).

Alternately, suppose either \(u \not\sim x^{(i)}\) or \(x^{(i)} \not\sim v\) in \(T\). If \(u \sim w \sim v\) in \(T\), then \(w \notin V(T^{(i)})\) as \((u, v) \in S^{(i)}\). Furthermore, it is clear that \(w \neq x^{(i)}\), hence \(w \notin V(T^{(i-1)}) = V(T^{(i)}) \cup \{x^{(i)}\}\). Thus, \(u \sim w \sim v\) in \(T\) \(\Rightarrow w \notin V(T^{(i-1)})\) and \((u, v) \in S^{(i-1)}\). By our inductive hypothesis, this implies \((u, v) \in E(T^{(i-1)})\), hence \((u, v) \in E(T^{(i)})\). This completes the proof that \(E(T^{(i)}) \supseteq S^{(i)}\).

Steps 1 and 2 imply that \(E(T^{(i)}) = S^{(i)}\) for \(1 < i \leq N\). Specifically, \(E(T^{(N)}) = S^{(N)}\).

Substituting \(V(T^{(N)}) = K\) into (E.3) yields

\[
E(T[K]) \triangleq E(T^{(N)}) = S^{(N)} = \{(u, v) \mid u, v \in K, u \sim v \text{ in } T, \text{ and } u \sim w \sim v \text{ in } T \Rightarrow w \notin K\}
\]

(E.4)

as claimed in item 2. Thus, both the vertex and edge sets of \(T[K]\) are independent of the ordering of the \(x^{(i)}\) and \(T[K]\) is well-defined.

We now consider item 3 (\(\Rightarrow\)) Let \(u \sim v\) in \(T[K]\). Then a sequence of vertices \(u = y^{(0)} \sim y^{(1)} \sim \cdots \sim y^{(m)} = v\) exists with \(m \geq 1\) such that \((y^{(j-1)}, y^{(j)}) \in E(T[K])\) for \(1 \leq j \leq m\).

By item 2 we therefore have \(u \sim y^{(1)} \sim \cdots \sim y^{(m-1)} \sim v\) in \(T\), hence \(u \sim v\) in \(T\). (\(\Leftarrow\)) Let \(u \sim v\) in \(T\) and define \(L_{u,v} \triangleq \{w \in K \mid u \sim w \sim v \text{ in } T\}\) be the set of vertices \(w \in K\) lying between \(u\) and \(v\) in \(T\). We proceed by induction on \(|L_{u,v}|\). If \(|L_{u,v}| = 0\), there is no such vertex, hence \(u \sim w \sim v \Rightarrow w \notin K\) holds trivially and \((u, v) \in E(T[K])\) implying \(u \sim v\) in \(T[K]\). Now, suppose the result holds for all \(x, y \in K\) such that \(x \sim y\) in \(T\) and \(0 \leq |L_{x,y}| < |L_{u,v}|\). Choose any vertex \(w \in L_{u,v}\). Now \(|L_{u,w}|\) and \(|L_{w,v}|\) are strictly less than \(|L_{u,v}|\), hence \(u \sim w \sim v\) in \(T[K]\) by our inductive hypothesis. Thus, \(u \sim v\) in \(T[K]\).

To prove item 4 it suffices to show that \(|E(T[K])| = |V(T[K])| - 1\) and \(T[K]\) is connected;
i.e. that \( r \leadsto v \) for all \( v \in V(T[K]) \setminus r \) [124]. Note that \( r \in V(T[K]) = K \) holds by hypothesis. Select an arbitrary \( v \in V(T[K]) \setminus r \). It is clear that \( r \leadsto v \) in \( T \), as \( T \) is itself a tree rooted at \( r \). Item 3 implies \( r \leadsto v \) in \( T[K] \) as well. Thus, \( T[K] \) is connected. Equations (E.1) and (E.2) make it clear that \( |V(T^{(i)})| = |V(T^{(i-1)})| - 1 \) and \( |E(T^{(i)})| = |E(T^{(i-1)})| - 1 \) both hold. An easy induction yields the result. Thus, \( T[K] \in \mathbb{T} \) with root(\( T[K] \)) = \( r \).
Isomorphism Results for Local Deformation Models

This appendix demonstrates that any match generated from primitives must define an isomorphic common tree. This result is required in the proof of Theorem 4.2. Recall that for $\phi$ to be an isomorphism between $T_1[D(\phi)]$ and $T_2[R(\phi)]$, it must be bijective and preserve the edge sets of the trees. Lemmas F.1-F.4 break up the proof by demonstrating separately that $\phi$ is one-to-one and preserves the ancestry relationships of $T_1[D(\phi)]$ and $T_2[R(\phi)]$. Theorem F.1 combines these results to show $T_1[D(\phi)] \cong T_2[R(\phi)]$.

