

1998

Internal Defect Detection in Hardwood Logs With Fast Magnetic Resonance Imaging.

Eyler Robert Coates Jr
Louisiana State University and Agricultural & Mechanical College

Follow this and additional works at: https://digitalcommons.lsu.edu/gradschool_disstheses

Recommended Citation

Coates, Eyler Robert Jr, "Internal Defect Detection in Hardwood Logs With Fast Magnetic Resonance Imaging." (1998). *LSU Historical Dissertations and Theses*. 6818.
https://digitalcommons.lsu.edu/gradschool_disstheses/6818

This Dissertation is brought to you for free and open access by the Graduate School at LSU Digital Commons. It has been accepted for inclusion in LSU Historical Dissertations and Theses by an authorized administrator of LSU Digital Commons. For more information, please contact gradetd@lsu.edu.

INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI

A Bell & Howell Information Company
300 North Zeeb Road, Ann Arbor MI 48106-1346 USA
313/761-4700 800/521-0600

**INTERNAL DEFECT DETECTION IN HARDWOOD LOGS WITH
FAST MAGNETIC RESONANCE IMAGING**

A Dissertation

Submitted to the Graduate Faculty of the
Louisiana State University and
Agricultural and Mechanical College
in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

in

The Interdepartmental Programs in Engineering

by

Eyler Robert Coates, Jr.
B.S.I.E., Louisiana State University, 1979
M.S.E.S., Louisiana State University, 1996
December 1998

UMI Number: 9922067

**Copyright 1998 by
Coates, Eyler Robert, Jr.**

All rights reserved.

**UMI Microform 9922067
Copyright 1999, by UMI Company. All rights reserved.**

**This microform edition is protected against unauthorized
copying under Title 17, United States Code.**

UMI
300 North Zeeb Road
Ann Arbor, MI 48103

©Copyright 1998
Eyler Robert Coates, Jr.
All rights reserved

ACKNOWLEDGMENTS

I would like to thank Dr. T. Warren Liao for the opportunity to conduct research on an interesting and practical topic. He has been very interested and supportive of my academic career. It is hard to even imagine a more helpful professor. My sincere thanks to him for his continuous guidance, support, encouragement and never-ending patience.

I would like to thank Dr. Sun Joseph Chang for his guidance concerning the physical interpretation of the magnetic resonance images, and his support in providing the images that form the basis of this study.

I would also like to thank Dr. Charles Harlow whose expertise in image processing was most important to my research. His passing comments were more helpful than he would ever know.

I would also like to thank Dr. Brian Marx for his extra effort in guiding me through the revision work and helping me learn more about validation of the results and correlated data. Also, I would like to thank Dr. Robert Holmes who went beyond the role of the Graduate School representative and provided helpful suggestions to improve the manuscript.

Although not on my committee, I would also like to thank Dr. Bhaba Sarker who has shown extraordinary support and interest in my academic career.

Finally, I would like to thank my wife, Brenda, for her unwavering support and financial sacrifice throughout my graduate school tenure.

TABLE OF CONTENTS

ACKNOWLEDGMENTS-----	iii
LIST OF TABLES -----	viii
LIST OF FIGURES -----	xi
ABSTRACT-----	xix
CHAPTER 1. INTRODUCTION -----	1
1.1 Overview -----	1
1.2 Hardwood -----	2
1.2.1 Hardwood industry -----	4
1.2.2 Nature of wood -----	6
1.2.3 Major types of defects-----	8
1.2.4 Hardwood grading -----	9
1.3 Image Processing-----	10
1.3.1 Image histograms -----	12
1.3.2 Relevant image enhancement methods -----	12
1.3.3 Median filtering -----	13
1.3.4 Relevant image segmentation methods -----	13
1.4 Magnetic Resonance Imaging -----	15
1.4.1 Overview-----	15
1.4.4 Application to hardwood log defect detection -----	18
1.4.5 Other scanning methods -----	19
1.5 Objectives of Research -----	19
CHAPTER 2. REVIEW OF LITERATURE -----	21
2.1 Overview -----	21
2.2 Magnetic Resonance Imaging -----	22
2.2.1 Echo-planar imaging -----	24
2.2.2 Magnetic resonance imaging scanning of logs -----	25
2.2.3 Economic feasibility of internal scanning for logs-----	30
2.3 Other Scanning Methods and Defect Detection Algorithms-----	34
2.3.1 X-ray CT scanning of logs and boards-----	34
2.3.2 Optical scanning of boards -----	40
2.3.3 Ultrasonic, microwave, and stress wave scanning of logs and boards -----	45
2.3.4 Multisensor scanning of boards-----	47
2.4 Log Processing from Defect Information-----	48
2.4.1 Automatic lumber grading-----	48
2.4.2 Sawing optimization for boards -----	49
2.4.3 Log sawing optimization -----	54
2.4.4 Direct log to product processing-----	60

2.5 Relevant Image Processing Methods-----	61
2.5.1 Image thresholding -----	61
2.5.2 Image segmentation -----	66
2.5.3 Image registration-----	71
2.5.4 Image smoothing-----	73
2.5.5 Morphological noise reduction -----	81
2.5.6 Image contrast enhancement -----	84
2.6 Validation of Regression For Prediction Purposes -----	90
2.6.1 Regression for prediction purposes -----	90
2.6.2 Validation of regression model -----	90
2.6.3 Cross-validation-----	91
2.6.4 N-fold cross-validation-----	92
2.6.5 Usefulness of individual PRESS residuals -----	93
2.6.6 Problems of autocorrelation -----	95
CHAPTER 3. METHODOLOGY-----	96
3.1 Overview -----	96
3.1.1 Data and equipment used in the research-----	97
3.1.2 Preliminary findings-----	98
3.2 Development of Defect Detection Algorithm for Spin-echo Images -----	100
3.2.1 Overview -----	100
3.2.2 Separation of foreground and background -----	102
3.2.3 Determination of average gray levels of each pixel's neighborhood -----	103
3.2.4 Defect region seed detection -----	103
3.2.5 Growing defect regions -----	105
3.3 Automatic Determination of Parameters for Spin-Echo Defect Detection Algorithm -----	106
3.3.1 Overview -----	106
3.3.2 Averaging window size and region-growing connectivity determination ----	106
3.3.3 Manual determination of parameters for each spin-echo image -----	111
3.3.4 Regression of image statistics to algorithm parameters for automatic determination of parameters -----	112
3.3.5 Obtaining predicted algorithm parameters from regression -----	114
3.4 Comparing the Spin-Echo Defect Detection Algorithm against Thresholding Methods -----	116
3.4.1 Overview -----	116
3.4.2 Finding equivalent number of defect pixels with thresholding -----	117
3.4.3 Comparing defect results between the spin-echo defect detection algorithm and thresholding -----	117
3.4.4 Seeking better threshold levels for the thresholding method -----	118
3.5 Development of Defect Detection Algorithm for Echo-planar Images-----	118
3.5.1 Overview -----	118
3.5.2 Cropping and cleanup of echo-planar images -----	118
3.5.3 Registration of echo-planar images to spin-echo images -----	119
3.5.4 Separation of foreground and background -----	120

3.5.5 Application of spin-echo defect detection algorithm to echo-planar images-	121
3.5.6 Finding pith defect in echo-planar images -----	122
3.6 Finding the Best Type of General Image Enhancement Method for Echo-Planar Images-----	124
3.6.1 Overview-----	124
3.6.2 Measure of the best image enhancement -----	125
3.6.3 Experimental design for testing differences in image enhancement-----	127
3.7 Use adjacent slices to eliminate noise in echo-planar images -----	129
3.7.1 Overview-----	129
3.7.2 Gaussian smoothing between frames -----	130
3.7.3 Median smoothing between frames -----	131
3.7.4 Comparing adjacent slice smoothing methods -----	132
3.8 Comparing Defect Detection Capability of Processed Echo-Planar Images -----	133
3.8.1 Overview-----	133
3.8.2 Generating defect images from pre-processed echo-planar images -----	133
3.8.3 Measures of best defect detection -----	133
3.8.4 Comparing defect images-----	135
3.9 Determine Defect Detection Losses Between Spin-Echo and Echo-Planar Images	136
CHAPTER 4. RESULTS AND DISCUSSION -----	137
4.1 Overview -----	137
4.2 Results of Spin-Echo Defect Detection Algorithm-----	137
4.2.1 Separation of foreground from background-----	137
4.2.2 Image results of spin-echo defect detection algorithm -----	139
4.3 Automatic Determination of Parameters for Spin-Echo Defect Detection Algorithm -----	142
4.4 Comparison of Spin-Echo Defect Detection Algorithm to Ordinary Thresholding	154
4.4.1 Overview-----	154
4.4.2 Using thresholding to find equivalent numbers of defect pixels-----	154
4.4.3 Testing various threshold levels for defect detection-----	156
4.5 Results of Echo-Planar Defect Detection Algorithm -----	158
4.5.1 Separation of foreground and background -----	158
4.5.2 Image results of unprocessed echo-planar defect detection algorithm -----	160
4.6 Automatic Determination of Parameters for Echo-Planar Defect Detection Algorithm -----	162
4.7 Comparison of General Image Enhancement Methods on Echo-Planar Images--	174
4.8 Comparison of Proposed Image Enhancement Methods on Echo-Planar Images	179
4.9 Comparison of Defect Detection Capability of Best Enhancement Methods on Echo-Planar Images -----	184
4.9.1 Comparison of defect images using <i>SNR</i> -----	184
4.9.2 Compare defect images using undetected defect pixels-----	190
4.9.3 Compare defect images using undetected defect regions -----	190
4.9.4 Compare defect images using false alarm pixels-----	191
4.9.5 Compare defect images using false alarm regions -----	192
4.10 Visual Comparison of Defect Images-----	194

CHAPTER 5. CONCLUSIONS AND FUTURE RESEARCH -----	201
5.1 Conclusions -----	201
5.2 Other Conclusions -----	205
5.3 Improving the Research -----	207
5.4 Further Research -----	208
 BIBLIOGRAPHY -----	 211
 APPENDIX A - PROGRAM SOURCE CODE LISTINGS -----	 223
 APPENDIX B - IMAGE STATISTICS -----	 266
 VITA -----	 294

LIST OF TABLES

Table 1. 1 Comparison of Softwood and Hardwood Cellular Structure -----	3
Table 1. 2 Uses of Hardwood -----	4
Table 1. 3 Usage of Hardwood by Species - Source: Weekly Hardwood Review -----	4
Table 1. 4 Hardwood Sawmills and Planing Mills by State -----	5
Table 1. 5 Summary of NHLA Hardwood Lumber Grades -----	10
Table 2. 1 Present worth of after tax value of benefits of a MRI log scanner-----	32
Table 3. 1 Analysis of Variance for a Randomized Complete Block Design -----	129
Table 3. 2 Gaussian Smoothing Weights for $\sigma = 5$ mm-----	131
Table 3. 3 Gaussian Smoothing Weights for $\sigma = 10$ mm -----	131
Table 3. 4 Gaussian Smoothing Weights for $\sigma = 15$ mm -----	131
Table 4. 1 ANOVA of Regression of Spin-Echo Upper Normal Variation Limit -----	146
Table 4. 2 ANOVA of Regression of Spin-Echo Lower Normal Variation Limit -----	147
Table 4. 3 ANOVA of Regression of Spin-Echo Upper Region Stopping Limit -----	149
Table 4. 4 ANOVA of Regression of Spin-Echo Lower Region Stopping Limit -----	150
Table 4. 5 ANOVA of Regression of Echo-Planar Upper Normal Variation Limit -----	165
Table 4. 6 ANOVA of Regression of Echo-Planar Lower Normal Variation Limit -----	166
Table 4. 7 ANOVA of Regression of Echo-Planar Upper Region-Stopping Limit -----	167
Table 4. 8 ANOVA of Regression of Echo-Planar Lower Region-Stopping Limit-----	168
Table 4. 9 ANOVA of Regression of Echo-Planar Pith-Area Lower Normal-Variation Limit-----	169

Table 4. 10 ANOVA of Regression of Echo-Planar Pith-Area Lower Region-Growing Stopping Limit -----	170
Table 4. 11 ANOVA of SNR of Echo-Planar Images (Unprocessed and General Processing Methods) -----	176
Table 4. 12 ANOVA of SNR of Echo-Planar Images (Unprocessed and Proposed Processing Methods) -----	181
Table 4. 13 Regressions of Echo-Planar Algorithm Upper Limit Parameters Against Image Statistics of Unprocessed and Processed Echo-Planar Images-----	185
Table 4. 14 Regressions of Echo-Planar Algorithm Lower Limit Parameters Against Image Statistics of Unprocessed and Processed Echo-Planar Images-----	185
Table 4. 15 Regressions of Echo-Planar Algorithm Upper Stop Parameters Against Image Statistics of Unprocessed and Processed Echo-Planar Images-----	186
Table 4. 16 Regressions of Echo-Planar Algorithm Lower Stop Parameters Against Image Statistics of Unprocessed and Processed Echo-Planar Images-----	186
Table 4. 17 Regressions of Echo-Planar Algorithm Pith Lower Limit Parameters Against Image Statistics of Unprocessed and Processed Echo-Planar Images-----	186
Table 4. 18 Regressions of Echo-Planar Algorithm Pith Lower Stop Parameters Against Image Statistics of Unprocessed and Processed Echo-Planar Images-----	186
Table 4. 19 ANOVA of SNR of Defect Images from Unprocessed and Processed Echo-Planar Images -----	189
Table 4. 20 ANOVA of Undetected Defect Pixels from Defect Images of Unprocessed and Preprocessed Echo-Planar Images -----	190
Table 4. 21 ANOVA of Undetected Defect Regions from Defect Images of Unprocessed and Preprocessed Echo-Planar Images -----	191
Table 4. 22 ANOVA of False Alarm Defect Pixels from Defect Images of Unprocessed and Processed Echo-Planar Images -----	192
Table 4. 23 ANOVA of False Alarm Defect Regions from Defect Images of Unprocessed and Preprocessed Echo-Planar Images -----	193
Table 4. 24 Defect Detection Losses by Preprocessing on Echo-Planar Images -----	193
Table B. 1 Parameters for Defect Detection Algorithm for Spin-Echo Images -----	266

Table B. 2 Image Statistics of Spin-Echo Images-----	268
Table B. 3 Prediction of Parameters for Spin-Echo Images -----	270
Table B. 4 Comparison of Defects Results of Spin-Echo Defect Detection Algorithm Versus Ordinary Thresholding-----	272
Table B. 5 Parameters for Defect Detection Algorithm for Echo-Planar Images -----	274
Table B. 6 Image Statistics for Echo-Planar Images -----	276
Table B. 7 Prediction of Parameters for Echo-Planar Images-----	278
Table B. 8 SNR of Echo-Planar Images (Unprocessed and General Processing Methods)-----	280
Table B. 9 SNR of Echo-Planar Images (Unprocessed and Proposed Processing Methods)-----	282
Table B. 10 SNR of Defect Images from Unprocessed and Processed Echo-Planar Images-----	284
Table B. 11 Count of Undetected Defect Pixels of Defect Images from Unprocessed and Processed Echo-Planar Images -----	286
Table B. 12 Count of Undetected Defect Regions of Defect Images from Unprocessed and Processed Echo-Planar Images -----	288
Table B. 13 Count of False Alarm Defect Pixels of Defect Images from Unprocessed and Processed Echo-Planar Images -----	290
Table B. 14 Count of False Alarm Defect Regions of Defect Images from Unprocessed and Processed Echo-Planar Images -----	292

LIST OF FIGURES

Figure 1.1 The Parts of a Mature Tree-----	6
Figure 1.2 Growth Occurs As A Sheath-----	7
Figure 1.3 Earlywood and Latewood Wall Thickness-----	7
Figure 1.4 MRI at the atomic level-----	17
Figure 2.1 Example of an Influential Data Point-----	94
Figure 3. 1 MRI Spin-echo Scan -----	98
Figure 3. 2 Photo of Cross Section-----	98
Figure 3. 3 Spin-echo Scan With Knot -----	99
Figure 3. 4 Center Profile of Figure 3. 3 -----	99
Figure 3. 5 Scan with Median Filtering -----	100
Figure 3. 6 Center Profile of Figure 3.5 -----	100
Figure 3. 7 Pixels With Absolute Difference > 40 Between Original Image And Median-Filtered Image-----	101
Figure 3. 8 Profile of Absolute Difference Image Between Original Image and Median-Filtered Image-----	101
Figure 3. 9 Signed Difference Image Between Original Image and Median-Filtered Image -----	101
Figure 3. 10 Comparison of Original Image and Difference Image-----	101
Figure 3. 11 Construction of Variable Threshold Limits -----	104
Figure 3. 12 Application of Threshold Limits to Original Image-----	104
Figure 3. 13 Original Profile with Various Window Size Smoothed Profiles -----	107

Figure 3. 14 Skewness of Residuals versus Averaging Window Size -----	107
Figure 3. 15 Original Profile with Various Window Size Smoothed Profiles -----	110
Figure 3. 16 Skewness of Residuals versus Averaging Window Size -----	110
Figure 3. 17 Contrast Enhanced Echo-Planar Image (x2) -----	119
Figure 3. 18 Echo-Planar Image After Cropping and Padding-----	119
Figure 3. 19 Warped Echo-Planar Image That Registers With Spin-Echo Image in Figure 3. 20-----	121
Figure 3. 20 Spin-Echo Image Taken at Same Location as Echo-Planar Image in Figure 3. 19-----	121
Figure 3. 21 Foreground of Echo-Planar Image From Moment-Preserving Threshold -	122
Figure 3. 22 Foreground of Echo-Planar Image With Additional Morphological Smoothing Step -----	122
Figure 4. 1 Spin-Echo Frame # 15 -----	137
Figure 4. 2 Mask Image for Frame #15-----	137
Figure 4. 3 Spin-Echo Frame # 20 -----	138
Figure 4. 4 Mask for Frame # 20 -----	138
Figure 4. 5 Spin-Echo Frame # 25 -----	138
Figure 4. 6 Mask for Frame # 25 -----	138
Figure 4. 7 Spin-Echo Frame # 30 -----	138
Figure 4. 8 Mask for Frame # 30 -----	138
Figure 4. 9 Spin-Echo Frame # 50 -----	139
Figure 4. 10 Mask for Frame # 50-----	139
Figure 4. 11 Seeds for Frame # 15 -----	140
Figure 4. 12 Defects for Frame # 15 -----	140

Figure 4. 13 Seeds for Frame # 20 -----	140
Figure 4. 14 Seeds for Frame # 20 -----	140
Figure 4. 15 Seeds for Frame # 25 -----	140
Figure 4. 16 Defects for Frame # 25 -----	140
Figure 4. 17 Seeds for Frame # 30 -----	141
Figure 4. 18 Defects for Frame # 30 -----	141
Figure 4. 19 Seeds for Frame # 50 -----	141
Figure 4. 20 Defects for Frame # 50 -----	141
Figure 4. 21 Skewness for Frames 5-10 -----	142
Figure 4. 22 Skewness for Frames 11-20 -----	142
Figure 4. 23 Skewness for Frames 21-30 -----	142
Figure 4. 24 Skewness for Frames 31-40 -----	142
Figure 4. 25 Skewness for Frames 41-50 -----	143
Figure 4. 26 Skewness for Frames 51-60 -----	143
Figure 4. 27 Loss Function for Frame 10 -----	144
Figure 4. 28 Loss Function for Frame 30 -----	144
Figure 4. 29 Correlation of Image Statistics Between Frames -----	146
Figure 4. 30 Spin-Echo Frame #5 -----	151
Figure 4. 31 Manual Parameters #5 -----	151
Figure 4. 32 Predicted Parameters #5 -----	151
Figure 4. 33 Spin-Echo Frame #54 -----	151
Figure 4. 34 Manual Parameters #54 -----	151
Figure 4. 35 Predicted Parameters #54 -----	151