Lemma F.1 Consider a local deformation model $\Psi$, two trees $T_1, T_2 \in T$, a valid match $\phi \in \Phi_\Psi(T_1, T_2)$, and two matched vertices $u_1, v_1 \in D(\phi)$ such that $u_1 \rightsquigarrow v_1$ in $T_1[D(\phi)]$. Then $\phi(u_1) \neq \phi(v_1)$ and $\phi(u_1) \rightsquigarrow \phi(v_1)$ in $T_2[R(\phi)]$.

Proof. By induction on the length of $u_1Pv_1$ in $T_1[D(\phi)]$. Let $L(u_1, v_1)$ be this length. If $L(u_1, v_1) = 1$, then $(u_1, v_1) \in E(T_1[D(\phi)]) \Rightarrow u_1, v_1 \in D(\psi^{u_1}_\phi)$, which must be a valid
Consider a local deformation model \( \Phi(T_1, T_2) \). The function \( \psi^u_\phi \) is therefore one-to-one by Definition [4.4]. Hence, \( \phi(u_1) = \psi^u_\phi(u_1) \neq \psi^u_\phi(v_1) = \phi(v_1) \). Furthermore, \( \phi(u_1) = \psi^u_\phi(u_1) \Rightarrow \psi^u_\phi(v_1) = \phi(v_1) \) in \( T_2 \), also by Definition [4.4]. Thus, \( \phi(u_1) \Rightarrow \phi(v_1) T_2[R(\phi)] \) as well, by item 3 of Theorem [4.1].

We now make the inductive assumption that the lemma holds if \( L(u_1,v_1) = N - 1 \). Suppose \( u_1 \sim v_1 \) in \( T_1[D(\phi)] \) with \( L(u_1,v_1) = N \). Define \( x_1 \) to be the first vertex along \( u_1Pv_1 \) in \( T_1[D(\phi)] \). Thus, \( u_1,x_1,v_1 \in D(\phi) \) with \( u_1 \sim x_1 \sim v_1 \), \( L(u_1,x_1) = 1 \), and \( L(x_1,v_1) = N - 1 \). By the preceding argument, \( \phi(u_1) \neq \phi(x_1) \) and \( \phi(u_1) \sim \phi(x_1) \) in \( T_2[R(\phi)] \). Furthermore, our inductive hypothesis yields \( \phi(x_1) \neq \phi(v_1) \) and \( \phi(x_1) \sim \phi(v_1) \) in \( T_2[R(\phi)] \). Now, we cannot have \( \phi(u_1) = \phi(v_1) \), or else \( \phi(u_1) \sim \phi(x_1) \sim \phi(v_1) \Rightarrow \phi(u_1) \sim \phi(v_1) \) in \( T_2[R(\phi)] \), which is impossible as the \( \sim \) relation is irreflexive (recall that \( T_2[R(\phi)] \) is a tree by Theorem [4.1]). Finally, \( \phi(u_1) \sim \phi(v_1) \) by transitivity. Thus, the lemma holds if \( L(u_1,v_1) = N \) and the inductive proof is complete. \( \square \)

**Lemma F.2** Consider a local deformation model \( \Psi \), two trees \( T_1, T_2 \in \mathcal{T} \), and a valid match \( \phi \in \Phi(\Psi(T_1, T_2)) \). Then \( \phi \) is one-to-one.

*Proof.* We must show that \( u_1, v_1 \in D(\phi) \Rightarrow \phi(u_1) \neq \phi(v_1) \). The cases where \( u_1 \sim v_1 \) or \( v_1 \sim u_1 \) in \( T_1[D(\phi)] \) are handled by Lemma [F.1]. We may therefore assume that \( u_1 \nshorteq v_1 \) and \( v_1 \nshorteq u_1 \) in \( T_1[D(\phi)] \). Let \( z_1 = lca(u_1,v_1) \) in \( T_1[D(\phi)] \). By definition, \( z_1 \) is distinct from both \( u_1 \) and \( v_1 \) (neither is an ancestor of the other) and both \( z_1 \sim u_1 \) and \( z_1 \sim v_1 \) in \( T_1[D(\phi)] \) hold. By Lemma [F.1], therefore, we have \( \phi(z_1) \neq \phi(u_1) \), \( \phi(z_1) \neq \phi(v_1) \), \( \phi(z_1) \sim \phi(u_1) \) in \( T_2[R(\phi)] \), and \( \phi(z_1) \sim \phi(v_1) \) in \( T_2[R(\phi)] \). Four cases are possible:

1. \( u_1 \in D(\psi^z_\phi) \) and \( v_1 \in D(\psi^z_\phi) \): We trivially have \( \phi(u_1) \neq \phi(v_1) \) as \( \psi^z_\phi \) must be
Consider a local deformation model \( \Psi \), two trees \( T_1, T_2 \in \mathbb{T} \), a valid match \( \phi \in \Phi_\Psi(T_1, T_2) \), and two matched vertices \( u_1, v_1 \in D(\phi) \) such that \( \phi(u_1) \rightsquigarrow \phi(v_1) \) in \( T_2[R(\phi)] \). Then \( u_1 \rightsquigarrow v_1 \) in \( T_1[D(\phi)] \).