Figure 4. 36 Spin-Echo Frame #36-----	152
Figure 4. 37 Manual Parameters #36 -----	152
Figure 4. 38 Predicted Parameters #36 -----	153
Figure 4. 39 Spin-Echo Frame #43-----	153
Figure 4. 40 Manual Parameters #43 -----	153
Figure 4. 41 Predicted Parameters #43 -----	153
Figure 4. 42 Threshold Defects # 15-----	155
Figure 4. 43 Threshold Defects # 20-----	155
Figure 4. 44 Threshold Defects # 25-----	155
Figure 4. 45 Threshold Defects # 30-----	156
Figure 4. 46 Threshold Defects # 50-----	156
Figure 4. 47 Lower Threshold at 40-----	157
Figure 4. 48 Lower Threshold at 50-----	157
Figure 4. 49 Lower Threshold at 60-----	157
Figure 4. 50 Lower Threshold at 70-----	157
Figure 4. 51 Lower Threshold at 80-----	157
Figure 4. 52 Echo-Planar Frame #15 -----	159
Figure 4. 53 Mask for Frame #15 -----	159
Figure 4. 54 Echo-Planar Frame #20 -----	159
Figure 4. 55 Mask for Frame #20 -----	159
Figure 4. 56 Echo-Planar Frame #25 -----	159
Figure 4. 57 Mask for Frame #25 -----	159
Figure 4. 58 Echo-Planar Frame #30 -----	160

Figure 4. 59 Mask for Frame #30 -----	160
Figure 4. 60 Echo-Planar Frame #50 -----	160
Figure 4. 61 Mask for Frame #50 -----	160
Figure 4. 62 Defects - Echo-Planar #15 -----	161
Figure 4. 63 Defects - Echo-Planar #20 -----	161
Figure 4. 64 Defects - Echo-Planar #25 -----	161
Figure 4. 65 Defects - Echo-Planar #30 -----	161
Figure 4. 66 Defects - Echo-Planar #50 -----	161
Figure 4. 67 Skewness for Frames 5-10 -----	163
Figure 4. 68 Skewness for Frames 11-20 -----	163
Figure 4. 69 Skewness for Frames 21-30 -----	163
Figure 4. 70 Skewness for Frames 31-40 -----	163
Figure 4. 71 Skewness for Frames 41-50 -----	164
Figure 4. 72 Skewness for Frames 51-60 -----	164
Figure 4. 73 Correlation of Image Statistics Between Frames -----	164
Figure 4. 74 Echo-Planar Frame #26 -----	171
Figure 4. 75 Manual Parameters #26 -----	171
Figure 4. 76 Predicted Parameters #26 -----	172
Figure 4. 77 Echo-Planar Frame #33 -----	172
Figure 4. 78 Manual Parameters #33 -----	172
Figure 4. 79 Predicted Parameters #33 -----	172
Figure 4. 80 Echo-Planar Frame #34 -----	173
Figure 4. 81 Manual Parameters #34 -----	173

Figure 4. 82 Predicted Parameters #34 -----	173
Figure 4. 83 Echo-Planar Frame#53 -----	173
Figure 4. 84 Manual Parameters #53 -----	173
Figure 4. 85 Predicted Parameters #53 -----	173
Figure 4. 86 Median Echo-planar #20 -----	176
Figure 4. 87 Despeckle Echo-planar #20-----	176
Figure 4. 88 Average Echo-planar #20 -----	177
Figure 4. 89 Median Echo-planar #30 -----	177
Figure 4. 90 Despeckle Echo-planar #30-----	177
Figure 4. 91 Average Echo-planar #30 -----	177
Figure 4. 92 Median Echo-planar #50 -----	178
Figure 4. 93 Despeckle Echo-planar #50-----	178
Figure 4. 94 Average Echo-planar #50 -----	178
Figure 4. 95 Gaussian ($\sigma = 5\text{mm}$) #20-----	181
Figure 4. 96 Gaussian ($\sigma = 10\text{mm}$) #20 -----	181
Figure 4. 97 Gaussian ($\sigma = 15\text{mm}$) #20 -----	182
Figure 4. 98 Z-axis median #20 -----	182
Figure 4. 99 Gaussian ($\sigma = 5\text{mm}$) #30-----	182
Figure 4. 100 Gaussian ($\sigma = 10\text{mm}$) #30-----	182
Figure 4. 101 Gaussian ($\sigma = 15\text{mm}$) #30-----	182
Figure 4. 102 Z-axis Median #30-----	183
Figure 4. 103 Gaussian ($\sigma = 5\text{mm}$) #50 -----	183

Figure 4. 104 Gaussian ($\sigma = 10\text{mm}$) #50-----	183
Figure 4. 105 Gaussian ($\sigma = 15\text{mm}$) #50-----	183
Figure 4. 106 Z-axis Median #50-----	183
Figure 4. 107 Correlation of Image Statistics Between Zmedian Frames -----	187
Figure 4. 108 Correlation of Image Statistics Between GaussianFrames -----	188
Figure 4. 109 Correlation of Image Statistics Between XY Median Frames -----	189
Figure 4. 110 Z-axis Median Defects #15-----	194
Figure 4. 111 Gaussian Defects #15 -----	194
Figure 4. 112 XYMedian Defects #15-----	195
Figure 4. 113 Unprocessed Defects #15-----	195
Figure 4. 114 Spin-Echo Defects #15-----	195
Figure 4. 115 Z-axis Median Defects #20-----	195
Figure 4. 116 Gaussian Defects #20 -----	195
Figure 4. 117 XYMedian Defects #20-----	196
Figure 4. 118 Unprocessed Defects #20-----	196
Figure 4. 119 Spin-Echo Defects #20-----	196
Figure 4. 120 Z-axis Median Defects #25-----	196
Figure 4. 121 Gaussian Defects #25 -----	196
Figure 4. 122 XYMedian Defects #25-----	197
Figure 4. 123 Unprocessed Defects #25-----	197
Figure 4. 124 Spin-Echo Defects #25-----	197
Figure 4. 125 Z-axis Median Defects #30-----	197

Figure 4. 126 Gaussian Defects #30 -----	197
Figure 4. 127 XYMedian Defects #30-----	198
Figure 4. 128 Unprocessed Defects #30-----	198
Figure 4. 129 Spin-Echo Defects #30-----	198
Figure 4. 130 Z-axis Median #50-----	198
Figure 4. 131 Gaussian Defects #50 -----	198
Figure 4. 132 XYMedian #50-----	199
Figure 4. 133 Unprocessed Defects #50-----	199
Figure 4. 134 Spin-Echo Defects #50-----	199

ABSTRACT

Identification of defects such as knots in logs before the cutting operation would allow lumber mills to maximize the value of lumber from each log. This dissertation presented images obtained from scanning an oak log with magnetic resonance imaging (MRI). The unique characteristics of MRI images of hardwood logs were noted and were used to derive a quick algorithm to isolate defects. Defect regions had some pixels that varied considerably in intensity from their neighborhood, providing a seed for initiating the defect region. There was an overlap between the pixel gray level of the defects and clear wood. Therefore, traditional thresholding techniques did not cleanly separate these regions. In this study, region-growing methods were used to extract the defects. The algorithm grew the defect region seed until the border-pixel gray levels approached the average level of the neighborhood. The region-growing methods obtained more accurate defect regions than thresholding methods because of the simultaneous consideration of gray level and adjacency information.

Two methods of MRI imaging were considered: spin-echo and echo-planar. Spin-echo imaging provided clear, detailed images but required about 20 seconds of acquisition time, which was too slow to be used in a production environment. Echo-planar images could be acquired in about 1/2 second, which was fast enough for production, but the images were fuzzy and noisy.

The dissertation presented an algorithm that found the defect regions in spin-echo images. Region-growing methods use a number of parameters and the best parameters

were unique for each image. However, common image statistics could be used to predict the proper parameters.

The dissertation also presented an algorithm that found most of the defect regions in echo-planar images. Enhancing the echo-planar images using common general-purpose image-enhancement techniques failed because the lack of discrimination allowed the process to smooth image structures as well as noise. By taking advantage of the structure of a tree, smoothing between MRI frames accomplished the goal of smoothing along homogeneous areas and not across image structures. This “z-axis” smoothing enhanced the echo-planar image visually and reduced the number of false alarm defect regions.

CHAPTER 1

INTRODUCTION

1.1 Overview

Identification of defects in logs before the sawing operation allows a lumber mill to optimize the value of usable lumber from each log. Normally, a log is graded by outward appearance to determine the cutting plan. However, most wood defects are internal and the exact locations are not detected until the sawing operation has already begun. The sawyer is forced to make instantaneous adjustments of the cutting plan as the log defects present themselves in an attempt to maximize the value of the lumber from a log. If these defects were identified prior to the sawing operation, it would be possible to plan the optimal cutting plan to produce higher quality lumber.

A study has shown that the optimal orientation of a log for its first cut versus the orientation used by current methods that rely on exterior indications produces an average 11% increase in value of lumber from logs (Steele *et al.* 1994). It has been estimated that an 11% increase in value represents roughly a \$430 million dollar increase in the value of hardwood lumber produced on an annual basis in the United States (Chang *et al.* 1989). Several studies have shown that magnetic resonance imaging (MRI) imaging can provide the locations of internal defects (Wang and Chang 1986, Hall and Rajanayagam 1986, Chang 1987, Hailey and Swanson 1987, Chang *et al.* 1989, Flibotte *et al.* 1990, Quick *et al.* 1990, Chang *et al.* 1991, Chang and Guddanti 1993). A MRI scan provides an internal view of a slice of the log. These slices can be accumulated to build a three dimensional model of the log from which an optimal cutting plan can be devised. One of the most common methods of MRI imaging is the spin-echo method.

This method produces good image results but the time for each slice is approximately 20 seconds. To detect small defects, the slices would have to be about 10 mm apart (1/4 inch). At 20 seconds per slice, the process would be too slow to be commercially practical.

A newer method of MRI imaging is the echo-planar method. This method can produce an image within half a second that is 40 times quicker than the spin-echo method and is commercially practical. The main drawback is that the image quality is noisy and fuzzy compared to the spin-echo method.

A study has also indicated that the development of algorithms for image interpretation, which is the main thrust of this dissertation, is one of the required research areas that would allow the MRI technology to be adopted by the lumber industry thus improving lumber production efficiency (Chang *et al.* 1989). The goal of this research is to develop image interpretation algorithms for the spin-echo method and the echo-planar method, then to demonstrate that the quicker echo-planar method despite its poorer image quality is very useful for defect detection.

1.2 Hardwood

Trees are in the botanical division called spermatophytes (meaning seed plants). Trees are further divided into two main subdivisions: gymnosperms (naked seeds) and angiosperms (seeds in fruit). Under these subdivisions are various orders. From the commercial viewpoint, the most important order under gymnosperms is the coniferales (softwoods) and the most important class under the angiosperms is the dicotyledon (hardwoods).

Besides the naked seed characteristic, softwood trees are characterized by needlelike leaves, the “evergreen” quality of leave production and most softwood species produce a scaly cone where the seeds are produced. Examples of softwoods include pine, cedar, juniper, cypress, fir, hemlock, spruce, larch, and redwood.

In contrast, hardwood trees produce seeds inside of fruit, pods, or acorns. They are also characterized by broad leaves that will annually change color and drop in temperate zones around the world. Examples of commercially important hardwoods in the United States include oak, popular, ash, gum, cherry, maple, alder, basswood, walnut, cottonwood, hackberry, hickory, pecan, birch, beech, tupelo, and elm.

The wood produced by hardwood and softwood trees is also different from a morphological standpoint. Table 1.1 below shows the differences in the cellular makeup of softwoods and hardwoods.

Table 1. 1 Comparison of Softwood and Hardwood Cellular Structure

Cell Type	Volume in Hardwood	Volume in Softwood
Fiber tracheid	15-60%	90-95%
Vessel elements	20-60%	0%
Longitudinal parenchyma	0-24%	0-2%
Ray parenchyma	5-30%	5-7%

Hardwoods are noted for the large number of vessel cells that are either nonexistent or rare in softwoods. With softwoods, about 95% of the volume is composed of fiber cells (longitudinal trachieds) that are much longer and more numerous than hardwood fiber cells. This characteristic makes softwoods especially suitable for paper and hardwoods unsuitable for paper.

1.2.1 Hardwood industry

Hardwood is used for a variety of products including lumber, flooring, dimension, millwork, veneers, plywood, pallets, poles and pilings. The uses of hardwood by volume are shown in Table 1.2 (Chang, 1992).

Table 1. 2 Uses of Hardwood

Use	Percent
Pallets, boxes, and containers	34
Furniture	29
Cross ties	13
Flooring	9
Decking	8
Miscellaneous	7

There are about 100,000 species of hardwoods in the world of which there are 1000 species in the United States of which 100 species are commercially important. The proportion of the hardwood production by species is given below in Table 1.3 (Chang, 1992).

Table 1. 3 Usage of Hardwood by Species - Source: Weekly Hardwood Review

Species	Percent	MBF
Red Oak	36.82	3,534,720
White Oak	15.11	1,450,560
Poplar	11.20	1,075,200
Ash	4.61	442,560
Sap Gum	4.32	414,720
Cherry	3.91	375,360
Soft Maple	3.86	370,560
Hard Maple	3.77	361,920
Alder	2.91	279,360
Basswood	2.18	209,280
Walnut	1.87	179,520
Cottonwood	1.74	167,040
Hackberry	1.29	123,840
Hickory	1.20	115,200

table continued on next page

Species	Percent	MBF
Pecan	0.95	91,200
Birch	0.68	65,280
Beech	0.44	42,240
Tupelo	0.32	30,720
Elm	0.03	2,880
Other Species	3.00	288,000
Total	100.0	9,600,000

Aside from the hardwoods grown in the United States, the U.S. imports only 4% of all the tropical timber products traded globally in 1989. In 1993, the value was \$533 million, which broke down into \$412 million for hardwood plywood, \$99 million for lumber, \$20 million for veneer, and \$2 million for logs. The most common species is mahogany which represents 59.0% of the value of all imports, followed by meranti, 12.7%, and teak, 11.7% (Smith *et al.* 1995). The hardwood industry is widespread in the United States as shown by the Table 1.4 (Chang 1992).

Table 1. 4 Hardwood Sawmills and Planing Mills by State

States	Number of Sawmills and Planing Mills	States	Number of Sawmills and Planing Mills
Pennsylvania	403	Texas	118
North Carolina	380	Maine	107
Virginia	322	South Carolina	103
Tennessee	299	Minnesota	100
Missouri	283	Louisiana	93
Kentucky	257	New Hampshire	79
West Virginia	202	Illinois	63
Arkansas	202	Massachusetts	48
New York	201	Vermont	45
Alabama	197	Iowa	29
Wisconsin	196	Maryland	28
Mississippi	187	Connecticut	15
Georgia	186	New Jersey	12
Michigan	153	Rhode Island	7
Indiana	143	Delaware	6
Ohio	139	TOTAL	4603

1.2.2 Nature of wood

Wood is a common name for what is referred to technically as xylem. The main four parts of a tree stem are outer bark, phloem (inner bark), cambium (the living part of the tree that produces new cells), and the xylem. Xylem is further broken down into sapwood and heartwood. Sapwood is distinguished from heartwood by the living parenchyma cells whereas all cells in heartwood are dead. The formation of heartwood is due to the buildup of biochemical waste materials that are polyphenolic. These chemicals are given the common name of extractives because they are often extracted from the heartwood to produce oils, waxes, gums, and resins. Some of these extractives are toxic which gives heartwood better rot and insect resistance. Some of the extractives are dark colored which often give heartwood a darker color than sapwood. The gums and waxes can inhibit moisture movement and make heartwood more difficult to dry. The locations of the parts of a tree stem are shown in Figure 1.1.

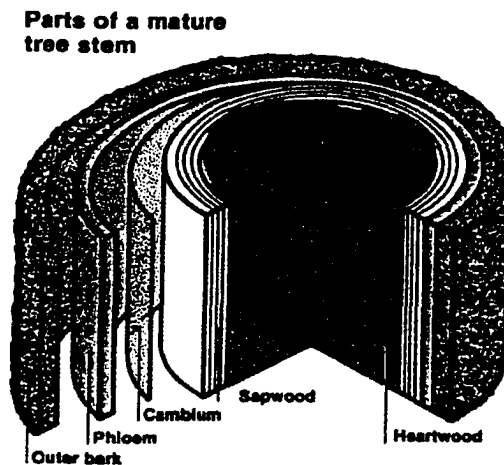


Figure 1.1 The Parts of a Mature Tree

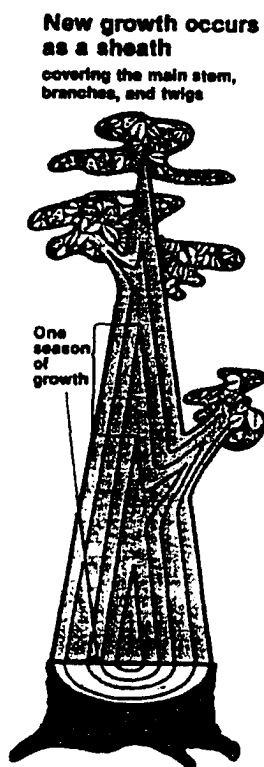


Figure 1.2 Growth Occurs As A Sheath

The growth of a tree occurs at the cambium layer. This layer completely sheaves the tree just under the bark layers which causes a tree to add growth only to the outside layer. Thus stationary foreign objects such as nails or even dead branches will eventually become encapsulated by newer layers of growth. Figure 1.2 illustrates how trees increase their volume as growth occurs.

The annual addition of growth to the outer edge of the xylem creates the well-known growth ring or annual ring. In temperate

climates, growth

occurs vary rapidly in the spring and slows down in the late summer before ceasing in the fall. During the early, rapid-growth period in the spring, the new xylem cells have thin cell walls and large

lumens (central cell cavities). During the late, slow-growth period of the summer, newly formed xylem cells have thick cell walls and small lumens. The relative difference between earlywood and latewood are shown in Figure 1.3. Because the specific gravity of the cell wall is about 1.5 and the lumen is filled with a mixture of air and water (in

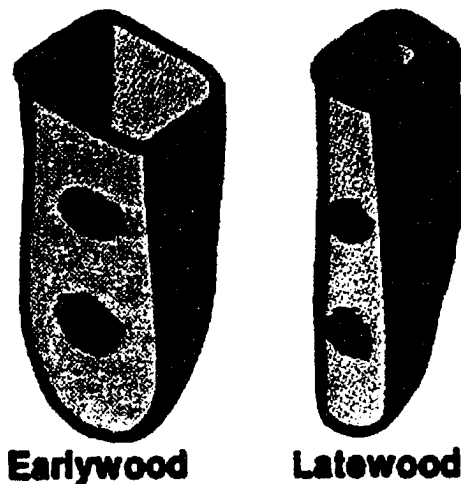


Figure 1.3 Earlywood and Latewood Wall Thickness

“green” wood) which has a specific gravity of 1.0 or less, the thick-walled, small lumen cells from the late season growth are denser and appear darker than the thin-walled, large lumen cells from the early season growth. This produces the alternating bands of light and dark wood that is a recognizable feature of annular growth rings.

1.2.3 Major types of defects

The following defects are used by the National Hardwood Lumber Association (NHLA) in grading lumber: bark pockets, checks (cracks), decay, grub holes, holes, knots, mineral streak, pith, shake, splits, stain, wane, and worm holes.

A study using comparison-of-means tests to look for differences in defects between grades found that knots, stain, and decay were the most important defect types encountered in red oak lumber. Considered to be of minor importance were bark pockets, splits, mineral streak, holes, grub holes, and worm holes. The defect types with the most defects per board foot were knots, checks, and wane. The number of defects per board feet was found to be a good indicator of lumber grade (Harding *et al.* 1993).

In addition to defects that are important for the visual grading of lumber, there are other defects in logs that are often the root causes for the visual defects in lumber. These include reaction wood (tension wood in hardwoods) and wetwood. Reaction wood is the wood formed from a leaning tree. The wood on the outer side of a leaning tree is subjected to tension forces. The properties of the wood are affected adversely. The only visual clues of tension wood in cut logs come from elliptical growth ring patterns. Because tension wood has greater longitudinal shrinkage over normal wood, lumber will

warp or twist or split during the drying process. Also, the machining qualities of tension wood are very poor (Haygreen and Bower 1989).

Wetwood is an abnormal condition caused by an invasion of anaerobic bacteria invading the tree trunk primarily through the root system. The most common distribution of wetwood is in the heartwood and inner sapwood. As wetwood progresses upward into the tree trunk, it tapers in toward the core of the tree. This results in a cone shaped volume of wetwood surrounded by normal wood. Visually, wetwood often appears as normal wood, but it may be darker. The moisture content is almost always greater than that of normal wood in the same tree. When a tree becomes infected with wetwood, there is no outward sign of the condition (Pettersen *et al.* 1993). Lumber from the tree may also appear normal, but when the green wood is kiln-dried, it usually develops checks, ring failure, or honeycomb and large economic losses result.

Oak lumber containing wetwood is more prone than normal oak lumber to develop honeycomb, ring shake, and deep surface checks. It is difficult to recognize the presence of wetwood in lumber on the green chain during mill operation; therefore, the drying defects that develop in oak wetwood are unexpected. Wetwood limits the potential for lumber drying at accelerated schedules (Ross *et al.* 1994).

1.2.4 Hardwood grading

The NHLA writes the rules for grading hardwood lumber in the United States and Canada. The rules are also used in other countries that export lumber to the United States. The most common lumber grades for hardwoods are: First-and-seconds (FAS), First-and-seconds-one-face (FIF), Selects (SEL), Number 1 Common (1C), Number 2 Common (2C), Number 3A Common (3AC), Number 3B Common (3BC). The major

determinants for a grade assignment are the overall size of the board, the percentage of the total area of the board that can be used for furniture parts, and the size of the clear areas. After finding the total size of the board, the grader determines the poorest side of the board. From that side he determines the size of the clear rectangular cuttings that the board can yield. The smallest rectangular cuttings that can be used for this step of the inspection are specified by the rules. Finally, the proportion of the board that is covered by these clear cuttings is determined. On the basis of the overall size of the board, the clear cutting sizes and the proportion of the total area that the clear cuttings cover, the board is categorized according to a set of rules that is summarized in Table 1.2.

Table 1. 5 Summary of NHLA Hardwood Lumber Grades

Grade Name	Required Yield	Minimum Size of Cuttings Used For Calculation of Yield	Minimum Allowed Size of Board
FAS	83%	4 in. x 5 ft. or 3 in. x 7 ft.	6 in. x 8 ft.
Select	83%	Similar to FAS	4 in. x 6 ft.
No.1 Common	66%	3 in. x 3 ft. or 4 in. x 2 ft.	3 in. x 4 ft.
No.2 Common	50%	3 in. x 2 ft.	3 in. x 4 ft.
No.3A Common	33%	3 in. x 2 ft.	3 in. x 4 ft.
No.3B Common	25%	3 in. x 2 ft.	3 in. x 4 ft.

1.3 Image Processing

An image refers to a two-dimensional (2-D) light intensity function $f(x,y)$ where x and y denote spatial coordinates and the value of f at any point (x,y) is proportional to the brightness (or gray level). The coordinate system most often used sets the $(0,0)$ point in the left top corner and the x values increase toward the right and the y values increase in a downward direction. This system follows the one used by 2-D matrices. A digital

image is an image that has been discretized both in spatial coordinates and brightness. Digital images are most often defined as a 2-D matrix of integers where the location of the point is given by the column (x) and row (y) indices of the matrix and the matrix element value defines the gray level at that point. The elements that make up the image matrix are called pixels, which is an abbreviation for picture elements.

Although images can be any size and have any number of gray level resolutions, there is a tendency in image processing to work with images that are square or at least rectangular and have a number of possible gray levels and dimension sizes that are a power of 2 (such as 16, 64, 128, 256, 512, etc.). Although people can distinguish only about 30 gray levels, many input devices can distinguish much greater numbers of gray levels. Because 256 levels is usually adequate for most tasks and 256 levels is the largest number that can be represented by an 8-bit byte which allows for convenient storage, much of image processing work is done with 256 gray levels. As a reference, an image from a television screen has been compared to a 512 x 512 image with 128 gray levels (Gonzales and Woods, 1992).

Connectivity between pixels is a concept used for boundaries and for region definitions. In two dimensional images, the 4-neighbors of a pixel p at coordinates (x,y) have the coordinates $(x+1, y)$, $(x-1, y)$, $(x, y+1)$, and $(x, y-1)$. These neighbors are said to be 4-connected to p . Similarly, there are four diagonal neighbors of p at coordinates $(x+1, y+1)$, $(x+1, y-1)$, $(x-1, y+1)$, and $(x-1, y-1)$. These diagonal neighbors along with the 4-connected neighbors constitute the 8-connected neighbors of p .

1.3.1 Image histograms

The histogram of a digital image in L gray levels in the range $[0, L-1]$ is a discrete function given by $p(r_k) = n_k/n$, where r_k is the k th gray level, n_k is the number of pixels in the image with the k th gray level and n is the total number of pixels in the image. Histogram equalization refers to a monotonic remapping of the pixel gray levels so that the histogram of the image is flat and the gray levels follow a uniform distribution (i.e., $p(r_k) = 1/L$).

1.3.2 Relevant image enhancement methods

Image enhancement methods include contrast stretching. Contrast stretching is useful where an area may have many pixels with similar gray levels resulting in low contrast. If the gray levels were remapped so that they differ widely from one another, the appearance would be an image of stark contrast. One method to maximize contrast across the image as a whole is the aforementioned histogram equalization method. This method also has the advantage of normalizing an image so that the overall gray levels are the same as all other images with equalized histograms. The same object recorded under two different image acquisition systems or by the same system under different conditions can be adjusted by the histogram equalization method to produce equivalent images.

If noise is uncorrelated and has a zero average value, then a noisy image can be improved by averaging a number of acquisitions of the same scene. Given a noisy image $g(x,y)$ formed by the original image $f(x,y)$ and the addition of noise $\eta(x,y)$, $g(x,y) = f(x,y) + \eta(x,y)$. If there are M acquisitions, then averaging the images will yield:

$$\bar{g}(x, y) = \frac{1}{M} \sum_{i=1}^M g_i(x, y) \Rightarrow E\{\bar{g}(x, y)\} = f(x, y)$$

$$\sigma_{\bar{g}(x, y)}^2 = \frac{1}{M} \sigma_{\eta(x, y)}^2 \Rightarrow \sigma_{\bar{g}(x, y)} = \frac{1}{\sqrt{M}} \sigma_{\eta(x, y)}$$

So, the averaged images will yield an image that becomes closer to the original noise-free image. Recently, a number of researchers have devised adaptive and other non-linear methods for image enhancement and noise removal.

1.3.3 Median filtering

Median filtering is widely used for smoothing and enhancing an image. Many smoothing filters tend to blur sharp image details. In median filtering, the gray level of each pixel is replaced by the median of the gray levels in the neighborhood of that pixel. Because any isolated outlier in the neighborhood will not have an influence on the median, the median filter has a very desirable property of rejecting the isolated outliers from an image. This is particular important if images are corrupted by pulse or spike-like noises, sometimes called salt and pepper noise. If there is a detail that is smaller than half the size of the median filter window, then the median filter will remove the detail as it removes an isolated noise pulse. Details like thin lines and sharp corners are often removed by median filters. Consequently, researchers have devised numerous methods that alleviate the drawback. The new methods are covered in the review of the literature.

1.3.4 Relevant image segmentation methods

Segmentation divides an image into its constituent parts. The degree of segmentation is dictated by the task at hand. One of the most important methods is called thresholding. A thresholded image, $g(x, y)$, by a global threshold, T , is defined as:

$$g(x, y) = \begin{cases} 1 & \text{if } f(x, y) > T \\ 0 & \text{if } f(x, y) \leq T \end{cases}$$

There are many methods for choosing the threshold level. In addition, the global use of more than one threshold, multiple thresholding, can also be applied to separate an image into its constituent parts. Also, rather than using a global threshold, another possibility is to use local-based thresholding where individual thresholds are applied to different segments of an image.

If the image histogram is bimodal it is assumed that the background creates one mode and the foreground object creates the other mode. On that assumption, the threshold between the modes is likely to separate an object from a background. Several automatic methods exist. These include the basic P-tile method, and the moment-preserving method.

To use the P-tile method one must have some a priori knowledge of the size of the foreground object. If size of object is known a priori and is lighter than the background and the proportion of the object area to the total area is P, then the correct threshold is the one that maps the highest P gray level percentile pixels to the object.

The moment-preserving method uses the histogram to compute a first moment from the origin. This moment is defined as the sum of the heights of the histogram bins (the number of pixels that belong to a particular gray level) weighted by their respective gray level values. The threshold is then chosen so that the average gray level of the two classes multiplied by the number of pixels in each class attains the same moment before the thresholding operation. There are a number of alternative methods in automatic thresholding that are covered in the review of the literature.

Region-oriented segmentation is another important set of methods in image segmentation. Let R represent the entire image region. Then segmentation is a process that partitions R in n subregions, R_1, R_2, \dots, R_n such that

$$(a) \bigcup_{i=1}^n R_i = R$$

(b) R_i is a connected region, $i = 1, 2, \dots, n$

(c) $R_i \cap R_j = \emptyset$ for all i and j , $i \neq j$

(d) $P(R_i) = \text{TRUE}$ for $i = 1, 2, \dots, n$

(e) $P(R_i \cup R_j) = \text{FALSE}$ for $i \neq j$

where $P(R_i)$ is a logical predicate over the points in R_i which represents a condition for belonging to a region and \emptyset is the null set.

Region growing by pixel aggregation is a method that directly applies the conditions for regions mentioned above. The method starts with “seeds” which have previously been established. The method then tests neighboring pixels of the seeds which automatically satisfies condition (b) by applying a test for condition (d) and also applying condition (c) by making sure the pixel does not already belong to another region. If the pixels pass, then they are added to the region. The process continues until no regions can be added. Neighboring regions are tested for merging by applying condition (e). This method assumes that the “seed” placement procedure is accurate.

1.4 Magnetic Resonance Imaging

1.4.1 Overview

Magnetic resonance imaging (MRI) is very useful in the medical field because it represents the only accurate non-ionizing radiation method of internal scanning. Since its

introduction to the medical field, MRI has been growing rapidly. In 1991, medical image market was \$8 billion of which \$1 billion was MRI and the MRI market was increasing at 20%/yr. MRI can give three dimension views inside of objects. 3-D voxels (volume elements) are the analog of the 2-D pixels (picture elements) the simplest 3-D model consist of successive 2-D images (cross-sections) (Ayache 1995). Magnetic resonance methods unlike X-ray methods are especially suitable for highlighting different tissues even though the tissue density may be identical.

Magnetic resonance equipment consists of a tunnel-like magnet that sets up a magnetic field around the object to be scanned. The most common element that is scanned by MRI machines is hydrogen. Hydrogen atoms have a nuclei made up of a single proton. Normally, the protons are spinning and the axis of the spin is in a random direction as shown in Figure 1.4 A. MRI is initiated by first aligning the nuclei of the object with a powerful external magnetic field. As the nuclei are aligned, they precess or wobble about in the direction of the external magnetic field at a specific rate called their Larmor frequency $\omega(X)$. This frequency is proportional to the strength of the magnetic field $B(X)$, and is also dependent on the gyromagnetic ratio r , which is unique for each kind of atomic element. So the precess rate can be expressed as $\omega(X) = rB(X)$. Nuclei from different elements have different gyromagnetic ratios so that in the same magnetic field they precess at different frequencies. So all hydrogen atoms can be distinguished because of their unique precess frequency at a given magnetic field strength.

The precess state is shown in Figure 1.4 B. When a radio pulse with the same identical frequency as the Larmor frequency is sent out into the object, the radio pulse excites the protons and causes them to change alignment from the external magnetic

field. This state is shown in Figure 1.4 C. After the radio pulse is halted, within milliseconds, the protons spiral back into alignment with the external field, releasing their own faint radio signal at the same Larmor frequency. This state is shown in Figure

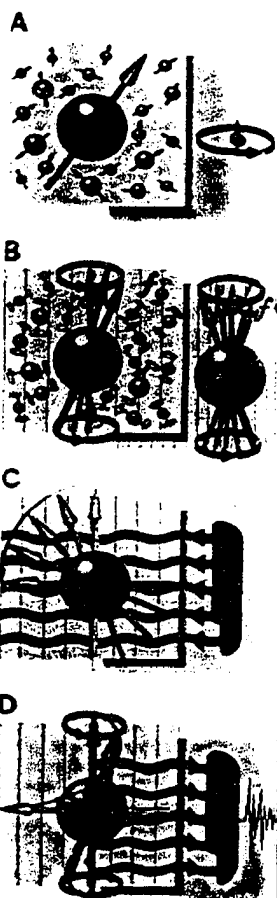


Figure 1.4 MRI at the atomic level

Together these relaxation times reflect the environment of the object under study at the molecular level.

Theoretically, any kind of nucleus with an odd number of neutrons and/or protons can be imaged. However hydrogen is the most often used nucleus in today's commercial

1.4 D.

Although the intensity of the MRI signal depends on the density of the resonating nuclei, the duration of the MRI signal is completely determined by the chemical environment of the nuclei. After the radio pulse is turned off, the nuclei simultaneously return to the equilibrium alignment with the external field and give off their own radio-frequency signal. These processes are exponential in time. The exponential time constant associated with returning to equilibrium is called the spin-lattice time and it denoted by T_1 . The exponential time constant associated with the signal decay is called the spin-spin time and is denoted by T_2 . In general, for a given chemical environment, $T_1 > T_2$. Also, it is worth noting that T_1 (solids) $> T_1$ (liquids) and T_2 (liquids) $> T_2$ (solids).

scanners because of its sensitivity and abundance. Most of the scanners have a magnetic field strength of 0.15 to 2.0 tesla (1 tesla equals 10^4 gauss) (Kaufman *et al.* 1981).

To make a MRI slice image, the overall magnetic field is varied in the Z-direction. At this point each cross-sectional slice of the object has a correspondence to a unique external magnetic field strength. Since the proton precess rate is proportional to the magnetic field strength, the protons in each slice precess at a unique rate. Then a radio frequency signal tuned to an exact frequency will only excite the protons that are located in one cross-section.

After the radio pulse is turned off, if nothing else were done the signal intensity would relate to all the protons from the whole cross-section. To further spatially encode the emitted signal, a small magnetic field gradient is imposed during the relaxation process. Such a gradient causes nuclei at unlike positions in the cross-section to emit signals at different frequencies. The magnitude reveals the density and the frequency reveals the location.

1.4.4 Application to hardwood log defect detection

In wood, the MRI parameters are set to emphasize the T_1 and T_2 time constants of the free water in the log. The isolation of water from the wood material is possible due to long relaxation times of water as compared to hydrogen protons in other wood components. In essence, a MRI scan can be adjusted so that it becomes a profile of moisture content instead of a profile of the hydrogen content. Researchers have found that MRI has great potential for the study of wood. New developments are covered in the review of the literature.

1.4.5 Other scanning methods

The other method for scanning logs most often cited in the literature is X-ray computerized tomography (CT). This method involves rotating an X-ray tube 360° around the object to obtain many views. The views are all assembled through a process called back projection to obtain the cross-sectional slice. The results are very sharp and quick. However, the object is exposed to appreciable amounts of radiation. X-rays measure density only and the final CT number does not allow the separation of the different X-ray absorption contributions due to different components in a voxel. Experiences with log defect detection with X-ray CT scanning is covered in the review of the literature.

Ultrasonics has been mentioned a few times for possible use in internal defect detection. The advantages of simple and inexpensive equipment are countered by the poor image quality, false readings, and the difficulties in obtaining good sensor contact with the material.

1.5 Objectives of Research

The overall goals are:

- (1) Obtain a set of algorithms that identify internal defects from MRI spin-echo images.

The images used in the study show that every defect has at least some pixels that vary considerably from the median gray level in the image neighborhood. These outlying pixels provide a seed or starting point where the region of the defect can be obtained. However, there is an overlap between the gray levels of the defects and the gray levels of clear xylem due to the presence of annular rings. Therefore, straight-

forward thresholding techniques will not cleanly separate the defect region from the clear xylem. It is expected that the region growing method starting with the outlying pixels (outlying from a gray level standpoint) and stopping before reaching the median gray levels will obtain very accurate defect region borders compared to thresholding methods.

- (2) Obtain a set of algorithms that identify the same set of internal defects from MRI echo-planar images that are found from the spin-echo images.

The echo-planar images contain noise that can be mistaken for small defects since the noise can also vary considerably from the median gray level in the image neighborhood. It is expected that this noise occurs at random locations in the image. Defects in the xylem will be detected in subsequent MRI slices in approximately the same locations because the MRI slices are stacked closely along the longitudinal axis of the log. Random noise that could be mistaken for defects can be almost virtually eliminated by comparing the immediate previous and subsequent MRI slices. The subsequential processing of the echo-planar image would easily follow with the removal of the noise.

- (3) Compare and explain any differences between the defects obtained from the two methods of MRI imaging.

Very small defects might appear in only one MRI slice depending on the size of the defect. If the defect appeared as one or a few pixels on just one slice, then such a defect could be indistinguishable from random noise. Also, any other variations such as the variations between the borders of the same defect from the two methods of MRI imaging would be reported and explained.

CHAPTER 2

REVIEW OF LITERATURE

2.1 Overview

The field of image processing is very broad and has found applications in many other fields of study. For that reason, image processing is covered in a multitude of journals. The scope of this literature review has to necessarily be limited to the areas and methods that are most relevant to scanning for internal defects. The first section that follows will explore the progress in using magnetic resonance imaging for scanning logs.

The second section will explore the progress made in scanning logs with other methods. The purpose of looking at other scanning methods is that the work done in this related area can often provide clues and direction for proceeding with magnetic resonance scanning of logs.

The third section covers the progress made in automated grading of lumber, sawing optimization, and direct log-to-parts processing. While these areas are beyond the scope of the dissertation, they represent the natural end uses of the internal scanning procedures and simultaneously give direction for the output required from the scanning process as well as demonstrate the importance of developing fast internal scanning algorithms.

The fourth section is devoted to the recent progress made in general image processing techniques that will find application in internal scanning. The area of automated thresholding is important because there is a need to quickly separate the log

image from its background. Image segmentation is required for segmenting the defect areas from the clear wood areas. Image registration is important because this dissertation is concerned with two sets of images acquired by different methods but made from the same subject. Also, many studies have shown the multisensor or multispectral approaches often provide the most reliable results for classification tasks. Image smoothing is important for cleaning up noisy images such as the echo-planar images. Noise may have to be removed if not to interfere with automatic defect detection at least to provide usable visual representations. Mathematical morphological operations are covered because they provide one of the most promising approaches to automated noise removal and region border cleanup. Contrast enhancement is covered because the echo-planar images used in this study in particular have poor contrast and in general most images can provide more information to human users after contrast is enhanced.

The last section covers methods of validating regression results for the purpose of prediction. The common regression performance measures are indications of fit to the existing data. These measures were not designed to indicate prediction capability. This section describes the traditional method of validation called data-splitting. It also goes on to describe the cross-validation method that is used when the available data set is small. The most accurate cross-validation method, N-fold cross validation or PRESS residuals, is described next. Finally, problems with autocorrelation of data is discussed along with the first-order autocorrelation model.

2.2 Magnetic Resonance Imaging

To obtain spatial discrimination a magnetic field gradient is superimposed on the object. The distribution of protons along one direction can be obtained by Fourier

transformation. To obtain 2-D, the direction of the gradient can be rotated and a back-projection algorithm used to reconstruct the data. This is the projection reconstruction method. A second method encodes one axis just as in projection reconstruction, but instead of rotating the gradient, orthogonal axis information is obtained by phase encoding. The data are reconstructed by a 2-D Fourier transformation. Let N be number of projections, and 2-D Fourier transformation N is number of rows in an image. The number of elements on each line can be large; along the first Fourier transform direction, the number of points/projection depends upon frequency resolution and gradient strength, but the imaging time is not affected by the number of elements per line. The repeated projections can not be obtained too rapidly. The reason is that nuclei in different tissues return to equilibrium at different rates (the T_1 relaxation time). Let b be this time. To improve the signal-to-noise ratio, SNR , the projections should be averaged over n acquisitions. So image time, $t = bNn$. If there are s sections, then image time, $t = bNns$. In planar imaging, Crooks *et al.* (1984), found that the signal to noise ratio was proportional to a number of factors, i.e., $SNR = ScV\sqrt{Nn}$ where S is the signal, c is an instrument constant, V is voxel size. So, in order to double resolution, N could be doubled, which decreases V to 1/4 of its original size. Thus SNR is scaled by $1/4 \times \sqrt{2} = 0.35$. So doubling resolution results in a SNR that is only 0.35 times the previous SNR and causes a doubling of acquisition time. Therefore, there is a substantial penalty for seeking increased resolution. Also, the time to reconstruct images and to do any processing is affected by the size of the image. So for log images, only the smallest resolution that will accomplish the task at hand should be used. The same study revealed

that for the interval b to be utilized to full advantage, while waiting for magnetization to recover, other sections of the object could be imaged. In this manner, 15 sections could be imaged in the time of one without compromising performance. They predicted that the levels of spatial resolution that will be available in MRI would eventually be limited by patient motion.