Lemma F.3

1. \( u_1 \notin D(\psi_0^z) \) and \( v_1 \in D(\psi_0^z) \): Define \( x_1 \) to be the first vertex along \( z_1 Pu_1 \) in \( T_1[D(\phi)] \). Thus, \( (z_1, x_1) \in E(T_1[D(\phi)]) \) and \( x_1 \in D(\psi_0^z) \). Furthermore, \( z_1 \rightsquigarrow x_1 \rightsquigarrow u_1 \) in \( T_1[D(\phi)] \), so \( \phi(x_1) \neq \phi(u_1) \) and \( \phi(x_1) \rightsquigarrow \phi(u_1) \) in \( T_2[R(\phi)] \). Now, if it were true that \( \phi(u_1) = \phi(v_1) \), we would have \( \phi(x_1) \rightsquigarrow \phi(u_1) \Rightarrow \phi(x_1) \rightsquigarrow \phi(v_1) \), which is impossible as \( x_1, v_1 \in D(\psi_0^z) \Rightarrow \phi(x_1) \neq \phi(v_1) \) by Definition 4.4.

2. \( u_1 \notin D(\psi_0^z) \) and \( v_1 \in D(\psi_0^z) \): Define \( x_1 \) as in case 1. As \( z_1 \) is the least common ancestor of \( u_1 \) and \( v_1 \), \( x_1 \) and \( y_1 \) must be distinct. Furthermore, as \( x_1 \) and \( y_1 \) are children of \( z_1 \) in \( T_1[D(\phi)] \), \( x_1, y_1 \in D(\psi_0^z) \) so \( \phi(x_1) \neq \phi(y_1) \). Again, \( z_1 \rightsquigarrow x_1 \rightsquigarrow u_1 \) in \( T_1[D(\phi)] \) \( \Rightarrow \phi(z_1) \neq \phi(x_1) \neq \phi(u_1) \) and \( \phi(z_1) \rightsquigarrow \phi(x_1) \rightsquigarrow \phi(u_1) \) in \( T_2[R(\phi)] \). Similarly, \( \phi(z_1) \neq \phi(y_1) \neq \phi(v_1) \) and \( \phi(z_1) \rightsquigarrow \phi(y_1) \rightsquigarrow \phi(v_1) \) in \( T_2[R(\phi)] \). Now, were it true that \( \phi(u_1) = \phi(v_1) \), we would have both \( \phi(z_1) \rightsquigarrow \phi(x_1) \rightsquigarrow \phi(u_1) \) and \( \phi(z_1) \rightsquigarrow \phi(y_1) \rightsquigarrow \phi(u_1) \) in \( T_2[R(\phi)] \). As \( T_2[R(\phi)] \) is a tree, path uniqueness implies either \( \phi(x_1) \rightsquigarrow \phi(y_1) \) or \( \phi(y_1) \rightsquigarrow \phi(x_1) \) in \( T_2[R(\phi)] \), both of which are impossible according to Definition 4.4 as \( x_1, y_1 \in D(\psi_0^z) \). Thus, \( \phi(u_1) \neq \phi(v_1) \).

\( \square \)
Proof. We prove the contrapositive, namely that \( u_1 \not\sim v_1 \) in \( T_1[D(\phi)] \Rightarrow \phi(u_1) \not\sim \phi(v_1) \) in \( T_2[R(\phi)] \). As such, assume \( u_1 \not\sim v_1 \) in \( T_1[D(\phi)] \). Now, if \( v_1 \sim u_1 \) in \( T_1[D(\phi)] \), then \( \phi(v_1) \sim \phi(u_1) \) in \( T_2[R(\phi)] \) by Lemma F.1.1, hence \( \phi(u_1) \not\sim \phi(v_1) \) in \( T_2[R(\phi)] \) by antisymmetry. We may therefore assume \( v_1 \not\sim u_1 \) in \( T_1[D(\phi)] \). Define \( z_1 = lca(u_1, v_1) \) in \( T_1[D(\phi)] \). As in the proof of Lemma F.2, \( z_1 \) must be distinct from \( u_1 \) and \( v_1 \) and four cases are possible:

1. \( u_1 \in D(\psi_y^{z_1}) \) and \( v_1 \in D(\psi_y^{z_1}) \): We trivially have \( \phi(u_1) \not\sim \phi(v_1) \) by Definition 4.4.

2. \( u_1 \notin D(\psi_y^{z_1}) \) and \( v_1 \in D(\psi_y^{z_1}) \): Define \( x_1 \) to be the first vertex along \( z_1 Pu_1 \) in \( T_1[D(\phi)] \).