Wang and Lei (1994) concluded that it was expected that two spin density volumes relatively far apart on the cell lattice were statistically independent. According to the generalized central limit theorem, regardless of the probability description of the microelements involved, the global magnetization appeared to be Gaussian. Thermal noise from the object being scanned was also characterized as Gaussian with a zero mean and a constant variance. By the summation-invariant property, the final MRI pixel images were summarized by the following statement: Each pixel is a random variable with a truncated asymptotic Gaussian distribution and the whole image is a Gaussian random field with multivariate normal distributions.

2.2.1 Echo-planar imaging

Iwaoka *et al.* (1984) and Ahn *et al.* (1986) noted that compared with X-ray computed tomography (CT), conventional MRI imaging systems required long scan times. This speed limitation was largely due to the spin-lattice relaxation time, T_1 , which required a certain period of relaxation before one spin system could be reexcited. Therefore, the time between pulses was generally much longer than the spin-lattice relaxation time to permit the nuclear spin system to return naturally to equilibrium. This drawback usually limited the MRI imaging to a time frame of a few minutes. Also, the

whole sequence was repeated several times and signals were averaged to improve the signal to noise ratio.

More recently, very high-speed imaging in the time frame of a few tens of milliseconds using the echo-planar technique was developed by Mansfield (1977) and later a number of variations were suggested. The short image-acquisition time of echo-planar imaging was achieved due to a simultaneous encoding of spatial information along two dimensions by applying, during data acquisition, a second constant gradient perpendicular to the switched gradient. The information necessary for a 2-dimensional reconstruction of a slice was thus obtained in a single free-induction decay signal. Although the methods appeared attractive, they suffered two main technical drawbacks. One drawback was the difficulty of realizing large high-speed gradient fields. The second drawback was the resolution limit of y directional sampling was determined by the number of echoes that could be generated by the number of alternating gradient pulses, which in turn was limited by the intrinsic MRI property, i.e., a finite T_2 decay. In addition, Mansfield's original echo-planar technique suffered anisotropic resolution in the image domain due to the different weighting of T_2 decay in directions x and y.

In short, the original echo-planar technique had several drawbacks such as impractical application and poor spatial resolution, especially y-directional resolution. Fortunately, since then, a number of researchers had proposed new methods to partially alleviate these drawbacks (Iwaoka *et al.* 1984, Bendel 1985, Ahn *et al.* 1986)

2.2.2 Magnetic resonance imaging scanning of logs

Explorations into the application of MRI to scanning wood began not long after the first MRI scans were announced. Wang and Chang (1986) examined a sample of

black cherry via MRI. Black cherry is a semi-ring porous species with gradual transition from earlywood to latewood. Nevertheless, they were able to report that the differences between earlywood and latewood are distinguishable on MRI scans. Hall and Rajanayagam (1986) used water based chemical solutions with unique T_2 values that could be imaged separately to study the penetration of water into wood as a function of time.

By 1987, Chang *et al.* (1987) reported on the results of a MRI spin-echo scan on a sample of white oak (*Quercus alba L.*) which was of a commercial size. The MRI unit used in this study was a Siemens 0.5 tesla scanner. A white oak log section, 28 cm in diameter and 60 cm long was scanned after it had been air-dried in an air-conditioned room for 50 days. The spin-echo imaging technique used a 30 msec echo time and a 150 msec repetition time. This technique emphasizes the contribution of proton density and T_1 relaxation of the log. Details such as sapwood, heartwood, annual rings, pith, knots, and wood rays were visible. As a result of moisture loss, through openings in the bark, dark radial bands throughout the sapwood indicated areas that had dried. However, the heartwood adjacent to these areas of sapwood was not appreciably affected. The knots were surrounded though not completely by thin bright (high-moisture) bands.

Hailey and Swanson (1987) reported that because the T_2 relaxation times are dependent on the chemical environment, MRI has the potential to generate separate images for water in the cell wall, earlywood lumen and latewood lumen. They reported that MRI shows the most promise in the long term over other scanning methods because of the imaging of different chemical environments and no radiation hazards. They found that the following defects could be scanned: knots, sapwood/heartwood boundary, decay,

incipid decay, moisture distribution, wet pockets, worn holes, shakes on splits, annual rings, juvenile wood, bark, resin pockets, blue stain, rocks, preservative impregnation and environmental pollutants.

Chang et al. (1989) noted that due to moisture variation inherent in wood, MRI was particularly suited for detecting internal features within wood. They reported on a study conducted on both white oak and black cherry samples that indicated that the species of wood did not affect the quality of log scanning images. Both longitudinal and cross-sectional views were taken. The cross-sectional images in the study were taken 15 images at a time 16mm apart. Image acquisition for the 15-scan group required 7.5 minutes and another 7.5 minutes for image reconstruction. Spatial resolution was 1.95 x 1.95 mm with total image area of 50 x 50 cm.

They were able to report that bark did not appear on the image due to low moisture content. However, features such as sapwood, heartwood, growth rings, pith, and some rays could be detected and the resulting image compared well with an actual view. In addition, several defects that were not detectable to the human eye were revealed by MRI. The eccentricity of growth rings associated with tension wood was revealed. Because tension wood has higher moisture than normal surrounding wood, MRI was able to determine the boundaries of the tension wood, which has never been done before. Also, they found that wetwood was detectable as it appeared brighter than normal wood.

The extensive imaging of the two species allowed a number of discoveries about the appearance of various defects in MRI images. They reported that wood exposed to the environment for some time (3 weeks) had moisture escaping from the sapwood areas underneath checks in the bark. These appeared as radial dark bands in the sapwood

areas. Gum spots could be detected with MRI. These appeared as dark spots in sapwood. Gum spots in dryer heartwood were undetectable because the moisture contents were identical. Knots tended to be dark areas with a bright band on the border. Also, knots varied in a radial direction from one cross-section to the next. Worm holes appeared bright if filled with water or very dark if empty. Decay appeared as dark areas with a jagged bright border. They also reported that for industrial purposes, progress would be required in the following areas: 1. additional samples and species must be scanned, 2. fast scanning techniques such as echo-planar method must be explored, 3. less expensive scanners would be required and 4. rapid identification and precise location of defects would be required.

Flibotte *et al.* (1990) conducted magnetic resonance experiments on various samples of wood and found that they could distinguish the signals from solid wood and water. Also, the water signals could be broken down into earlywood tracheid lumen water, latewood tracheid and ray lumen water, and bound water on the basis of spin-spin relaxation times, T_2 . Heartwood and juvenile wood had less water and shorter spin-spin relaxation times than the sapwood. The rot sample had more water so it stood out in heartwood and juvenile wood. Sapwood/heartwood boundaries could be detected but heartwood/juvenile wood was more difficult. MRI images were traced mainly to earlywood tracheid water and for that reason MRI images only showed about 60% of the water in a normal western red cedar log.

They reported that the direct measure of the moisture content of wood was possible. Because early wood tracheids have a cell lumen diameter about 3 times the latewood tracheids, early wood tracheids should show as bright areas. Oddly, the (T_2)

spin-spin relaxation times were proportional to the lumen diameter which made it possible to scan for just earlywood or latewood if desired.

In the study, the brown rot sample had a high moisture content of 235%. Most large MRI facilities were incapable of imaging protons with T_2 values less than a few msec. So they could not in general image solid wood or bound water. They found that, fortunately, T_2 values measured at 0.15 Tesla (6.4 MHz) are not different than at 2.1 Tesla (90 MHz) indicating no frequency dependence. A spin-echo image at 26 msec would contain no bound water (which has T_2 between 1 and 4 msec) and very little signal from latewood and ray lumen water (which has T_2 at about 35 msec). Also at 26 msec, about 60% of the water was imaged because the other water responded to different T_2 times. So even though the rot was very moist, the signal was not strong because of the low T_2 values.

Quick *et al.* (1990) had used MRI to study the radial moisture profile of cedar sapwood over time during drying. MRI allowed submillimeter resolution of the water distribution. They found that the drying rates for earlywood and latewood differ in the same sample. The earlywood with more water in its larger lumen, lost moisture content more rapidly then tapered off. The latewood with more bound water in its cell wall lost moisture content more slowly at first. As drying continued, the difference in the moisture content disappeared before oven-dry condition was reached. The authors noted that the bulk moisture content could be accurately determined with MRI.

Chang *et al.* (1991) reported on a fast scanning method applied to hardwood logs. Recent economic analysis had shown that for a 12 million bd.ft/year hardwood sawmill, the MRI scanner needed to scan logs at a rate of 2.5 images per second assuming that the

log was 16" diameter and scans were 1 inch apart. The Instascan method was capable of scanning at 10 images/sec. This paper compared the quality of the image with the spin-echo image. The Instascan image had a lower resolution and the pith was not visible. Also, some branches appeared only as a dark shadow in the Instascan method. Thus, off-line image enhancement processing would be a requirement for fast scanning to be used.

Chang and Guddanti (1993) went on to develop a software system that generated 3-D log images given cross-sectional images from an X-ray CT scanner. The primary purpose was to generate cut faces (longitudinal views) for logs for any angle of rotation and at any depth. The cut faces could then be processed to identify defects such as knots and worm holes on the cut-face surface. This allowed the application of National Hardwood Lumber Association (NHLA) rules to grade the lumber. The process could be repeated for various angles to obtain the best sawing sequence and log angle for the maximum value. The cross-sectional images were 256 x 256 pixel format with 256 gray scale.

The defect detection algorithm used was a simple two-threshold method. One threshold was high and above which was found high-density defects like knots. The second threshold was low, below which was found low-density defects such as decays, cracks, worm holes, and voids. The defects were boxed in a rectangle. The generation of a cut face at any angle, depth and detection of defects required a couple of seconds.

2.2.3 Economic feasibility of internal scanning for logs

The price of logs has risen dramatically. The proportion of production costs has shifted from the processes to the raw material, the logs themselves. Part of that shift was due to the increases in efficiency in hardwood sawmills but the greatest part of the shift

was due to the ever-increasing cost of hardwood logs. In 1987, the cost of logs accounted for nearly 70% of the cost of producing lumber (Szymani 1987). By 1990, it was reported that the cost of logs had increased from 20% to 80% of the total production costs of the hardwood mill over the past 30 years (Hodges *et al.* 1990). The need for more effective use of each log is becoming greater as time goes on.

By 1989, enough was known about MRI scanning of hardwood logs to make basic assumptions even though a working system was not in place. Chang (1989) presented a preliminary economic analysis for MRI scanning. He reported that the potential yield gains were significant because the sawmilling industry averaged a modest 3% profit over sales revenue.

The paper assumed the following:

- (1) 6 million board feet of hardwood logs could be scanned annually with an 8 hour shift
- (2) lumber price = \$535 MBF - Appalachian 4/4" red oak as of 8/19/89
- (3) 3 yield gains: high - 16%, medium - 12%, low - 8%
- (4) annual expenses = \$100,000 which broke down to \$30,000 salary, \$30,000 electricity, \$40,000 maintenance contract
- (5) federal and state income tax rates amounted to 40%

Based on the assumptions, the gain in lumber value was 513,600 (16%), 385,200 (12%), & 256,800 (8%). Total after tax net gain was 248,160 (16%), 171,120 (12%), & 375,379 (8%).

The present value of after tax net revenues (not the investment costs and depreciation expenses), assuming $i = 8\%$ is given in table 2.1

Table 2. 1 Present worth of after tax value of benefits of a MRI log scanner

Life of Project	16% yield gain	12% yield gain	8% yield gain
3 years	640,252	441,490	242,726
5 years	990,152	682,769	375,379
7 years	1,292,916	891,535	490,157

The net present values given above do not reflect the investment cost for the scanner itself and were intended to be compared to the initial price of a MRI scanner (minus the net present worth of the depreciation tax credits which would be substantial). The gains would be greater by assuming higher prices for cherry (\$750 MBF) and walnut (\$855 MBF). Also, as is typical in other industries, expensive equipment is put to better utilization by running it over more than one shift. Thus, better gains would be produced by assuming a 16-hour (two shift) operation or 12,000,000 bd.ft./yr. This would more than double the gain because the maintenance contract remains the same.

Considering that the prices of MRI scanners have actually dropped in the past and the prices of hardwood logs has rising for decades, the point of economic feasibility is approaching. Industrial MRI scanners would not need all the features of medical scanners. Scanning speeds of 1 image per second would be required to fulfill the assumptions of the study.

Hodges *et al.* (1990) conducted a study on the economic feasibility on X-ray CT scanning which were currently less expensive than MRI scanners. They conducted a net present worth economic analysis with the following assumptions. Initial price of equipment was \$1.2~1.5 million with a \$175,000/yr. service contract. Another \$100,000 was required for additional equipment & training. \$10,500 (6% of operating expense) was required for working capital. There were three production levels of 5, 10, & 25

million board-feet to investigate because these represented the typical classes of sawmills. The lumber currently produced was distributed by grade as follows: 12% for First-and-Seconds, 23% for No.1 Common, 27% for No.2 Common, 38% for No.3 Common and lower. Prices per grade were: \$935/mbf-FAS, \$505/mbf-No.1C, \$255/mbf-No.2C, \$195/mbf No.3 and lower. The top marginal Federal tax rate was 34%. No state tax was assumed. Minimum attractive rate of return (MARR) = 15%, The equipment was classified in the 7-yr. Modified Accelerated Cost Recovery System (MACRS) class. Annual gross income was multiplied by an overall value yield increase. Standard after-tax net present worth was computed. The overall project life was estimated at 7 years. Their results indicated that at 25 million bd.ft./yr., the required value yield increase would be 8.5%. At 10 million bd.ft./yr., the value yield increase would have to be an unlikely 32%. The main conclusion was that higher production rates which means faster image processing rates were essential for economic feasibility.

Chang (1992) provided additional information that can prove helpful in more detailed economic analyses. These included overall economic statistics on hardwood production such as: Hardwood sawmill production ranged from a few thousand to 100,000 bd. ft. each day. There were 4600 hardwood sawmills in U.S. in 1990. Annual production was 10.7 billion board ft or 21% of total lumber (50 billion board feet) produced in the United States. The annual value of hardwood was \$3.5 billion at the sawmill. More than half of all hardwood produced was oak. Current practice attempted to make inferences about internal defects from studying bark patterns. It took years of practice to be reasonable accurate. 10,000,000 bd.ft. annual production corresponded to a log/minute for an 8 hour shift. Cost estimates of a MRI scanner were \$1.5-2M and the

annual service contract was \$150,000. Annual operating cost estimates for electricity, cryogen, and miscellaneous supplies were about \$30,000.

2.3 Other Scanning Methods and Defect Detection Algorithms

2.3.1 X-ray CT scanning of logs and boards

Currently, optical scanners are used to select sawing patterns and log orientations that maximize the volume of lumber sawn from each log. Taylor et al. (1984) began investigating the use of X-ray tomography for internal scanning of logs. They found that tomographic detection of knots was more difficult because the density of knot tissue was not greatly different than that of normal wood. Detection of knots was especially difficult in freshly sawn logs where water tended to mask the location of knots.

Their paper only considered knots for the defect detection. The scan time per slice was about 1 second. The image was divided into an array of 24x24 pixels (1/4" x 1/4"). The image had background areas of nearly 99-100% black, clear wood areas of nearly 0% black and knots and juvenile wood had intermediate values. The background and the juvenile wood at the center were deleted from consideration. A preselected threshold was used to separate the knot areas from the clear wood. The results were mixed because green knots (knots with water) were hard to identify.

To achieve production speed, they recommended that logs could be segregated for scanning and no scanning. Also, they recommended that very preliminary and hopefully quick defect detection algorithms could be developed to perform a cursory examination to skip over scans which contain all clear wood. When the potential for defects was noted by the cursory examination, then the processing of scans would involve more accurate algorithms.

Funt and Bryant (1985) described a system to identify knots and rot. The first step of the procedure was a histogram analysis. The histogram was used to define a function that in turn was used to compute the first, second, and third deviations. The zero crossings of the third derivative where the second derivative was non-zero were considered as thresholds.

The threshold between air and wood was set at the first threshold on the lower gray levels or left side of the histogram. The threshold between knots and good wood was harder to find because there may not have been any knots. Sapwood and heartwood were also thresholded. After thresholding, the boundaries were very irregular. To get convex regions, the authors used an iterative approach where the 8 neighbors of the pixel were examined to determine whether to fill in a pixel, leave it alone or delete the pixel from the region. The number of iterations was unknown. The knot regions were checked again to see that the principal axis of inertia pointed toward the center. Circular knots could have large variations in principal axis so less importance was assigned to circular regions. Rot was characterized by lack of growth rings. So an edge detector was used to count number of edge elements. The maximum number of edge elements in one of four directions divided by the total number of edge elements is highly uniform except in rotten areas. So regions of low density and low uniformity of edges were marked as rot. The rot regions were turned into convex regions with the same method used for knots.

Funt and Bryant (1987) reported that their algorithm required about 3 minutes per scan. The Siemens scanner used in the study only required 1.5 seconds of acquisition time per scan. At common production rates, the scanning time and especially the interpretation time were not acceptable.

Birkeland and Holoyen (1987) provided another motivation for introducing internal scanning. They mentioned that pruning gave excellent results in the form of knot-free wood. However, forest owners claimed that they would not get their investment back as long as the price for sawlogs was not based on absence of knots. An internal defect scanning method that would enable buyers to judge the degree and time of pruning should in the long run influence forest practice and promote pruning.

Using the general strategy of cutting a log so that the most serious defects were concentrated in some boards, leaving other boards with just a few small defects would usually give an increase in lumber value of about 8%-9% if internal defects were located prior to cutting. For individual pieces of lumber, they reported that the difference in value increase could be around 30%.

In Norwegian spruce, the contours of knots in sapwood were lost with CAT because the density of knots and sapwood were similar. The knots of spruce had a somewhat higher density than the clear dry wood but as the wood in the knots was fairly dry and the moisture in the sapwood was somewhat higher, the X-ray system saw the two types of wood to be of about the same density.

Perceived density from the CAT was a combination of the dry-wood density and the density from the moisture content. The moisture content profile resembled a soup plate with high edges corresponding to sapwood areas and a bowl region corresponding to the heartwood area. Moisture content dominated the dry-wood density as regards to the perceived density from the CAT. A dry knot with high density had the same perceived density as “green” sapwood that had low density but higher moisture content. The image-processing algorithm could give correct borders for knots in heartwood but

not in sapwood. This substantiated the experience that was also reported earlier by Funt and Bryant.

Roder *et al.* (1989) and Wagner *et al.* (1989) reported on a new design for CT scanners that radically decreased the scan time. The first commercial CT scanner was introduced by EMI in 1973. There was a four-minute acquisition time followed by a four minute image reconstruction per slice. Most current scanners had an image acquisition time of one second, which appeared to be the limit of the current design due to the centrifugal forces on the X-ray tube. A new design had a single beam that was deflected around like the electron beam in a cathode ray tube and then a ring deflected the beam so that it traveled into the target in a perpendicular direction. Because there were no moving parts, this new design allowed scan acquisition times as fast as 50 msec. Reconstruction of a 512x512 image required 10 seconds and a 256x256 image required 5 seconds. The image had 2000 gray levels.

Because the machine was not open on both ends, it would have been impossible for logs of arbitrary length to be run through the scanner on axis. Since the upper part of the scanner was open, logs could be tilted and scanned. The resulting cross-sections would be oval. At about 34 scans per second and an 8mm (0.31 inch) thickness, the throughput of logs would be about 50 feet per minute, which was half the typical lumber mill production rate. Changing the distance between slices to 16mm (0.62 inches) would have attained the typical production rate. The image reconstruction still took more time.

Like other researchers earlier, Grundberg and Gronlund (1991) noted that CT scanners had trouble distinguishing between sound knots and sapwood. The authors concluded that a 5 x 5mm-pixel size was sufficient. The authors reduced the search for

knots by analyzing about 10 concentric surfaces around the pith. The path of the knot was calculated in the heartwood area and the path was projected into the sapwood area where the sound knot was hard to distinguish. This had a disadvantage in that the path of the knot was not necessarily straight and the knot may have stopped somewhere in the sapwood or may not have been in the sapwood at all. However the data reduction was large. In addition, they noted that after classification, all pixel values could be stored in a 2 or 3 bit word.

Lindgren (1991a) reported how the CT scanner perceives density when scanning logs. The density at moisture content level MC was, ignoring air space:

$$Q_{MC} = \frac{\text{Volume of Wood}}{\text{Total Volume}} Q_{wood} + \frac{\text{Volume of water}}{\text{Total Volume}} Q_{water}$$

where $Q_{wood} = 1.5 \text{ g/cm}$, $Q_{water} = 1 \text{ g/cm}$

The absorption coefficient at moisture level MC was:

$$\mu_{MC} = \frac{\text{Volume of Wood}}{\text{Total Volume}} \mu_{wood} + \frac{\text{Volume of water}}{\text{Total Volume}} \mu_{water}$$

where $\mu_{wood} = 0.2646$ and $\mu_{water} = 0.1856$

There was a linear relationship between density and X-ray absorption.

CT-number = $1000 [\mu_x - \mu_{water}] / \mu_{water}$, where $\mu_{water} = 0.1856$. Since ratio of the absorption coefficients between wood and water had the approximate ratio as the density coefficients, the X-ray CT numbers gave a good measure of the total density. However, since the total contribution to X-ray absorption was made from wood and water, then volumes with high wood density and low moisture contents such as knots could have the same X-ray absorption coefficient as a moderate density volume with high moisture

content such as sapwood. In a later report, Lindgren (1991b) found that the standard deviation of the noise in CAT-scan images was given by $\sigma = k\sqrt{\frac{e^{\mu d}}{w^3 h D}}$, where σ = standard deviation of CT number, k = constant for the machine, μ = total absorption coefficient of test material, d = diameter of subject cross section, w = pixel width, h = slice thickness, D = X-ray dose.

Zhu *et al.* (1991) described a computer vision system to identify red oak defects from CT scans. The system had a low-level module for image segmentation and 3-D volume growing and a high-level module for defect classification.

The authors had a goal of removing the growth rings to make the segmentation process easier. One of the major characteristics of the annual rings was their high frequency property. An adaptive filter that was not described well was used to eliminate the annual ring structure while preserving the defects. The filtered image was thresholded on an image by image basis using a multithresholding method. The histogram was computed and the histogram itself was smoothed with a Gaussian function. Three thresholds were computed based upon the first and second derivatives of the smoothed histogram function.

After the filtered image was thresholded, tiny regions were eliminated by morphological erosion followed by dilation (opening). This also smoothed the remaining defect border regions. Next, regions were connected between slices to form connected volumes. Small volumes were eliminated. Five basic features of each volume were used to classify them into bark, knots, and clear wood. The five basic features were mean gray

level, variance of gray levels, minimum distance to center of image, a binary indicator for whether the region bordered the “air” region, and volume of region.

There were various rules based on these features to classify the regions. For example, the mean gray level value of knots was lower than clear wood. The variances of bark and knots differed. The minimum distance of bark to the center was high, while clear wood was close to the center. Knots usually did not share a border with the air. Clear wood had the greatest volume of all regions. An expert system using these types of production rules classified the regions.

2.3.2 Optical scanning of boards

Most scanning research work on boards uses some form of image sweep-and-mark algorithm. Because image processing is data intensive, automatic defect detection algorithms can spend a lot of time on defect-free areas. If the defect-free portions of the image can be excluded at an early stage in the process, the computational demands can be reduced significantly. This is the goal of image sweep-and-mark (ISM) algorithms. An ISM algorithm divides the image into a number of disjoint rectangles, called tiles. After sweeping through the tiles to analyze them (for example by quickly computing a tile’s average and/or variance), the algorithm marks those tiles which may contain defects. The marked tiles are then subjected to closer scrutiny and unmarked tiles are never examined again.

McMillin *et al.* (1984) described a proposed system to produce hardwood lumber. Board-surface defect detection was accomplished by sweep and mark detection using only gray level statistics of mean, variance, skewness and kurtosis. For marked tiles, the

tiles were further classified as clear wood or a certain type of defect from predefined rules that used both gray level statistics and texture properties.

In another study, researchers found that even motivated employees could find about 68% of the defects on rough boards. The ability to resist boredom and maintain an alert mental attitude was hampered by the rough mill work environment. The significance of the study was that a computer vision system did not need to be perfect to improve on current practice. The economic potential was considerable for such equipment even if only a small yield improvement could be obtained. The range of the defects found varied among operators from 59 to 74% with an overall average of 68% (Huber *et al.* 1985).

The three basic types of ISM algorithms are statistical (based on tonal properties), gray-level morphology (for classifying textures) and color-clusters (an example being a brown cluster would be classed as a knot). Funck *et al.* (1987) noted that proper illumination was critical for ISM algorithms. They tested various types of sweep and mark algorithms and concluded that statistical ISM performed best. They indicated that ISM combined with Hough transform accurately found knots on boards.

Forrer *et al.* (1988) found that gray scale combined with color allowed detection accuracy that matched human workers. The most important measures were $(R+G+B)/3$ (intensity) which was the same as gray scale and $(R-B)/2$ (color). In a follow-up report, the same group found that for loose and tight knots, open holes, and pitch pockets/streaks, statistical ISM performed best, followed by morphological ISM, then color clustering ISM. Statistical ISM achieved a 99% defect detection accuracy and an 85.5% clear wood accuracy. If defects only occupied 10% of a board, then the image

data reduction was 77% with 99% defect accuracy. The 99% accuracy figure excluded pitch pockets and streaks where statistical ISM did not perform well (Forrer *et al.* 1989).

Butler *et al.* (1989) noted that conventional ISM algorithms did not use neighborhood relationships in classifying a tile as marked or unmarked. Because defect tiles naturally tended to be clustered, they suggested that it would be advantageous to employ a more sophisticated marking rule that more readily marks tiles whose neighbors were already marked. The new statistical ISM, called NSISM, differed in that the marking rule used two threshold values instead of one. Initially, tiles were marked with respect to a primary threshold just like a conventional statistical ISM. Then, each of the four neighbors tiles (above, below, left, right) of each marked tile was compared to a secondary threshold. By making the secondary threshold more stringent than the primary threshold, neighbors of marked tiles were more readily marked. It was suggested that the stringent threshold would result in better performance on tiles that are on the boundary of a defect. Also, it was hypothesized that a secondary marking rule would allow the primary threshold to be made less stringent, thus eliminating some “false positive” tiles (i.e., clear wood tiles marked as defects).

Defect detection accuracy was more important than clear wood accuracy since defects not detected by sweep & mark algorithms would not be analyzed further. Clear wood accuracy prevents unnecessary processing, but improvements in clear wood accuracy should not come at the expense of defect detection accuracy. The new algorithm reduced the overall error rate in clear wood by 17.3% and reduced the error rate in defect detection by 66.6% compared to the conventional statistical ISM. Thus both types of detection were improved.

Although NSISM was based upon statistical features of the tile, the basic idea of a primary and a secondary-marking rule could apply equally well to ISM algorithms based on other features, such as morphology or color clustering.

Connors *et al.* (1989) reported on a computer vision system for locating and identifying defects. They had the goal of a robust system that could handle a variety of species. They also wanted a system that was fast so as to be industrially useful and yet have high resolution to catch small defects. Problems of rough lumber included variations of appearance & color due to variations of surface moisture, ultra-violet radiation (sunlight) exposure, weather exposure, sap, and roughness itself. In this paper, three species of rough lumber were scanned with a digital color camera. The segmentation system had the purpose of classifying pixels as background, clear wood, or defects. The segmentation system worked on various hardwood species on surfaced lumber. The recognition system was designed to assign labels to the defects. The system easily recognized the background color because it could be controlled. The clear wood region was identified simply as the region with the most pixels. The classification of pixels into defect regions was generous since isolated pixels could be eliminated later. Next small regions were merged and rules were then applied to classify the regions into wane, checks, and knots based on location and shape.

Koivo and Kim (1989) used statistical classification methods to classify 8 types of defects and clear wood on surfaces of boards. The input used was the gray scale digitized image from a camera. 64x64 windows of training data were taken from areas that were previously manually classified. Five features in all were computed for each sample. Two of the features were the mean and variance of the gray levels. The three remaining

features were from texture parameters that predict the gray level of a pixel given the west, north, and northwest neighbors of the pixel, or $y(i,j) = \theta_1 y(i-1,j) + \theta_2 y(i,j-1) + \theta_3 y(i-1,j-1) + \sigma w(i,j)$.

The estimated parameters could come from conventional least-squares estimation or by a robust estimation procedure. The feature vector $[\mu, \theta_1, \theta_2, \theta_3, \sigma]$ was calculated for each sample. The classification of the vector was accomplished by constructing a binary decision tree. The decision tree was constructed using a recursive, partitioning algorithm described in another paper. The result was a 97.2% correct classification percentage on an additional set of test data. The study used 20 sets of each class for training and 20 sets for testing.

Butler *et al.* (1993) noted in demonstrating the need for automated vision systems in classifying wood surfaces that 19% of veneer graded as DC by manual veneer graders should have been in the higher AB grade. Such mistakes lowered the value of 1/10" Douglas-fir veneer from \$112 to \$54 per 1000 sq.ft. They reported on an improvement to the NSISM algorithm described earlier. ASISM (adapted statistical sweep & mark) was similar to NSISM, but ASISM made four passes (called primary, mark-contraction, mark-expansion, and fill-in) through the tiles. The primary pass was the same as the primary pass in NSISM. In the mark contraction pass, isolated tiles and tile pairs were subjected to a very stringent (1/10 of primary threshold) threshold. The mark expansion pass was similar to NSISM except more candidates were considered, namely, those unmarked tiles that were diagonally adjacent to two or more marked tiles. The filling-in pass attempted to fill in concavities in a connected group of tiles. Tiles that were

vertically and horizontally in between two already-marked tiles were subject to being marked by the secondary threshold.

There was not a practical difference in defect accuracy between ASISM and NSISM, but ASISM achieved a 40.31 percent reduction in clear wood error rate. ASISM used more rules but the execution time was similar to NSISM because most of the time was in the sweep phase, which was identical. While further refinements might have improved the accuracy, the researchers concluded that any additional complexity would have precluded it from being considered as a preprocessing algorithm.

2.3.3 Ultrasonic, microwave, and stress wave scanning of logs and boards

Martin *et al.* (1987) reported that the propagation times of microwaves through wood were influenced by wood density, moisture content, and slope of grain. To a lesser extent, the presence of large knots influenced the propagation time. However, the effect of moisture content could overshadow the presence of knots, especially small knots. Metallic objects were well detected by microwaves. Other defects, such as splits, resin pitches, wormholes, sapwood, and wane, were not detected. They conclude that because of the large variations found in wood, the complete detection of defects would require more sensors in addition to the microwaves.

Birkeland and Han (1991) reported that ultrasonics has been used for material testing in metal parts for a long time. The principle was simple: a train of ultrasonic waves with known characteristics was sent through the material and received either at the other end or as a reflected signal. The changes in the signal were correlated to physical properties of the material. The advantage of ultrasonics was that the equipment was very cheap in relation to the other proposed scanning methods. Thus, one could

operate several ultrasonic systems in parallel to achieve high production rates. The disadvantages of ultrasonics for logs were that the testing must be done under water. Bubbles gave false readings. They found that ultrasonics had less resolution than X-ray tomography.

Steele *et al.* (1991) used a slope-of-grain indicator to locate defects or hardwood lumber. This indicator measured the dielectric constant of wood, which is affected by the slope of the wood. Knots, bark, holes, and splits could be detected. Rot could not be detected because it did not disturb the grain. The actual detection method used a manual method of noticing an increase in the digitized data. One disadvantage seemed to be that the exact borders were not given but the method gave a general location. Also, this device could only be used on boards.

Another related method involved the use of a stress wave nondestructive evaluation technique that utilized simple time-of-transmission measurements to measure the speed of sound. This technique used a mechanical impact to impart a wave in the board. Two sensors were placed on the board and the time for the wave to travel between the boards was measured and used to calculate wave speed. Ross *et al.* (1994) found that this measurement could distinguish between heartwood and sapwood or between heartwood and wetwood, but sapwood and wetwood were indistinguishable. The goal was to find a nondestructive test for wetwood detection. The accuracy of detecting wetwood averaged about 84% in red oak. Unfortunately, accuracy of detecting wetwood in white oak was about 45%.

Another research group studied the use of ultrasonics to detect checks and honeycomb in dried lumber (Fuller *et al.* 1995). They noted that lumber drying was a

critical step in manufacturing hardwood lumber because degradation in the form of surface checks and honeycomb was especially severe in oak lumber and a major source of value loss and waste. These types of drying defects were caused by internal stresses from different shrinkage within pieces of lumber. They noted that X-ray techniques were useful but ultrasonic techniques were also useful and much cheaper. Ultrasonics could be used because the presence of honeycomb and surface checks in dried lumber significantly increased sound transmission time.

Samples of red oak lumber were passed between two 84-KHz rolling transducers that were positioned on the narrow sides of the lumber. A profile of the transmission time versus the length position of the board was recorded. There was a strong correlation between time and presence of defects. 98% of sections with transmission time above 300 $\mu\text{sec/ft.}$ contained defects. 96.5% of sections with transmission time less than 250 $\mu\text{sec/ft.}$ contained clear wood. Oddly, knots only created a small increase in transmission time that was less than checks and honeycomb.

In another application of ultrasonics, Kimmel and Janowiak (1995) showed how ultrasonic testing could be used to rate veneer and sort it into two grades to improve physical properties of laminated veneer lumber such as modulus of elasticity, modulus of rupture, and shear strengths and flatwise flexure properties.

2.3.4 Multisensor scanning of boards

Although others (Martin et al. 1987) have reported a need for multisensor scanning to increase the reliability of defect classification, the results had not been overwhelmingly successful and there were additional problems such as registration and

heavy data burdens associated with a multisensor approach. Hagman and Grundberg (1993) used multiple sensors: microwave attenuation, microwave dephasing, color camera for red, blue, and yellow, and X-ray scan. The six signals were all scaled to the 0-255 range and the resulting 6 images were registered. The six sets of data inputs were regressed to the indicator outputs for knot, hidden knot, pitch, decay, log blue, sound knot, and clear wood. They reported that the results were good, and the magnitude of the regression coefficients provided an indication about which inputs were important. They reported that the results for compression wood, decay, and checks were not stable. Because of the multilinear approach, causality was not assignable.

Kline *et al.* (1993) described a prototype system that sought to detect features such as visual surface features, board geometry, and internal features. Consequently, a multiple sensor approach was required. The system was designed for hardwood lumber of various species and sizes. The system used a color camera, an X-ray scanner similar to airport luggage scanners and a laser scanner to measure widths. The bottleneck was in the image processing time. The computer used for image processing was an IBM RS/6000 520 series workstation and an IBM PS/2 was used for overall system control. The system at the time of the report had not been completed.

2.4 Log Processing from Defect Information

2.4.1 Automatic lumber grading

McMillin (1984) reported briefly on an interesting application of image processing to plywood shear strength inspection. Plywood was frequently tested for shear strength and after shear failure, the sample was examined to determine the proportion of the shear area created by wood shear versus shear across the glue area. An image of the shear area

could be easily thresholded to separate the wood and glue areas. Then, the area proportions were easily calculated by computer.

Conners *et al.* (1987) provided a preliminary report on a hardwood lumber grading program. The program assumed that the defect areas were previously marked. The grading program boxed the defect areas, calculated clear wood area and applied a preliminary grade based on size, faces, and clear wood area. Then, the true amount of clear cuttings was determined to verify the preliminary grade. If it failed, the next lower grade was tested and so on.

Klinkhachorn *et al.* (1988) provided the final report on the hardwood lumber grading program. This program was an improvement over previous versions in that it incorporated both standard grade rules and exceptions allowed for many species. The program also considered both sides of the board simultaneously.

The program received input about defects and locations from a yet-to-be designed computer vision system. The program boxed in defects. Then the program found all rectangular clear areas. Overlapping clear rectangles were resolved. Defects that were allowed on some grades but not others were placed on a third “pseudo” face. Based on the overall length, width and clear areas, preliminary grades were assigned. Closer examination performed a final check on the grade. If the board failed to meet the grade requirements, the next lower grade was checked until a grade could be assigned. The program was written in FORTRAN 77 and stored all rules in the data block.

2.4.2 Sawing optimization for boards

Klinkhachorn *et al.* (1989a), described a program to optimize the placement of cuttings on lumber based on a description of each board in terms of shape and defect

location and a cutting bill. A heuristic approach was used to shorten computation time. No crosscutting or ripcuts were assumed. An assumption was that the parts could be cut by laser or by router. There were 16 heuristic algorithms that were tested via simulation with four different cutting bills. The best algorithm was chosen from the simulation.

The algorithm searched the board for clear rectangular areas. The leftmost area was chosen first. Then, that area was suboptimized with only one kind of part from the cutting bill. The area occupied was then marked as a defect and the process started over again and was repeated until no clear area existed that was larger than the smallest part. The heuristic worked better than current tables used by industry. However, aside from the laser cutting assumption, there were other drawbacks. These included: no consideration was given to how the weights for each part on the cutting bill should be assigned, no consideration was given on how to minimize cutting time, and the system could not handle non-rectangular parts.

Klinkhachorn *et al.* (1989b) described a program for training hardwood lumber graders. They claimed that millions of dollars were lost each year due to inaccurate grading. Many years of experience were required to grade at industrial rates of production. The program allowed inexpensive practice.

Schwehm *et al.* (1990) described a program that determines whether edging or trimming will increase the value of a board. The program balanced the current prices of the different grades against the loss of board area due to trimming. It also considered the cost of trimming in its calculations. The program allowed current prices to be used and was designed to work with a hardwood-grading program.

Brunner *et al.* (1990) presented a modified dynamic programming algorithm for solving two-dimensional knapsack problems, first reported by Gilmore and Gomory. They applied this algorithm to the cutting plan for a rectangular clear region. The algorithm minimized the wasted area. Given a cutting bill as input, the algorithm considered every possible clear region size up to some specified maximum. The optimal cutting plan for each size was predetermined for later use. Computation time was reduced by not considering every possible clear region size since the longer a clear region was the more likely the clear region width would be smaller.

The table of optimal solutions for clear regions was used later in an unspecified heuristic that resolved the limited number of overlapping clear regions that were found on a given board. The results of these comparisons were then used to determine the algorithms intermediate path to a final solution. This approach did not guarantee optimal results, but it was better than manual methods and could be executed on microcomputers for real-time process control.

Carnieri *et al.* (1993) described a heuristic procedure to solve the optimal cutting of lumber that has only one defect. The procedure inscribed the defect within a box and optimized the cutting given the location of this box. The procedure used a branch and bound approach.