Thus, \( \phi(x_1) \sim \phi(u_1) \) in \( T_2[R(\phi)] \). Now, if \( \phi(u_1) \sim \phi(v_1) \) in \( T_2[R(\phi)] \), we would have \( \phi(x_1) \sim \phi(u_1) \sim \phi(v_1) \Rightarrow \phi(x_1) \sim \phi(v_1) \) in \( T_2[R(\phi)] \), hence in \( T_2 \). This is impossible as \( x_1, v_1 \in D(\psi_y^{z_1}) \).

3. \( u_1 \in D(\psi_y^{z_1}) \) and \( v_1 \notin D(\psi_y^{z_1}) \): Define \( y_1 \) to be the first vertex along \( z_1 Pv_1 \) in \( T_1[D(\phi)] \).

Now, if \( \phi(u_1) \sim \phi(v_1) \) in \( T_2[R(\phi)] \), we have both \( \phi(z_1) \sim \phi(u_1) \sim \phi(v_1) \) and \( \phi(z_1) \sim \phi(y_1) \sim \phi(v_1) \) in \( T_2[R(\phi)] \), hence either \( \phi(u_1) \sim \phi(y_1) \) or \( \phi(y_1) \sim \phi(u_1) \) in \( T_2[R(\phi)] \) by path uniqueness. As \( u_1, y_1 \in D(\psi_y^{z_1}) \), both are impossible by Definition 4.4.

4. \( u_1 \notin D(\psi_y^{z_1}) \) and \( v_1 \notin D(\psi_y^{z_1}) \): Define \( x_1 \) as in case 2 and \( y_1 \) as in case 3. If \( \phi(u_1) \sim \phi(v_1) \) in \( T_2[R(\phi)] \), then \( \phi(x_1) \sim \phi(v_1) \) in \( T_2[R(\phi)] \). Thus, both \( \phi(z_1) \sim \phi(x_1) \sim \phi(v_1) \) and \( \phi(z_1) \sim \phi(y_1) \sim \phi(v_1) \). As in case 3, this is impossible because \( x_1, y_1 \in D(\psi_y^{z_1}) \).

\( \square \)

Lemma F.4 Consider two trees \( T_1, T_2 \in \mathbb{T} \) and a function \( f \) that is a bijection between \( V(T_1) \) and \( V(T_2) \). Suppose also that \( f \) preserves ancestor relationships in \( T_1 \) and \( T_2 \); i.e.,
that \( u_1 \rightsquigarrow v_1 \) in \( T_1 \) \( \Leftrightarrow \) \( f(u_1) \rightsquigarrow f(v_1) \) in \( T_2 \) for all \( u_1, v_1 \in V(T_1) \). Then \( f \) is an isomorphism between \( T_1 \) and \( T_2 \).

**Proof.** Recall that for \( f \) to be an isomorphism between \( T_1 \) and \( T_2 \) it must be both a bijection between \( V(T_1) \) and \( V(T_2) \) and satisfy \( (u_1, v_1) \in E(T_1) \Leftrightarrow (f(u_1), f(v_1)) \in E(T_2) \) for all \( u_1, v_1 \in V(T_1) \). As \( f \) is a bijection by hypothesis, we need only verify the second part of the definition, that \( f \) preserves the parent/child relationships in \( T_1 \) and \( T_2 \).

Now, for any \( T \in T \), the edge set \( E(T) \) can be determined from the \( \rightsquigarrow \) relation via

\[(u, v) \in E(T) \Leftrightarrow u \rightsquigarrow v \text{ in } T \text{ and } \not\exists x \in V(T) \text{ with } u \rightsquigarrow x \rightsquigarrow v \text{ in } T. \quad (F.1)\]

Equation \( (F.1) \) states that \( u \) is the parent of \( v \) iff \( u \) is an ancestor of \( v \) and there is no other vertex \( x \) lying on the path between \( u \) and \( v \) in \( T \). We can verify that \( f \) preserves the parent/child relationships in \( T_1 \) and \( T_2 \) by the following sequence of implications.

\[(u_1, v_1) \in E(T_1) \Leftrightarrow u_1 \rightsquigarrow v_1 \text{ in } T_1 \text{ and } \not\exists x_1 \in V(T_1) \text{ with } u_1 \rightsquigarrow x_1 \rightsquigarrow v_1 \text{ in } T_1 \quad (F.2)\]

\[\Leftrightarrow f(u_1) \rightsquigarrow f(v_1) \text{ in } T_2 \text{ and } \not\exists x_1 \in V(T_1) \text{ with } f(u_1) \rightsquigarrow f(x_1) \rightsquigarrow f(v_1) \text{ in } T_2 \quad (F.3)\]

\[\Leftrightarrow f(u_1) \rightsquigarrow f(v_1) \text{ in } T_2 \text{ and } \not\exists x_2 \in V(T_2) \text{ with } f(u_1) \rightsquigarrow x_2 \rightsquigarrow f(v_1) \text{ in } T_2 \quad (F.4)\]

\[\Leftrightarrow (f(u_1), f(v_1)) \in E(T_2) \quad (F.5)\]

Here, the implications in \( (F.2) \) and \( (F.5) \) are direct applications of \( (F.1) \). The implication in \( (F.3) \) follows from the hypothesis that \( u_1 \rightsquigarrow v_1 \text{ in } T_1 \Leftrightarrow f(u_1) \rightsquigarrow f(v_1) \text{ in } T_2 \). Finally, \( (F.4) \) is true because \( f \) is a bijection between \( V(T_1) \) and \( V(T_2) \), thus for any \( x_2 \in V(T_2) \) there exists a unique vertex \( y_1 \in V(T_1) \) \( f(y_1) = x_2 \). \( \Box \)
Theorem F.1 Consider a local deformation model $\Psi$ and trees $T_1, T_2 \in \mathbb{T}$. Each $\phi \in \Phi_\Psi(T_1, T_2)$ is an isomorphism between $T_1[D(\phi)]$ and $T_2[R(\phi)]$.

Proof. By Lemma F.2, $\phi$ is a one-to-one mapping between $D(\phi) = V(T_1[D(\phi)])$ and $R(\phi) = V(T_2[R(\phi)])$. It is trivially onto. It is therefore a bijection between $V(T_1[D(\phi)])$ and $V(T_2[R(\phi)])$. Furthermore, Lemmas F.1 and F.3 yield $u_1 \sim v_1 \in T_1[D(\phi)] \iff \phi(u_1) \sim \phi(v_1) \in T_2[R(\phi)]$ for all $u_1, v_1 \in D(\phi) = V(T_1[D(\phi)])$. We can therefore apply Lemma F.4 to show $\phi$ to be an isomorphism between $T_1[D(\phi)]$ and $T_2[R(\phi)]$. □
Analysis of LocateOptimalPrimitive Subroutine

The running time of the LocateOptimalPrimitive subroutine defined in Algorithm 4.2 can be analyzed as follows. The sets $PSN(T_1^{u_1})$ and $PSN(T_2^{u_2})$ each consist of $\delta(u_1) + 1$ and $\delta(u_2) + 1$ possible supernodes. Thus, the main loop in lines 8–21 is executed $O(\delta(u_1)\delta(u_2))$ times. The obvious bottlenecks in the main loop are the construction of $K_{N,N}$ in line 11 and the maximum-weight matching problem in line 13.

To analyze line 11 note that the effective degree of $SN_1$ is bounded by

$$\delta_{\text{eff}}(SN_1) \leq \delta(u_1) + \delta(T_1) - 1 = O(\delta(T_1)). \quad (G.1)$$

A similar bound can be placed on $\delta_{\text{eff}}(SN_2)$. Computing each of the $O(\delta(T_1)\delta(T_2))$ non-
trivial edge weights requires enumeration of the set $X_i \times Y_j$. The size of $X_i$ is at most

$$|X_i| \leq 1 + \delta(T_1) + \delta(T_1)^2 + \delta(T_1)^3 = O(\delta(T_1)^3).$$  \quad \text{(G.2)}$$

Similarly, $|Y_j| = O(\delta(T_2)^3)$. Thus, $K_{N,N}$ can be constructed in $O(\delta(T_1)^4 \delta(T_2)^4)$ time. The Hungarian algorithm solves line 13 in $O(N^3) = O((\delta(T_1) + \delta(T_2))^3) = O(\max\{\delta(T_1), \delta(T_2)\}^3)$ time. Thus, the main loop is dominated by the construction of $K_{N,N}$ and the overall sub-routine running time is $O(\delta(u_1)\delta(u_2)\delta(T_1)^4 \delta(T_2)^4)$.

It is also possible to bound the maximum number of vertices matched by the primitive returned by $\textbf{LocateOptimalPrimitive}$. This result is important in analyzing the space requirements of Algorithm 4.1 which stores optimal primitives between all pairs of subtrees of $T_1$ and $T_2$. The bound is determined by noting that the maximum number of edges connecting non-dummy vertices in $K_{N,N}$ for any potential pair of supernodes $SN_1$ and $SN_2$ is bounded by the minimum of the number of effective children of $SN_1$ and the number of effective children of $SN_2$. Thus, the maximum number of matches in a primitive obtained from a matching in $K_{N,N}$, including the trivial $(u_1, u_2)$ match, is

$$1 + \min\{\delta_{eff}(SN_1), \delta_{eff}(SN_2)\} \leq$$

$$1 + \min\{\delta(u_1) + \delta(T_1) - 1, \delta(u_2) + \delta(T_2) - 1\} =$$

$$O(\min\{\delta(T_1), \delta(T_2)\}).$$  \quad \text{(G.3)}$$

This bound holds for any pair of potential supernodes. The maximum number of vertices matched by $\psi^* \{T_1^{u_1}, T_2^{u_2}\}$ is therefore $O(\min\{\delta(T_1), \delta(T_2)\})$. 
User Manual for TreeMatch Tool

This appendix describes the usage of the TreeMatch Tool, which enables the matching and simultaneous visualization of two anatomical trees extracted from 3D MDCT data. The appendix is organized in four parts. First, Section H.1 describes the input data used by the system. Section H.2 describes the controls for simultaneously visualizing two trees and two MDCT images. Section H.3 describes the usage of the automatic matching algorithm. Finally, Section H.4 describes the controls for matching two trees by hand or manually refining an automatic match.