Carnieri *et al.* (1994) developed another approach to optimize the cutting of parts from lumber. The problem was modeled as a knapsack problem and was solved using dynamic programming. The optimal cutting of lumber is a dual problem of determining the right mix of lumber from the different sizes and grades available and how to cut the

particular lumber pieces to satisfy the customer bill and minimize waste and raw material cost. The generic lumber cutting problem was formulated as:

$$\text{Min } TC = \sum_k \sum_j c_k x_j^k \quad (1) \text{ minimizes cost of lumber input}$$

$$S.T. \quad \sum_k \sum_j a_{ij}^k x_j^k \geq d_i, i = 1, \dots, m \quad (2) \text{ constraint to satisfy demand}$$

$$\sum_j x_j^k \leq N_k, k = 1, \dots, K \quad (3) \text{ constraint to satisfy stock availability}$$

$$x_j^k \geq 0$$

where c_k = cost of lumber type k

x_j^k = the number of lumber pieces of type k cut following cutting pattern j

a_{ij}^k = the number of dimension parts i obtained from one lumber type k following cutting pattern j

d_i = amount ordered for dimension part i

N_k = amount of lumber type k available in stock

The 2-dimensional knapsack algorithm could be used to determine the optimal cutting pattern of one piece of lumber into smaller dimension parts. The cutting of each piece of lumber should not be done independently but by comprehensively considering the cutting bill & the availability of lumber pieces. This paper solved the problem in two stages: one stage optimized the allocation of lumber based upon the second stage's optimal cutting patterns. The interface between the two stages was made by shadow prices (simplex multipliers) associated with the demand constraints. Rip cut-first approach and crosscut-first approach were both solved, then compared to choose the best approach.

Åstrand and Rönqvist (1994) reported that current automatic scanning systems for wood surfaces gave accurate location data comparable to manual systems. They presented a new optimization procedure that gave 7% more value than an existing commercial optimizing system. This system required an automatic defect detection system as a prerequisite. All sides of parts were divided into predefined quality regions. A board was divided into 1mm lengths. Each 1mm segment was represented by a bit. If the bit was on, then a cutting was allowed. In a second dimension, each collection of bits represented a part in one of its orientations. A defect was also represented by a bit list. But in this case, the bit list was shown where the parts could not be cut with respect to the defect dimension. The XOR operation of two binary inputs is defined as outputting a 1 when only one and not both of the inputs are equal to 1. Thus, an XOR operation with the defect bit list and the board bit list left a final bit list showing where all cuts could be made to produce a useful part.

An integer linear program was used to maximize the value of the possible cutting combinations. This was an example of the set packing problem. The formulation was:

$$\text{Max} \sum_i^m \sum_j^n c_i x_{ij} \quad \text{where } n = \text{length of board}, m = \text{number of possible cutting lengths}$$

$$c_i = \text{value of cutting } i, i = 1, \dots, m$$

$$S.T. \sum_i^m \sum_j^n a_{ijp} x_{ij} \leq 1, p = 1, \dots, n$$

$$x_{ij} \in \{0,1\} \quad i = 1, \dots, m \quad j = 1, \dots, n$$

$$a_{ijp} = \begin{cases} 1 & \text{if cutting } i \text{ is allocated at position } j \text{ also covers position } p \\ 0 & \text{otherwise} \end{cases}$$

a_{ijp} has a certain structure, namely,

$$a_{ijp} = \begin{cases} 1 & \text{if } j \leq p \leq j + li \quad i = 1, \dots, m \quad j = 1, \dots, n \quad p = 1, \dots, n \\ 0 & \text{otherwise} \end{cases}$$

The solution method used was a forward dynamic programming approach. The average value increase was 7% with a 90% confidence of the value increase being greater than 4.4%. Automatic wood inspection would do more than replace a manual operation. It could give higher output yield and use raw material of lower quality. There were no manual counterparts to 2-D optimization. There was a strong need for improvement of the detection of defects. The more the rest of the production chain was automated, the more important it was that the defect detection was reliable.

Recently, one author (Blackman 1995b) described a current rip saw optimizer that used color cameras to scan boards at 400 fpm (120m/min.). 10 Pentium computers were used to process the 10-board queue between the scanner and the rip saw. Each computer had about 40 seconds to find the optimal rip saw-cutting pattern for its board. Boards were allowed to be random width and length. Product and price information guided the optimization process.

2.4.3 Log sawing optimization

Most hardwood sawmills practice the grade sawing method to increase the value of lumber sawn from a log. The grade sawing method uses log position and rotation to move externally indicated log defects to a location that will have the least impact on lumber value. External indicators of internal defects are used by the sawyer to choose the starting face of the log sawing operation in an attempt to maximize sawn lumber value. After the log is opened, the internal defects exposed during the sawing process are used as additional information to determine the value-maximizing log turning sequence.

Wagner *et al.* (1989) described the grade sawing method as cutting from the best of 4 faces on a cant. Anytime, the current sawn face was judged to be worse than one of the 3 other faces, the cant was turned to expose the best face and sawing continued at this face only as long as it was the best face.

Some hardwood sawmills use the live sawing method, where logs are sawn with parallel sawlines. This method is often used with lower grade sawlogs. Wagner *et al.* (1989) noted that live sawing had been shown by several studies to yield more value and volume. Hardwood mills continue to use grade sawing because live sawing produces a high proportion of quarter-sawn lumber, which poses a marketing problem for some lumber manufacturers.

Tomographic grade sawing refers to the method of sawing that assumes the internal defects have already been located. This approach is not used by industry but rather by researchers to estimate the value of internal scanning. Tomographic grade sawing uses the same methods of turning the cant between sawing cuts in order to maximize the value of the log as the sawyer's grade method. The only difference is that tomographic grade sawing uses the internal defect information provided by internal scanning rather than relying on the exterior surfaces as sawyers must do now.

In a similar manner, tomographic live sawing refers to the method of live sawing that uses internal defect information to position and rotate the log for optimal value recovery before live sawing begins.

Occeña and Tanchoco (1988) described a graphic based sawing simulation system. This simulator used solid modeling concepts to perform its tasks. The methods used for solid modeling included constructive solid geometry which used basic geometric shapes

and Boolean operations to build more complex shapes, and boundary representation which used faces, vertices, and edges to build up polyhedral representations of objects. The log was represented by a polyhedral, which was an advance over the traditional cylindrical and conic models. The defects were represented as polyhedrals “subtracted” from the log. The sawing cuts were represented by an intersection of the log with a solid rectangular block. This intersection was repeated in order to break down the log into a sawing pattern. Solid modeling allowed the ease and speed of Boolean operations to simulate sawing.

Another study looked at a linear programming approach that was based on the probable yields from various size logs and log grades. The program minimized the total value of the log requirements given a list of output requirements that included volume requirements for various grades of lumber. The authors found that greater profit could be achieved by processing higher yield, larger diameter logs despite their higher purchase and processing costs (Sim *et al.* 1991). This approach did not actually optimize the cuts within a log but it attempted to determine what size and grade of logs to purchase to minimize raw material costs.

Harless *et al.* (1991) noted that various studies have indicated that the improvement in lumber value ranged from 9-12% between the best and worst orientation when logs were sawed with defect information. They noted the inadequacy of the previous studies and attempted to find the improvement of tomographic sawing over typical industrial methods. The researchers used a CAD/CAM wire frame representation of the log. Four different sawing methodologies were simulated: sawyer-grade sawing, tomographic grade sawing, live sawing, tomographic live sawing.

For live sawing, the tomographic live sawing method improvement was 10.8% over the average (not the worst) orientation. The value improvement was the result of better grade yield rather than better volume yield. The tomographic grade method yield was 5.0% above the average orientation for grade sawing. The tomographic live sawing yield over current industrial sawyer-grade methods was 7.05% and tomographic grade method improvement over current industrial sawyer-grade method was 1.46%. However, these results were from a single test-log.

In another study, Lee *et al.* (1991) scanned the surface of a log optically and located knots on the surface. A mathematical model was used to predict the interior shape of the knot. Based on these assumptions, the log was rotated 360°, one degree at a time to obtain the best value yield. They distinguished knot from bark by color and texture. Also, they ignored knots under 1" in diameter. They stated that among all the defects used in grading, knots occurred most frequently. Also, they noted that the presence of knots greater than 1" usually determined the grade of the board.

Using an extensive database of 2-axis scanned logs containing scans of 834 logs, Maness and Donald (1994) conducted a statistical study on the effect of log rotation on the value yield of dimension lumber produced from logs. The data was stratified to cover the lengths and diameters found in industry. The set of cross-sectional cant sawing profiles was picked to mimic the normal set used in industry to produce dimensional lumber. The 0°-rotation position was selected where the sweep of the log was vertical and one end pointed up. The simulation of the log sawing occurred at 45° rotation increments.

Statistical tests first confirmed that the 0° rotation, the position where the sweep end of the log was pointed up, the “horns up” position, was the best overall fixed position. They also confirmed that rotation to each log’s optimal position yielded statistically significant higher yield than the 0° position. Further research would look at benefits of smaller than 45° rotation increments.

In a more recent experiment, Steele *et al.* (1994) noted that previous studies that investigated the optimization of lumber value from knowledge of internal defects used limited sample sizes or simulated data. Another problem with some of the earlier studies was that they did not compare the optimal sawing pattern with the methods used by experienced sawyers. Also, the value increase for all 3 USDA Forest Service hardwood grades was not studied with the exception of one study that used only 2 logs per grade.

The authors used 24 red oak logs with 8 logs each in the three log grades. All lumber sawn was 4/4 (1.00 inch) with 0.25 saw kerf. The standard methods for live sawing and grade sawing were used to simulate the experienced sawyer. To determine the optimal live sawing orientation, each log was “cut” via simulation at 15-degree intervals for 24 rotational positions. For the optimal grade-sawing orientation, only 6 of the 24 rotational positions were tested as only these provided unique solutions.

A split-plot experimental design was developed for this analysis. The whole-plot factor used was log grade with the subplots consisting of each log. The subplot factor was the sawing method. Whole plots had a completely randomized design and subplots were randomized complete blocks. The treatment structure was a full factorial arrangement with the factors being log grade and sawing method.

The statistical model used was: $E(\text{Value}) = \beta_0 + \beta_1 G + \beta_2 L(G) + \beta_3 S + \beta_4 G * S$ where G = grade influence, L = individual log influence, S = sawing method influence. Comparisons-of-means testing revealed a statistically significant increase in tomographic versus sawyer methods. The best result was that the increase of value of tomographic methods appeared to be about 10% regardless of the sawing method and the log grade.

Guddanti and Chang (1998) have shown that they could replicate the actual performance of a sawmill on the computer using a custom software program called TOPSAW. TOPSAW was able to assemble CT images from a log and generate a board face from any cut that was specified by the software user. The program could also grade the board produced by the simulated sawing. After a 12-foot log was scanned, it was sent to a sawmill to be cut by an experienced sawyer who chose the grade sawing technique. The sawing sequence, log angle and sawblade positions were recorded. Then the TOPSAW program was used to make the same cuts in the log model. The board faces generated by TOPSAW were virtually identical to the actual board faces. The grading program in TOPSAW was used to grade the boards and the total value came within 97 percent of the actual value of the boards at the sawmill. This study represented the first time that software could be shown to replicate the sawmill results. Thus, the groundwork has been laid to use TOPSAW to generate the exact gains that could be realized from the use of internal defect information to guide the sawing process.

In the most recent and best study to date, Chang *et al.* (1997) were the first group to compare actual sawmill results with tomographic methods. All previous studies had

simulated the sawmill method of sawing and predicted the value of the boards that would have been produced. This study was the first to scan the logs before the logs were sent to a sawmill to be cut. Thus, the actual value produced at the sawmill using current methods was known exactly. The authors used TOPSAW to generate the optimal live-sawing pattern. Guddanti and Chang had previously shown that TOPSAW was capable of replicating the sawmill results. Thus, a very accurate measure of the possible optimal value was known. Based upon the 7 log sample (3 logs were Grade 1 and 4 logs were Grade 2), the live-sawing optimal patterns produced 15.08% more value than the patterns actually used at the sawmill. Since the sawmill used live-sawing methods and grade-sawing methods, the additional consideration of optimal grade-sawing patterns was estimated to produce 19.43% to 29.71% more value than the patterns actually used at the sawmill. Thus, the potential for tomographic methods may be much higher than previously thought.

2.4.4 Direct log to product processing

In order to obtain better yield from hardwood logs, the direct processing of logs to produce final parts has been proposed. This method bypasses the step where lumber is produced and then parts are produced from the lumber.

Lin *et al.* (1995) conducted a simulation study to evaluate mill designs whose purpose was to cut green dimension parts directly from logs. The impact of changes in mill configuration, log grade input, and cutting bill on the production rate was investigated. Four different mill designs were studied. The potential high production rates indicated that direct log-to-dimension manufacturing might be feasible. A previous

study found a large potential for increasing dimension yield from logs by converting hardwood logs directly into dimension parts.

Blackman (1995a) reported on a French company that has introduced direct log to product processing. Groupe Menuiseries Gregoire manufactured finished furniture, doors, and windows. The company handled all operations including breaking down logs, turning lumber into molding and other components and then manufacturing the final products. There was one exception: Asian hardwoods were purchased as cants and resawn at the plant. They accounted for about 40% of the raw material.

2.5 Relevant Image Processing Methods

2.5.1 Image thresholding

Kittler *et al.* (1985) presented a review of current thresholding methods and proposed a new method that does not require a histogram. Among their various proposals were an iterative method where one arbitrarily computed a threshold, found the averages of the two classes, then let the next threshold be the average of the two class averages. The process would continue until a stable solution was found. Another proposal suggested picking the threshold whose class edges most closely approximated the edges from an edge detector. Both of these proposals inherently required many data passes.

They noted that the method of minimizing the mean square error between the gray level picture and its binary representation is computationally simple, stable and effective. The only drawback seemed to be the a priori decision on the number of thresholds. Entropy minimization had been suggested and it had similar results to mean square minimization.

Another proposal involved picking the threshold where the gray level at the borders matched most closely to the average gray level in a $1 \times M$ window that lied orthogonal to the border contour. This method was repeated until a stable solution was reached. Another proposal suggested a threshold selection that maximized intensity contrast between border pixels. They noted that these thresholds based on simple statistics required restrictive assumptions such as constant levels in regions, uniform illumination, and distinct gray-level populations.

Sahoo *et al.* (1988) provided a survey of the current thresholding techniques. Thresholding, they noted, could be classified into global and local techniques. The global techniques were further classified into point-dependent and region-dependent.

Point-dependent techniques included the P-tile method, the basic bimodal thresholding, the minimum variance (or minimum mean square) method that was already discussed, the maximum entropy method, and the moment-preserving method. Not all the methods were automatic in nature. The authors evaluated the automatic global methods on three pictures that were not binary by nature. They used a uniformity measure and a shape measure to rank the methods. The two best overall methods were the minimum variance method and the moment-preserving method.

To improve on the maximum entropy method, Abutaleb (1989) proposed an entropy method that considered spatial relationships as well as just gray level. He showed that choosing the threshold as the value that maximizes the entropy of the 1-dimensional histogram of an image could separate the background from the foreground of an image. This approach did not take into consideration the spatial correlation between pixels in an image. He propositioned that the performance might degrade

rapidly as the spatial interaction between pixels became more dominant than the gray-level values. In the proposed approach, a two dimensional histogram was created with the pixels gray level on one axis and the average gray level of the pixel's neighborhood on the other axis.

Presumably, the two-dimensional histogram had two peaks corresponding to the background and foreground. Assuming that the background had low pixel gray levels and low neighborhood averages and the foreground had high pixel gray levels and high neighborhood averages, then a threshold pair (s, t) where s is the gray level threshold and t is the neighborhood average threshold would divide the 2-dimensional histogram into four sectors. Two sectors would contain the peaks and other two sectors would contain distant diagonal components that were close to 0 in value and could be ignored. The entropy in the peak sectors was calculated by

$$H(A) = -\sum_{i=1}^s \sum_{j=1}^t \left(\frac{p_{ij}}{P_{st}} \right) \ln \left(\frac{p_{ij}}{P_{st}} \right), \text{ where } P_{st} = -\sum_{i=1}^s \sum_{j=1}^t p_{ij}$$

The off-diagonal sectors were ignored to reduce computation time. The entropy based function $\Psi(s, t) = H(A) + H(B)$ was maximized by an exhaustive search of s and t . When compared to 1-dimensional entropy, 2-D entropy took more computational time (an order of magnitude).

1-D and 2-D entropy thresholding methods gave similar results when the signal-to-noise ratio is high (above 12dB). The signal-to-noise ratio has a number of definitions in the literature but the traditional definition and the one used in this paper is given by:

$$SNR = \log_{10} \frac{\text{noise free power}}{\text{noise power}}$$

Power was the summation over all pixels of the square of the gray levels. For low *SNR*, the 2-D entropy thresholding was superior. Some dilation or other postprocessing would also improve the output.

Lee *et al.* (1990) noted that the previous survey by Sahoo *et al.* (1988) evaluated the performance of several global thresholding techniques on a set of test images that did not satisfy a two-class assumption. Therefore they proposed that a new study was required for images where it was known a priori that the image contained two principal brightness values, object and background. The purpose of this paper was to evaluate the performance of five global thresholding techniques based on a set of two-class test images. The measures of performance were the probability of error, a shape measure and a uniformity measure both described by Sahoo *et al.* (1988). The parameters that were varied in the study on the test images included the object (foreground) size and the mean difference in gray level between the object and the background. The algorithms tested include the simple image statistic method by Kittler and Hollingsworth, the minimum variance method, the maximum entropy method, the moment-preserving method and the quad-tree method. The two test images included a high contrast image and an image with gradual shading.

The probability for error was defined as $Pr(err) = Pr(O) \times Pr(B/O) + Pr(B) \times Pr(O/B)$ where $Pr(B/O)$ was the probability of classifying an object pixel as a background pixel. The values for $Pr(O)$ and $Pr(B)$ had to be obtained from previously manually-segmented images.

The performances of the various thresholding methods were image dependent. No single algorithm performed best across all test images and performance criteria. Of the

five algorithms tested, the minimum variance and simple image statistic methods performed relatively well. The performance of the entropy and quad-tree methods were sensitive to image characteristics such as contrast and histogram distributions. Also, the entropy method was more computationally complex. They concluded that the minimum variance and the simple image statistic method were the best choices for 2-class machine vision thresholding applications.

As noted by previous researchers, Hannah *et al.* (1995) mentioned that one drawback to thresholding was that there was no single method that could be universally applied. The researchers examined single and dual threshold methods based on variance and entropy measures of the image histogram.

Automatic thresholding based on the minimum squared error and based on the coefficient of correlation between the original and thresholded image yielded the same result. These methods were inefficient for multiple thresholds since they required exhaustive searches. Also, very small objects (population ratios) in the histogram could be overlooked. A second group of methods used entropy. The gray level with the maximum entropy of the two separated distributions was chosen. For single thresholds, studies have shown that the minimized variance method worked better. The minimum variance method and the entropy method could be used for single or dual thresholds.

The researchers extended the entropy method to automatically determine whether to use single or dual thresholds based upon the data. Basically the entropy was calculated for every possible gray level threshold. Current methods chose the highest point to be the single threshold. The new method simply looked for minor peaks and chose the second most significant peak if it existed. So the method did not force two

thresholds if a minor peak did not exist. The new method also was very sensitive to small foreign objects. The sensitivity could be adjusted by adjusting the criteria for finding a minor peak.

They concluded that the minimum variance method was best for large foregrounds, the extended entropy method worked best for thresholds for small objects.

2.5.2 Image segmentation

Kass and Witkin (1987) presented a survey of flow techniques used in image processing. These methods involved finding the dominate-flow direction in local areas and creating flow orientation images. Some of the uses of flow orientation images were using the flow coordinates to provide preferred directions for edge detection and using the dominant flow to aid in the detection of anomalies that stand out from the flow. A knot located in a grain pattern was characterized by having low flow coherence in a background of very high flow coherence (the annual rings or grain). Also, direction derivatives along the dominant flow direction would show anomalous elements such as checks that were perpendicular to the flow while diminishing the edges in the flow direction such as grain.

Kennedy *et al.* (1989) presented a method of segmenting medical images and calculating region volume in pseudo 3-D scans using Sobel edge detectors. They also used the edges of a previous MRI slice to begin the segment border search on the current slice.

Raya (1990) described an expert system approach to image segmentation. The system used a pair of MRI images for each slice. One image highlighted proton density and the other image highlighted T_2 -weighted intensity. These two images helped

differentiate the various types of tissue. Histograms for various types of tissue had already been established. The pixels were given membership values to each class of tissue according to the histogram height for a given tissue. If the gray level of a pixel did not appear in a certain histogram, there was a 0 value given. If the gray level of a pixel coincided with the mode of a histogram, then the pixel was assigned a 1 membership value for the tissue type represented by the histogram.