H.1 System Initialization and Inputs

The TreeMatch tool takes VN navigational path (.npth) and quantitative path (.qpth) files as input. These files describe the 3D centerline structure of the input trees ($T_1$ and $T_2$) and contain quantitative measurements such as branching angles and minimum and maximum branch diameter. Both path file types describe the centerline locations in terms of the original 3D image coordinates. The voxel dimensions ($\Delta x$, $\Delta y$, and $\Delta z$) of the 3D image
data are therefore necessary in order to determine the true physical dimensions of the input trees. The voxel dimensions are acquired from the Analyze header file (.hdr) of the images used to define $T_1$ and $T_2$.

The TreeMatch Tool therefore requires four files as input:

1. The image header for $T_1$
2. The path file describing $T_1$
3. The image header for $T_2$ (typically different from the header for $T_1$)
4. The path file describing $T_2$

The controls for loading these files into the TreeMatch tool are illustrated in Figure [H.1](#). Note that the file loading controls also contain a check box labeled “Load image data,” which is unchecked by default. If the box is unchecked, the tool will load only the header file associated with the two input images. This is very fast, as header files contain only a few hundred bytes of data. If checked, however, the tool will load the entire 3D volumes of the two images. This may take a few seconds, as the images are typically several hundred megabytes each. Once the images are loaded, however, they can be simultaneously visualized and superimposed with $T_1$ and $T_2$ as described in the next section. Note that “Load image data” check box is disabled as soon as an image or image header is loaded. It is therefore not possible to load the image data for only one of the two input trees.
Figure H.1. File input controls for the TreeMatch Tool. The numbered inputs correspond to the four required input files described in the text.

H.2 Simultaneous Visualization of Two Trees

Once the image information and input trees have been loaded, the centerline representations of $T_1$ and $T_2$ are displayed in two parallel panes, with $T_1$ on the left. Figure H.2 illustrates the initial view. The trees can be viewed from any angle using a set of 3D camera controls. The camera controls, and all keyboard controls used by the TreeMatch Tool, are described in a help dialog summoned by clicking on the “Help” button in the upper-right of the main window. Figure H.3 illustrates the help dialog.

The 3D camera locations of the left and right panes are initially linked. This means that any camera control applied to the left pane is mirrored in the right pane and vice-versa. The panes can also be unlinked. When unlinked, the views in the left and right panes can be independently manipulated. Figure H.4 illustrates the difference between the linked and unlinked modes. The key combination “Ctrl + L” toggles between the linked and unlinked modes.

If the full 3D image volumes have been loaded into the TreeMatch Tool, oblique cross-sections through the data can be simultaneously visualized in the left and right panes. The key combination “Ctrl + I” toggles the oblique cross-section on and off in both panes.
Figure H.2. The initial view of the TreeMatch Tool after all four input files have been loaded. The left pane displays $T_1$.

Figure [H.5] gives an example of the cross-section view.

Note that the oblique slices in the left and right panes of Figure [H.5] display very different image information. The difference arises because the image coordinate systems between the left and right panes do not correspond. Thus, although the camera is located at the same position in the image coordinate systems of both panes, it views different anatomical
Figure H.3. The TreeMatch Tool help dialog summarizes all system controls and commands. structures. As it is frequently desirable to view corresponding anatomical structures in the oblique slices of both planes, the TreeMatch Tool provides a simple method for rigidly aligning the two coordinate systems. To perform the alignment, the user simply places three alignment “darts” near corresponding anatomical landmarks in the left and right panes. The landmarks can be located in the images or on the trees themselves, as illustrated in Figure H.6(a). Note that the alignment typically only works well in a local area as the true global
mapping between the two image coordinate systems is typically non-rigid.

H.3 Automatic Matching Algorithm

The automatic tree-matching algorithm described in Chapter 4 of this thesis can be invoked by pressing the “Match trees” button in the upper right corner of the main window. The “Match trees” button brings up a dialog containing a set of automatic matching algorithm parameters. The default matching parameters are tailored to matching pairs of human airway trees. In particular, as described in Section 4.3.3, the “Branchpoint distance threshold” parameter is far too large for trees extracted from mice and other small animals.

The “Branch sim. weight,” “Branchpoint sim. weight,” “Length ratio weight,” “Length ratio threshold,” and “Cosine running dir angle threshold” correspond to the similarity measure parameters described in Section 4.2.2. The “Max nodes per matched branch” and “Max nodes per supernode” parameters correspond to items 3 and 3 of Definition 4.6. Note that the “Max nodes per matched branch” parameter expresses path length in terms of number of vertices as opposed to number of edges. Finally, the “Max supernode dist” represents the maximum distance (in mm) between vertices allowed to belong to a supernode.