A number of statistics for each pixel was gathered from both images of the same slice. Then rules were applied on the statistics and membership functions to classify the pixels. The rule-based system was very problem specific. For a different imaging protocol, one would have to formulate a different set of rules, and a new set of statistical properties. The process was less efficient in terms of computation time, but production-rule systems were more flexible.

Thomas *et al.* (1991) described an algorithm that separated an object from the background with mathematical morphology. The algorithm first performed a gray scale morphological opening with a flat round structuring element larger than the smallest dimension of the foreground object. Subtracting the result from the original image gave a difference image. This difference image had its contrast stretched to cover the entire gray scale length. This also served to normalize all the difference images. A constant threshold that was previously determined by experimentation was applied to the stretched difference image. From the thresholded image, the largest region was assumed to be the desired foreground object. A binary morphological closing was performed to remove small holes and crevices in the region. Next, a binary morphological opening was performed to remove thin protrusions from the foreground object. Next, the

foreground object was skeletized using a hit-or-miss morphological operator. The hit-or-miss operator produced an image of edge pixels, which is subtracted from the foreground object. The process was repeated until a one pixel-wide skeleton was formed. To measure the length of the object, the distance between the endpoints of the skeleton was calculated and converted into real measurement units. The complete algorithm required approximately 10 minutes.

Toulson and Boyce (1992) used neural nets to segment MRI scans of brains. The data had to be converted to another form for input to the feedforward neural network. The neural network was trained using manually segmented images.

Deklerck *et al.* (1993) mentioned that segmentation and labeling remained the weakest step in many medical vision applications. Generally, oversegmentation was preferable to under-segmentation because it was easier to attribute the same label to the split segments than to split a segment and label its parts differently. This paper introduced a modified split & merge algorithm. The splitting step was performed according to edge information. Merging occurred on the basis of similarities in gray level statistics of the segments. The edge detection was based on first and second derivatives of the image gray levels. First, Gaussian smoothing was applied, then edge detection, then root selection. Root selection was finding the geometric centers of spaces between edges. After root selection, came region growing. The region growing method used was called “steepest path growing.” It used the derivatives that were calculated before. Each segment’s gray level statistics were calculated, then region merging was done based on the statistics.

They also introduced a method for body cavity detection that used a rough segmentation between the objects and background based on automatic thresholding by minimum error method. (i.e., choose the threshold to minimize the mean square error between the input gray level and the average gray level of regions). An adaptable salt and pepper filter was applied to the binary image. It was adaptable in the sense that the threshold for the noise area was higher for elongated segments and segments in the center of the image. Region growing was by steepest path. The last step was border smoothing.

Manos *et al.* (1993) mentioned the two main approaches to image segmentation: edge and region based. They noted that edge-based methods usually require a well-defined model of the object boundaries to produce successful results. Region-based segmentation methods have produced promising results for scenes that exhibit uncertainty regarding their content and boundaries of objects in the image, as in natural scenes.

The researchers discussed the method they used to segment bones in a hand-wrist x-ray image. The approach the authors used combined region growing, region merging, edge detection and a list of scene labeling rules. The edge detection was accomplished using the Canny edge detector because it provided a single edge response. The edge strengths were cutoff below a threshold so only strong edges remained. Next, edge-preserving smoothing was applied. Then, conservative region growing using 4 connected pixels with a threshold of 3 was used. This produced 500-1300 regions in the images used in the study. Region merging was applied based on a combined score of gray level mean similarity, size (small regions are preferred) and connectivity (ratio of common

border to overall borders). This merging reduced the regions by 40%. Regions were merged if their common boundary did not correspond with the edge detector output. Each region was classified as background or bone based on nine rules. Each rule was applied to the image in turn. The last step was application of two labeling error correction rules to eliminate small regions (holes). The process was robust and reliable but the trade-off was a time-consuming procedure with a very narrow application.

Vincent (1993) described morphological reconstruction that was termed in other papers as conditional dilation. Binary reconstruction was described as the union of connected components of a binary image which were “marked” or seeded by another binary image. The author extended morphological reconstruction into the grayscale case. He provided two standard algorithms and a newer algorithm which was an order of magnitude faster than any previously known algorithm. Binary reconstruction was described as a series of conditional (or geodesic) dilations.

Deruyver *et al.* (1994) examined the use of segmentation for separating white and gray matter in MRI images. They found that thresholding can only separate the background from the foreground. They found the noise too complex to remove even with frequency filters. The algorithm consisted of local histogram equalization to enhance contrast with white and gray brain matter. The original image was automatically thresholded using the local histogram to separate background from head. This image was then intersected with the enhanced image to see only the head area. The enhanced image was thresholded into 3 groups: non-gray, undefined, and gray matter. Isolated non-gray pixels were removed. Next, morphological operators were applied. Dilation over the non-gray and undefined pixels was performed to remove gray “bridges” created by

noise. Next, a morphological opening (erosion followed by dilation) over undefined pixels and gray matter was performed to remove thin protrusions of gray matter. Finally, small region suppression was used to reduce the number of regions. They claimed that the results were as good as manual segmentation.

Philip *et al.* (1994) presented an algorithm that used the generalized Hough transform along with fuzzy set theory to define a subregion of interest around a possible target, which provided an initial estimate of the location. The fuzzy set theory allowed the Hough transform to work with an approximate shape of the target objects. In their work with tomography, they allowed each previous slice's results to become the approximate model for the subsequent slice.

2.5.3 Image registration

Knoll and Delp (1986) noted that a difference image was formed by subtracting one image from another. One practical problem of imperfect registration was that, if images were not in perfect spatial registration before subtraction, the difference image would contain incomplete cancellations of unchanged background objects. Usually this noise was reduced by smoothing but this also blurred the true differences. Another method used is thresholding but this was effective only for subpixel registration error.

A useful property of registration noise is that it has a zero mean. Let $g(i,j)$ be the difference image between a discrete image and a translated version of itself. Then $g(i,j) = f(i,j) - f(i-\Delta i, j-\Delta j)$. Because subtraction is a linear operation, the mean of the difference image is equal to the difference of the means of the two images. Since $f(i,j)$ and $f(i-\Delta i, j-\Delta j)$ images have the same mean, the mean of the difference image is zero.

The adaptive gray scale mapping attempted to cancel dark areas with nearby bright areas. In neighborhoods with nothing but registration noise, the two bright and dark areas were canceled. At each pixel a brightness moment and a darkness moment were calculated. Assume that there were some true differences that were bright areas. The dark areas were scaled down to the point that the reduction equaled the increase of the dark area, like shifting part of a hill to fill a valley. Thus, the areas with true differences were not totally canceled out. This procedure assumed that the registration differences were close to each other. If not, the image had to be converted to an edge image via the Sobel operator. They found that this method worked better than smoothing without blurring the differences.

Herbin *et al.* (1989) developed a 6 parameter registration model consisting of x and y translations, a rotation, a magnification, and a 2 parameter linear transformation of the gray levels. They noted that classical similarity criteria such as correlation coefficient or correlation function could lead to misregistration if the images were known a priori to be dissimilar. Systematic calculation of similarity criteria for every possible value of the registration parameters could have led to prohibitive computation times. The problem of estimating the registration parameters was a problem of robust estimation in the presence of outliers. The authors proposed two methods of optimizing the similarity: the adaptive random search and an adapted simplex method. The random search performed faster.

Chiang and Sullivan (1993) noted that in conventional image registration, one of three kinds of measurement was usually chosen: correlation coefficient, correlation function and the sum of the absolute values of differences. A new class of similarity measures based on nonparametric statistical considerations had been proposed. The pixel

value differences were taken and the various displacement vectors were tested until the number of sign changes was maximized. This automatically took into account a symmetric noise distribution as well as misregistration.

If the noise was not symmetric such as Poisson noise, then the parametric method was not accurate. Because noise affected the lowest order bits of a pixel value, images could be registered according to the best matches between high order bits of pixel values. This method did not assume any noise distribution. This coincident bit counting (CBC) method was more robust under non-symmetric (one-sided) noise environments.

Matsopoulos *et al.* (1994) noted that, in medicine, information provided by MRI and CT was complementary. The authors described the process of combining information from the two modalities. The two major steps were registration and fusion (blending) of images. They noted that for fusion, morphological filters had advantages over linear filters, such as speed of computation and edge and feature preserving properties.

Van den Elsen *et al.* (1995) described an automated approach to register CT and MRI images. Using differential operators, edges were extracted from both images. Matching was then accomplished by correlation of edges only. The process was completely automatic. The matching was good even when the images only partially overlapped each other and when some of the features were not similar.

2.5.4 Image smoothing

Scollar *et al.* (1984) mentioned that for Gaussian noise, the variance of the mean was less than the variance of the median for a window of fixed size. When the tails of a probability density function increased, eventually the variance of the median became

smaller than the variance of the mean. They noted that algorithms based on means and standard deviations were faster than median methods.

The median absolute deviation (MAD) is a dispersion measure. It is the median of the absolute deviations from the original median. To calculate the MAD for an image, one needs to apply a median filter, subtract the result from the original, and then apply the median filter again to the absolute differences. The authors said that mean-standard deviation methods could usually be used if more than 400 pixels were used in the window, since the median approached the mean very quickly except at very broad edges. Thus, for windows greater than 19×19 , the authors used mean methods.

Davies (1988) also noted that median filtering was the most widely used technique. Median filtering is valuable for outlier rejection. If the noise was Gaussian and was uncorrelated within a neighborhood then the mean, which was simpler, would suffice. If, however, the noise was not known to be Gaussian, then the median might be the best choice because of its outlier rejection properties.

The mode filter seemed particularly suited to the enhancement objective and the median filter for simple noise suppression. Because modes were hard to calculate especially in small neighborhoods where the bins were exposed or where there were multiple modes, the author devised a “truncated median” filter that simulated the mode.

Hunt *et al.* (1990) looked at the line-detection performance of 3 processors for very noisy ($SNR < 1$) images with additive noise. They found that the Hough transform and Gaussian signal-detection theory were noise-insensitive compared to the Laplacian processor. For unknown noise, they recommended the Hough transform. For known

noise, they recommended the appropriate signal-detection theory or processor designed for the noise distribution at hand.

Rank and Unbehauen (1992) proposed an adaptive recursive filter for removing Gaussian noise. This filter employed smoothing where the surrounding pixels were scaled according to the output of three feature detectors. The smoothing operation was given by $y(m,n) = (1/k)[x(m,n) + \beta y(m,n-1) + \beta y(m-1,n-1) + \beta y(m-1,n) + \beta y(m-1,n+1)]$ where $k=1+4\beta$ and $x(m,n)$ was the pixel from the original image. The factor β was related to the output of an edge detector, a corner detector, and a flat detector. Therefore, the apparent features determined how much smoothing is done. The filter was recursive in the sense that the output of the filter was used to smooth other pixels. The filter ran left-to-right, top-to-bottom and then right-to-left, bottom to top to provide symmetric behavior. Slight edge enhancement was applied to compensate for the tendency of noise filtering to smear edges. The improvement in S/N ratio was about 6~3 dB and the algorithm took about 10 seconds per image.

Kundu and Zhou (1992) proposed a combination median filter to compensate for the tendency of median filters to smooth over fine lines. The median filter could not preserve thin line details although it did a good job of suppressing impulse noise and preserving edges. The multilevel median filter had been proposed to preserve small details. The multilevel median took the median of $1 \times n$ vertical, horizontal, and the two diagonals centered on a pixel. The largest and smallest medians of these 4 subwindows and the gray level of the center pixel were considered as candidates. Finally the median

of these 3 candidates replaced the center pixel. This multilevel median could preserve thin line details but the smoothing was inferior to the median filter.

The authors sought an improvement on the multilevel median filter by adding the choice of using a median filter, multilevel median filter, or a directional filter. The authors found the median values of the same 4 subwindows used by the multilevel median filter. Dixon's r-test was used to determine if one of the medians of the 4 subwindows was an outlier. Also, they tested if a thin line is present. The results determined which of the 3 types of median filters to use.

Let $f(x)$ be the probability density function of a certain population whose median is ξ . If x is a sample median of a sample size $(2n+1)$, then x_m is known to have an asymptotically normal distribution with mean ξ and variance $\sigma_n^2 = \frac{1}{4[f(\xi)]^2(2n+1)}$.

For a symmetric parent population, ξ is also the population mean. The variance of the sample mean of a finite population with Gaussian parent distribution can be found using

the following approximation: $c_n = \sqrt{\frac{\pi}{2} - \frac{0.2563}{2n+1} - \frac{0.0699}{(2n+1)^2} \dots}$ for $n = \text{odd}$. Also,

$c_n = \frac{\text{std.deviation of median of } (2n+1)\text{values}}{\text{std.deviation of average of } (2n+1)\text{values}}$. Since the standard deviation of the

average of $(2n+1)$ values is $\sigma/\sqrt{2n+1}$ where σ is the standard deviation of the parent normal population, c_n can be used to compute the standard deviation of the sample median.

Itagaki (1993) proposed an adaptive nonlinear filter with two parts. The first part used pixels in the 5x5 neighborhood that are in close gray value proximity to the center

pixel. So impulse noise was not included in the smoothing and the impulses were not altered either by the first part. The smaller the standard deviation of the noise, the smaller the window. So images with low noise were barely altered. Iterations could be applied. All that remains was the impulse noise, which could be easily removed.

Hussian and Reed (1994) presented a novel approach in image smoothing without blurring edges. They first segmented the image into regions (oversegmenting is acceptable), then all smoothing was done by standard low pass filters using 3x3 or 5x5 neighborhood averaging within segments. No smoothing across boundaries was permitted. This interruption of smoothing was what makes it nonlinear, otherwise linear filters were used. The noise over a wide range was smoothed and the edges preserved. The region-confined average enhanced contrast between regions, which sharpens the image. So noise was removed but the structures were retained. The *SNR* did not improve but the qualitative difference between this region approach and ordinary linear smoothing was striking.

Chen *et al.* (1994) noted that noise filtering of images was essentially a smoothing process, but simple low-pass filtering would blur image edges and other structures and thus damaged image fidelity. Not only were the structures significant to human viewers but they were also very important for automatic image analysis such as computer vision and image registration. In medical imaging, the sequence of images and the number of pixels per image required high processing efficiency as a fundamental consideration.

This paper described an algorithm that could be run in parallel to speed up the process. For each pixel, the algorithm ran a low pass filter in four directions: 0° , 45° ,

90°, & 135°. The type of low pass filter was not important. The absolute difference between the filter result and the original pixel value were compared. Low pass filtering along lines would have small differences and little distortion. Low pass filtering across lines would have large differences between filter output and the original input. The weights assigned to the four filters were calculated to emphasize the least distorting filter and de-emphasize the most distorting filters. In homogeneous areas, all filters were given equal weight. Therefore, the algorithm would minimize distortion around structures and consider larger areas for filtering in homogeneous areas. The paper also used the same method for edge sharpening. Since the four filters could be processed in parallel, speed could be increased.

Hanke *et al.* (1994) noted that median filters tended to fail at corners. At corners, the authors switched to direction sensitive medians, 4 one-dimensional medians, perpendicular and diagonal to the window.

Hwang and Haddad (1995) presented two new adaptive median filters. The ranked-order based adaptive median filter determined whether the center pixel was an impulse or not. If not, it left the pixel alone. If the center pixel was an impulse, it computed the median output. If the median output was an impulse (due to very dense noise) then the window size was increased iteratively until the output was not an impulse.

The second filter was an impulse-sized based adaptive median filter. This filter detected the width of the impulse and adjusted its window size to eliminate the impulse.

These filters performed better than the median filter under conditions of high noise density and strong impulse noise.

Karaman et al. (1995) presented a smoothing technique for speckle suppression for ultrasonic images. The basic idea was to grow a region from the center pixel. The region size was restricted to a square window and the criteria for aggregating pixels was that the gray level of the pixels was within a range of the center pixel. Then the center pixel was replaced by the mean (or median) of the region of similar pixels. The range of similarity was based on the typical variances of the speckles. So, pixels that were within the range were assumed to vary due to speckle noise and pixels outside of the range were assumed to be image structure details. As with the method of Itagaki (1993), small variations were suppressed and large variations were left alone. This was followed by a merging step of regions with similar statistics, and smoothing on the merged regions. This adaptive speckle suppression filter compared well to others in the literature. The basic assumption was that the speckle noise was small compared to the image structure details.

Li and Ramsingh (1995) proposed a multi-shell median filter. The concept of the multi-shell median filtering scheme was to develop a rotation invariant filter structure so that numerous subfilters for preserving details in different orientation were not required. Consider the median of a set of three numbers in 1-D:

$$\text{Median}[a_1, a_2, a_3] = \text{median}[\min[a_1, a_3], a_2, \max[a_1, a_3]].$$

This could easily be extended to rotating the 1-D median equivalent through a complete circle and collecting all the corresponding samples inside the min and max operators.

Let $S(m,n)$ be the set of samples surrounding the central sample. For a 3x3 filter window:

$$S(m,n) = \{ a(m-1,n-1), a(m-1,n), a(m-1,n+1), a(m,n-1), a(m,n+1), a(m+1,n-1), a(m+1,n), a(m+1,n+1) \}.$$

The output of a multi-shell median filter for the window size of 3x3 was

$$y_{m/shell}(m,n) = \text{median}[S_{\min}(m,n), a(m,n), S_{\max}(m,n)].$$

The filter was able to remove isolated impulses of one pixel in size and preserve features of size larger than or equal to two consecutive pixels regardless of feature orientation.

If we let $S_{(r)}(m,n) = r^{\text{th}}$ order statistic of $S(m,n)$; $r=1,2,3,4$ where $S_{(1)}(m,n) \leq S_{(2)}(m,n) \leq \dots \leq S_{(8)}(m,n)$. Then we could modify the 3x3 multi-shell filter to become a 3x3 multi-shell order statistics median filter by:

$$y_{m/shell(r)}(m,n) = \text{median}[S_{(r)}(m,n), a(m,n), S_{(9-r)}(m,n)].$$

The authors had shown that the 3x3 multistage max/median filter with 4 subfilters and the 3x3 median-based multi-shell order-statistics filter, $y_{m/shell(1)}(m,n)$ which was the multi-shell median filter, were identical. They also showed that when $r = 4$, the filter $y_{m/shell(4)}(m,n)$ was identical to the standard median filter.

The filters with $r \geq 2$ had better noise removal characteristics and filters with $r \leq 3$ had better detail preserving characteristics. Therefore, one could balance the noise removal and detail preservation by adjusting the rank value, r .

Soltanian-Zadeh *et al.* (1995) mentioned that the noise in an MRI scene sequence was characterized by an additive zero-mean white Gaussian noise field that was uncorrelated between different frames. Averaging several acquisitions of the same slice was the conventional method for reducing the additive noise. This method had the

following practical difficulties: 1. It increased imaging time and cost. 2. It limited throughput.

An alternative approach to reducing the noise was off-line conventional image restoration filters. However, conventional filters were not specifically designed for MRI. Recent developments in nonlinear filters like adaptive smoothing are indications of a need for new nonlinear methods to preserve edges while suppressing noise.

The new filter proposed by the authors used MRI signal models to implement an approximate maximum likelihood or least squares estimate of each pixel gray level from the gray levels for the same location in all the images in sequence; this corresponded to using interframe information.

In the author's paper they defined $SNR = \frac{E[g(i,j)]}{\sqrt{Var(g(i,j))}}$.

The usual assumption was that the statistical noise is modeled as a Gaussian distributed zero-mean white noise field with standard deviation σ . When using the background for estimating the noise standard deviation, the distribution was truncated because the pixel gray values were saturated on the lower side. Thus, the background result needed to be divided by 0.655 to yield to the proper noise standard deviation. The authors reported on a method of filtering using multispectral data for each MRI slice.

2.5.5 Morphological noise reduction

Cheng and Venetsanopoulos (1992) presented new opening and closing operators. These operators did not use any structuring elements. Rather a number of pixels, N , was specified. For gray level opening, any peak with less than N pixels in a cutoff cross section was flattened at that level. The shape of the cross-section was irrelevant. The

closing operator worked the same way, but it eliminated pits that had less than N pixels in its cross section. The adaptive part of the title meant that the cutoff worked or adapted to any cross section 8-connected shape. The filter effectively removed impulse noise (or any other feature) which was smaller than N pixels. The advantage over morphological operators using a group of structures was that there was no distortions or artificial patterns that might occur when a limited group of structuring elements was used for morphological operations. The authors developed an algorithm to implement the operator. Basically, for an opening, at a given pixel, one determined if there were more than 8-connected pixels with gray values higher than the current pixel. If not, the 8-connected pixels with higher gray values were all flattened, regardless of the shape.