Once the matching parameters have been chosen appropriately for the particular type of anatomical trees to be matched, the matching algorithm is invoked by pressing the “OK” button. Once the matching algorithm has been run, the branches of $T_1$ and $T_2$ are colored according to whether or not they are included in the common tree of the automatic match. Blue branches are included in the common tree and green branches are “spurious”.
Corresponding branches can be determined by right-clicking on any matched branch. The selected branch, and its corresponding branch in the other tree, will both be highlighted in yellow, as illustrated in Figure H.8.

An automatic match can be saved to an ASCII text file by pressing “Ctrl + S.” This brings up a standard windows file dialog from which the user can select an appropriate file name. By default, tree-match files are saved with a .tmc extension. Figure H.9 provides an example of an output file. Tree-match files can also be loaded into the tool by pressing “Ctrl + L.”

Once a match exists, it can be used to perform a global rigid registration between the coordinate systems of $T_1$ and $T_2$. The rigid-registration is performed by a random sampling and consensus (RANSAC) algorithm [182]. The algorithm is invoked by pressing the “Register views” button in the upper-right corner of the main window. This brings up a RANSAC algorithm parameter dialog, which enables the user to select appropriate values for the number of random samples considered by the algorithm and the maximum distance between two matched branchpoints that are considered to be “inliers.” Again, the inlier distance has been tuned for the case when $T_1$ and $T_2$ represent human airway trees.

H.4 Hand Matching

The TreeMatch Tool also provides tools manually matching branches between $T_1$ and $T_2$. This capability was used to determine ground-truth matches for comparison in Section 4.3, but may also prove to be useful in correcting a small number of errors in a generally-correct automatic match. The interface is straightforward. To add a match, the user simply
highlights one branch each from $T_1$ and $T_2$, then presses the ‘M’ key. The system omits a warning if the user attempts to match any branch more than once. To remove a match, the user highlights a matched branch in either $T_1$ or $T_2$. The corresponding branch in the other tree will be highlighted automatically, as described in the previous section. The match is removed by pressing the ‘U’ key.
Figure H.4. Effect of camera controls in the TreeMatch Tool. (a) The default view from Figure H.2. (b) The view after moving the camera in the left pane when the panes are linked. Note that the camera remains in the same relative position in both panes. (c) The view after moving the camera in the left pane when the panes are unlinked. The cameras are now in different relative positions in the left and right panes.
Figure H.5. The oblique cross-section view in the TreeMatch Tool.
Figure H.6. Local rigid registration using three alignment darts in TreeMatch Tool. (a) The user selects three darts corresponding to visual landmarks in the left and right panes. Landmarks are selected by double-clicking on the trees or image data. (b) The three pairs of darts are used to rigidly align the camera coordinate systems of the left and right panes. Once the camera coordinates are aligned, the user can view corresponding image data in the left and right panes.
Figure H.7. The match parameters dialog.
Figure H.8. TreeMatch Tool view after automatic matching algorithm has been run. (a) Branches matched by the automatic algorithm are colored blue. Unmatched branches are green. (b) Correspondences between $T_1$ and $T_2$ can be determined by right-clicking on any matched branch. Here, the user clicks on the left main bronchus in $T_1$. Both this branch and the corresponding branch in $T_2$ are highlighted in yellow.
Figure H.9. An example of a TreeMatch Tool output file.
User Manual for the TreeLabel Tool

This appendix describes the usage of the TreeLabel tool, which is useful for generating hand-labeled human airway trees. The tool is invoked by double-clicking on the TreeLabel tool icon illustrated in Figure I.1. Figure I.2 illustrates the main window of the tool at startup. The tool takes an Analyze image and a VN path file (.pth, .qpth, or .npth) as inputs. Once the input files have been loaded, the user is presented with a 3D rendering of an unlabeled input tree, as illustrated in Figure I.3.

![Image](image_url)

**Figure I.1.** Windows icon for the TreeLabel tool.

Pressing the “Help” button summons a dialog summarizing the tool’s keyboard and mouse controls, as illustrated in Figure I.4. Checking the “Show Netter” box summons a dialog containing Netter’s 2D anatomical map illustrated in Figure 5.1(b). Labels are assigned by right-clicking on a branch and selecting the appropriate anatomical name from
Figure I.2. TreeLabel tool main window at startup.

Figure I.3. An unlabeled input tree.
a list. Figure I.5 illustrates the labeling process for the trachea. Labeled branches change color to match the corresponding colors in Netter’s diagram [141]. By default, the label name is also displayed in 3D. These text labels can be disabled by pressing the 'L' key. When a label is assigned to a segmental bronchus, all of its descendant branches receive the same label as they must all belong to the same pulmonary segment. Figure I.6 illustrates the labeling of a segmental bronchus.

![Help dialog](image)

**Figure I.4.** The TreeLabel tool help dialog.

Once the tree has been completely labeled, the user can save an augmented VN path file by pressing the “Save Paths” button at the top of the dialog. Figure I.7 depicts a labeled .npth file. Branch labels are stored at the viewsite level under the “LABEL” data category, which was previously unused. All viewsites on a branch receive the same label. As the file format require the data to be numerical, labels are assigned numerical representations according to the convention in Table I.1. Numbers from 1-22 represent segmental bronchi. As all branches belonging to a pulmonary segment receive the same label, the true segmental bronchus is identified as the (unique) branch with a particular segmental label whose
Figure 1.5. Labeling the trachea.

parent has a different label. Thus, in the numbering scheme of Table I.1, the right anterior segmental bronchus is the unique branch with label number 6 whose parent does not also have label number 6.
Figure I.6. Labeling a segmental bronchus ($RB^1$).
Figure I.7. A labeled VN .npth file. The branch labels are stored at the viewsite level under the “LABEL” data category. The category was previously unused in VN. Here, the first viewsite in Branch 0 receives label number 23, which corresponds to the trachea as per the numbering convention of Table I.1.
<table>
<thead>
<tr>
<th>Label</th>
<th>Number</th>
<th>Label</th>
<th>Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unlabeled</td>
<td>0</td>
<td>RUL, unknown segment</td>
<td>41</td>
</tr>
<tr>
<td>RB¹</td>
<td>1</td>
<td>RML, unknown segment</td>
<td>42</td>
</tr>
<tr>
<td>RB²</td>
<td>2</td>
<td>RLL, unknown segment</td>
<td>43</td>
</tr>
<tr>
<td>RB³</td>
<td>3</td>
<td>LUL, unknown segment</td>
<td>44</td>
</tr>
<tr>
<td>RB⁴</td>
<td>4</td>
<td>LLL, unknown segment</td>
<td>45</td>
</tr>
<tr>
<td>RB⁵</td>
<td>5</td>
<td>Right lung, unknown lobe</td>
<td>51</td>
</tr>
<tr>
<td>RB⁶</td>
<td>6</td>
<td>Left lung, unknown lobe</td>
<td>52</td>
</tr>
<tr>
<td>RB⁷</td>
<td>7</td>
<td>RB¹+²</td>
<td>60</td>
</tr>
<tr>
<td>RB⁸</td>
<td>8</td>
<td>RB¹+³</td>
<td>61</td>
</tr>
<tr>
<td>RB⁹</td>
<td>9</td>
<td>RB²+³</td>
<td>62</td>
</tr>
<tr>
<td>RB¹⁰</td>
<td>10</td>
<td>RB¹+²</td>
<td>63</td>
</tr>
<tr>
<td>LB¹+²</td>
<td>11</td>
<td>RB²+³</td>
<td>64</td>
</tr>
<tr>
<td>LB³</td>
<td>13</td>
<td>RB⁹+¹⁰</td>
<td>65</td>
</tr>
<tr>
<td>LB⁴</td>
<td>14</td>
<td>RB⁷+⁸</td>
<td>66</td>
</tr>
<tr>
<td>LB⁵</td>
<td>15</td>
<td>RB⁷+⁹</td>
<td>67</td>
</tr>
<tr>
<td>LB⁶</td>
<td>16</td>
<td>RB⁷+¹⁰</td>
<td>68</td>
</tr>
<tr>
<td>LB⁷+⁸</td>
<td>17</td>
<td>LB⁹+¹⁰</td>
<td>69</td>
</tr>
<tr>
<td>LB⁹</td>
<td>19</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LB¹⁰</td>
<td>20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RB*</td>
<td>21</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LB*</td>
<td>22</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trachea</td>
<td>23</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Right main bronchus</td>
<td>24</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Right upper lobe bronchus</td>
<td>25</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Intermediate bronchus</td>
<td>26</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Middle lobe bronchus</td>
<td>27</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Right lower lobe bronchus</td>
<td>28</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Left main bronchus</td>
<td>30</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Left upper lobe bronchus</td>
<td>31</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Superior division bronchus</td>
<td>32</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lingular bronchus</td>
<td>33</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Left lower lobe bronchus</td>
<td>34</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 1.1. The label numbering scheme used by the TreeLabel tool.
References


Vita

Michael W. Graham

Michael W. Graham was born in Jeanette, Pennsylvania on December 6th, 1978. He received a B.S. with honors in Electrical Engineering from the Pennsylvania State University in May 2001. From June 2001 through June 2003, he taught high school mathematics in Warrenton, NC through the Teach For America program of Americorps. He has been enrolled in the Ph.D. program in the Department of Electrical Engineering at the Pennsylvania State University since August, 2003, and received an M.S. in Electrical Engineering in May, 2005. His research interests include 3D medical image processing and visualization, algorithm design and analysis, machine learning, and pattern recognition. He will continue work as a software engineer at Google, Inc. in Pittsburgh, Pennsylvania.