Peters (1995) described a morphological image-cleaning algorithm. It was designed to enhance scanned or still-video images, or any image where the standard deviation of the noise was less than the standard deviation of the features. The goal of “cleaning up” images was subjective to most users, but two common characteristics of a good image were: 1) edges, thin lines and small features were sharp and clear, and 2) areas between these features were smoothly varying. Because of characteristics of human perception, a dark or bright line often appeared to be noise free even though the noise did not stop on the line. So the perception of low amplitude noise was largely within broader areas.

Linear filters invariably smooth out noise and blur fine features. Morphological filters are nonlinear filters that are used for noise reduction. The new algorithm used the difference image between an image smoothed with openings and closings and the original image. The difference image contained features as well as noise. Provided the

noise was low amplitude, then the difference image could be thresholded (or actually, center-clipped since the difference image was a signed image). The result was a support map or mask of the thin features in an image. The mask could then be used to recombine the thin features in the difference image with the smoothed image leaving the noise behind. The result was an image that was smoothly varying except for edges, thin lines, and small spots.

The morphological smoothing of images was accomplished by the average of the gray-scale open-close and close-open operations on an image. The structuring element was increased in size until all the noise from an image was removed. By using progressively larger structuring elements, the difference images between each step represented a morphological size distribution.

If the noise standard deviation was smaller than the features, thresholding could separate the two in the difference image. The author computed the overall square root of the second moment of the gray-levels in the difference image. The threshold was set by experimentation between 1 and 2 of these square-root second moments. The pixels in the difference image greater than the threshold were assumed to be features and not the low amplitude noise and became the support map.

The algorithm removed isolated pixels in the support map. The clipping or thresholding removed some of the base of a feature. This base was grown back by reattaching the portion of the residual image that was associated with the support map. The new residual image was recombined with the smoothed image.

The method worked better than other morphological methods and the median filter provided the noise amplitude was low. The disadvantages included the large number of

parameters that must be arrived at through experimentation. These included the number of difference image bands, the sizes of the structuring elements and the threshold factor at each difference band. The method performed well but was computationally expensive. A 1M image required 30 minutes on a Sun SPARCstation 1.

2.5.6 Image contrast enhancement

Wang *et al.* (1983) gave a survey of the current image enhancement techniques and found that most of the existing image-enhancement techniques were heuristic and problem-oriented. They noted that human perception followed a nonlinear transformation of the light intensity. Thus, $g'(x,y) = g_{\min} \left[\frac{g_{\max}}{g_{\min}} \right]^{p(x,y)}$ would give a logarithmic output histogram that appeared to human vision systems as having the most contrast.

Cocklin *et al.* (1983) characterized contrast enhancement as a monotonic remapping of the gray levels in an image. The methods of contrast enhancement fell into two broad categories: image content independent, image content dependent. The effectiveness of an enhancement technique should be balanced against the cost in terms of computer resources for its implementation. They stated that other studies have shown that X-ray film reading error rates may be as much as 20-30 per cent. Image enhancement techniques could selectively enhance particular features of interest while suppressing other information. Unsharp masking consists of superimposing a slightly blurred (averaged) negative on the original positive. This can be represented by $B = cA + (c-1)(-A)$ They noted that unsharp masking was a very common technique in medical

image processing primarily for its edge enhancement ability. Unsharp masking could be

performed by the convolution mask $\frac{1}{9} \begin{bmatrix} 1-c & 1-c & 1-c \\ 1-c & 1+8c & 1-c \\ 1-c & 1-c & 1-c \end{bmatrix}$

Moore (1986) proposed that most image enhancement techniques could be characterized by the change they produced in the histogram entropy. The histogram entropy was given by $\sum_{s=1}^q P(s) \log_2 P(s)$ where s is a gray level and q is the total number of gray levels. This expression gave the statistical information content of an image in terms of the average number of bits required per pixel. For unconstrained cases, the histogram entropy was a minimum when all but one $P(s)$ values were zero and a maximum when all $P(s)$ values were equal. The corresponding histogram entropies were zero and $\log_2(q)$ respectively. In imaging terms, the two extremes corresponded to highly structured and complex, poorly structured pictures, respectively.

The author looked at the effect on histogram entropy by various image enhancement methods. He pointed out that edge-enhancing methods were histogram entropy reducing. Edge-enhancing techniques improved edge visibility by removing redundant information (pixel correlation), which lowered entropy: Contrast emphasizing techniques tended to be histogram entropy increasing. Contrast emphasizing methods produced more uniform histograms, which gave the highest entropy.

Pizer et al. (1987) introduced interpolated adaptive histogram equalization where local histograms of each tile of an image was equalized and the mapping of any pixel

was based on the bilinear interpolation of the mapping in the 4 nearest tiles. This dramatically speeded up the processing time over the local histogram method.

Weighted adaptive histogram equalization where the histogram was weighted by the proximity of the other pixels was investigated and found no improvement but more computation burden.

The tiling size was recommended to be $1/16$ th of the image area. Less than $1/64$ th area caused oversensitivity and artifacts. Also overlapping tiling had no advantage over mosaic tiling.

In a homogeneous region, histogram equalization would create too much contrast, enhance noise and create artifacts. Since a uniform region would yield a histogram with a high peak, the histogram was clipped at a user specified-level. So the histogram was not actually equalized because the stretching of low contrast areas was limited. The authors recommended the clipped variation because of ease of computation and comparable results to interactive windowing.

Leszczynski and Shalev (1989) noted that local histogram modification was an approach where a square neighborhood was defined around a pixel. The histogram was equalized and the center pixel was changed according to this histogram equalization. This step was performed over every pixel. Therefore, the main drawback was a heavy computational burden. Adaptive histogram equalization involved dividing the image into an integer number of tiles. Histogram equalization was performed on each tile independently. The modified value for each pixel was then calculated by bilinear interpolation for the four closest tiles.

The authors proposed a moving histogram equalization. This was a very similar to the adaptive histogram method. However, only a horizontal (or vertical) strip was tiled. All the points on the row (or column) of the center points of the tiled region were calculated by interpolation just between two tiles. The strip was shifted down (or right) by one pixel. This allowed a quick updating of the cumulative density function as one row (column) or pixels was dropped and another row (column) was added. Again, another row (column) of pixels was modified by interpolation between just two regions. This continued until the image had been covered. The outside edges of the image were unaltered. The computational load of the moving histogram equalization was comparable to the adaptive histogram equalization, but the enhancement of the image was superior. Adaptive histogram equalization could leave artifacts. The results of moving histogram equalization method was comparable to the heavily computational local histogram equalization method while having the light computation load of the adaptive histogram equalization.

Dale-Jones and Tjahjadi (1992) proposed four algorithms for images with large peaks in the histogram. They pointed out that if there were large peaks in the histogram that were composed of pixels from uniform areas of the image, and smaller bins that contained important image information, then the histogram equalization algorithm would not be effective in enhancing this image.

The first algorithm stretched peaks that are above a user-specified level and histogram bins below the same level were not merged. This method needed empty histogram bins to start.

The second algorithm was the opposite of the first. All peaks above a user-specified level were compressed into one bin. The remaining bins below the level were stretched according to their respective height.

The third algorithm required that a second parameter was chosen. All bins smaller than the second parameter were preserved. All peaks above the first parameter were compressed into one bin. All bins with heights between the two parameters were stretched.

The fourth algorithm was like the third. The difference was that all bins with heights between the two parameters were stretched in inverse proportion to their size. The rationale was that large peaks typically represented background. So compression would not lose information.

The authors noted that a large Gaussian distribution with a large population and a large standard deviation implied a broad peak that contained most of the image population. Ordinary histogram equalizations would produce a good result for this type of image, because most bins would be stretched and only a few bins would be compressed.

The authors also noted that local histogram equalization or local histogram modification is similar in operation to local operators such as edge detectors. A square neighborhood was defined around each pixel in the image. For each pixel, the histogram of the neighborhood was calculated and equalized. The pixel was assigned the value from the transformation that was obtained by the equalization. Local histogram equalization would generate artificial noise in backgrounds because it produced a powerful stretch of all parts of the image. They concluded that although the image

revealed much more detail than a global operator, indiscriminate stretching introduced artifacts. The algorithm would be more effective if it were possible to limit the stretching to only a required area of the image.

Deng *et al.* (1995) noted a previously existing simple algorithm for image enhancement, which could be expressed as $F'(i,j) = \alpha A(i,j) + \eta + \beta (F(i,j) - A(i,j))$ where $A(i,j)$ is average of neighborhood and $F(i,j)$ is original input, α is a rescaling factor, β is a sharpness factor. A logarithmic version of this algorithm had been proposed. It followed saturation characteristics of the human vision system. Details in very dark and very bright areas were enhanced by adjusting α . Sharpness was enhanced by adjusting β . One drawback was that noise was amplified while sharpness was increased. The previous method of linear contrast stretching was limited when the original image histogram was broad. The logarithmic version would increase contrast better.

Russo and Ramponi (1995) described a set of fuzzy if-then rules that performed both sharpening of details and smoothing of broad areas. In a neighborhood, the difference of the gray level of each pixel from the center pixel was calculated. Then each difference was assigned a membership value to classes such as large negative difference, small negative, small positive, and large positive. These initial conditions (antecedent clauses) were linked by “fuzzy and” operators and applied to the set of fuzzy rules. The outputs of each rule were changes in gray level value such as large negative change, small negative change zero change, small positive change, and large positive change. The author suggested a complete set of rules, along with membership functions for the input and output. The results were comparable to the unsharp masking technique.

2.6 Validation of Regression For Prediction Purposes

2.6.1 Regression for prediction purposes

When faced with a data on a set of regressors, one is confronted with the problem of finding the best subset of regressors to use. The best subset is dependent on the ultimate purpose of the regression model. Some purposes include: (1) learning something about the system under study, (2) learning which regressors are important, or (3) using the regression model for prediction. This study will use the regression results to predict the best parameters to use in the algorithm based upon the characteristics of the image. Thus, the prediction capability of the regression is of paramount importance.

Some of the traditional criteria for comparing models such as the coefficient of determination, R^2 , the estimate of error variance, s^2 , and the adjusted coefficient of variation, adjusted R^2 , are not based upon prediction capability. These criteria are based upon the ordinary residual, $y_i - \hat{y}_i$. “The ordinary residuals ... are not generally indicative of how the regression model will predict. Indeed, the least squares procedure is designed to produce properties in the regression function that will result in residuals that are smaller than true prediction errors; one must be reminded that \hat{y}_i is not independent of y_i and is, in effect, drawn to it. These residuals are measures of the quality of fit and do not assess quality of future prediction” (Myers 1990).

2.6.2 Validation of regression model

To measure the prediction capability, a set of data points should be selected that will be indicative of the range under which the model will perform. This selected data set should be independent of the data set that is used to generate the regression model. If

one has a finite data set and no additional data, one practice of validation is generating internal estimates by means of data-splitting, which is the partitioning of a data set into a fitting set and a validation set.

The candidate model can be fit and the coefficients can be estimated with the fitting set. The fitted model is then used to estimate the responses on the validation set, \hat{y}_i . The prediction errors, $y_i - \hat{y}_i$, on the validation set can be used to construct measures such as $\sum (y_i - \hat{y}_i)^2$ or $\sum |y_i - \hat{y}_i|$. These prediction measures on the validation set can then be used for comparison of various candidate regression models or to make projections of future prediction errors.

Frequently, the fitting set is taken to be 2/3 of the original data set and the validation set is 1/3 of the original data set. However, there is no theoretical justification for this split (Breiman *et al.* 1984). The drawback to this simple approach is that it reduces the effective sample size for fitting and for estimating the prediction error. If the data set is small, then this approach can be inefficient in the use of the available data.

2.6.3 Cross-validation

When the data set is small, v-fold cross-validation is preferred (Breiman *et al.* 1984). In v-fold cross validation, the data set is subdivided into v subsets of as nearly equal size as possible. Let the data set is denoted by L and the v subsets of data be denoted by L_1, L_2, \dots, L_v . For the i^{th} subset L_i , the remaining data points, $L - L_i$, are used to estimate the coefficients for the regression model. The responses for the i^{th} data set, $\hat{y}_j, j \in L_i$, are estimated from the regression model. The actual responses of the data set, L_i , are used to compute the prediction errors for that data set, $y_j - \hat{y}_j = e_j^*$. The

prediction, \hat{y}_j , is independent of y_j because y_j was not used in estimating the coefficients. The process is repeated for all ν data sets resulting in N prediction errors. The estimate of the prediction error as computed by Breiman, *et al.* (1984) is given by

$$\text{MCV}_\nu = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2 = \frac{1}{N} \sum_{i=1}^N (e_i^*)^2.$$

For large ν , the ν learning sets are nearly as large as the original data set. Also, the number of responses used to estimate the prediction error is equal to the original data set size.

Breiman *et al.* (1984) have noted in their work on regression trees that computational burden for cross-validation goes up linearly with ν . Also, as ν increases the accuracy of the prediction error estimate increases. Below 10, they noticed that the estimation performance was degraded. At $\nu = 10$ and above, they noticed no significant improvement in estimation performance. For their purposes, $\nu = 10$ represented a good compromise between computational burden and estimation accuracy. Burman (1989) has shown that the bias of the ν -fold prediction estimate decreases as ν increases until the bias is negligible when $\nu = n$, the number of data points. In addition, Zhang (1993) has shown that the most dramatic improvement in the estimate of prediction error occurs between $\nu = 2$ and $\nu = 10$. For $\nu > 10$, the improvement is small. If the intent of ν -fold cross validation is to reduce computation, then $\nu = 10$ is again shown to be a good compromise between computational burden and estimation accuracy.

2.6.4 N-fold cross-validation

The PRESS (Prediction Sum of Squares) statistic represents the simple leave-one-out case of cross validation (CV). This method is also called N-fold cross-validation as it represents the largest ν possible for ν -fold cross-validation. This method of data splitting

involves leaving one data point out and using the remaining data points to estimate the coefficients for the regression model. The left-out data point is used to compute the prediction error for that data point, called the PRESS residual, $y_i - \hat{y}_{i,-i} = e_{i,-i}$. The prediction, $\hat{y}_{i,-i}$, is independent of y_i because y_i was not used in estimating the coefficients. The process is repeated for all N data points resulting in N prediction errors.

The PRESS statistic is computed as
$$\text{PRESS} = \sum_{i=1}^n (y_i - \hat{y}_{i,-i})^2 = \sum_{i=1}^n (e_{i,-i})^2.$$

The computational burden would appear to be great, but with linear regression, there is an alternative method that does not require repeated regressions to be run. The PRESS residual can be calculated from the ordinary residual, namely, $e_{i,-i} = \frac{e_i}{1 - h_{ii}}$, where h_{ii} is the i^{th} element of the diagonal of the HAT matrix. The HAT matrix is defined from the set of independent variables, \mathbf{X} , namely, $\mathbf{H} = \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'$.

2.6.5 Usefulness of individual PRESS residuals

Because the PRESS residual is the prediction error from a regression that does not include the corresponding individual data point, the size of the individual PRESS residual also indicates the degree that the data point would influence the outcome of a regression of all the data points. That is because the regression, in an attempt to minimize the squared error, would move the fitted \hat{y}_i toward the observation y_i . A large PRESS residual would mean that there would be a much stronger influence to move the regression plane to the particular observation when the observation is included in the data. Thus, inclusion of the data point associated with a large PRESS would influence the outcome of the regression parameters.

The HAT diagonal element h_{ii} is the standardized distance for data point \mathbf{x}_i from the center of the regressor space. Data points at the extreme borders of the regressor space would have more potential for influencing the regression outcome due to their higher “leverage.” Whether that potential influence is realized depends on the degree that the data point in question falls out of line with the other data points (the actual regression error). Thus, the PRESS residual $e_{i,-i} = \frac{e_i}{1 - h_{ii}}$ indicates the influence on the regression because it considers both the actual error e_i and the potential for influence h_{ii} together. An example of an influential data point is given in Figure 2.1.

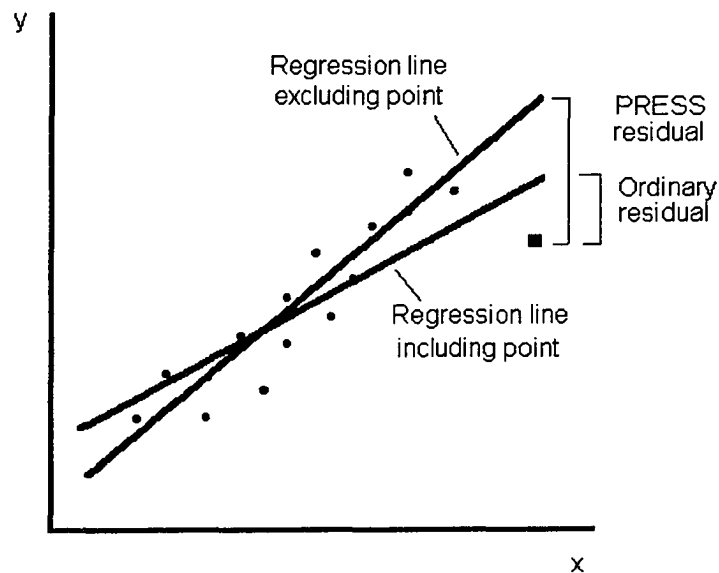


Figure 2. 1 Example of an Influential Data Point

A high PRESS residual should signal a reinvestigation of a particular data point. “If a serious problem is found through a reevaluation of a high influence point, then the point’s presence should surely be questioned. However, if reevaluation verifies that an

influential point is a valid observation, there can be no reasonable justification for its removal” (Myers, 1990).

2.6.6 Problems of autocorrelation

Error terms correlated over time (or space) are said to be autocorrelated or serially correlated. If autocorrelation exists there are a number of problems including:

1. The estimated regression coefficients are still unbiased but may be inefficient.
2. The mean squared error may seriously underestimate the variance of the error terms.
3. The estimated standard deviations of the coefficients may seriously underestimate the true standard deviation of the regression coefficients.
4. The confidence intervals and t and F tests are no longer strictly applicable (Neter *et al.*, 1989).

A simple linear regression model with first order autoregressive error terms is:

$$Y_t = \beta_0 + \beta_1 + \varepsilon_t$$

$$\varepsilon_t = \rho\varepsilon_{t-1} + u_t$$

where: ρ is a parameter such that $|\rho| < 1$ and u_t are independent $N(0, \sigma^2)$. It can be shown (Neter *et al.*, 1989) that the autocorrelation coefficient ρ is also the coefficient of correlation between adjacent error terms. In addition, the coefficient of error terms that are s periods apart is ρ^s . Therefore, all error terms are correlated but because $|\rho| < 1$, the correlation is smaller, the further apart the terms are. Error terms are uncorrelated only when ρ is 0.

CHAPTER 3

METHODOLOGY

3.1 Overview

This first section, 3.1, describes the data and equipment that was available as well as the preliminary findings upon which the methodology was based. The remaining sections explain the methodology that was used. Those sections fall under the three main research objectives and are listed as follows:

- (1) obtain a set of algorithms that identify internal defects from MRI spin-echo images

Section 3.2 Development of Defect Detection Algorithm for Spin-Echo Images

Section 3.3 Automatic Determination of Parameters for Spin-Echo Defect

Detection Algorithm

Section 3.4 Comparison of Spin-Echo Defect Detection Algorithm and

Thresholding Methods

- (2) obtain a set of algorithms that identify the same set of internal defects from MRI echo-planar images that are found from the spin-echo images

Section 3.5 Development of Defect Detection Algorithm for Echo-Planar Images

Section 3.6 Finding the Best General Image-Enhancement Method for Echo-Planar
Images

Section 3.7 Using adjacent slices to eliminate noise in echo-planar images

Section 3.8 Comparing defect detection capability of processed echo-planar
images

- (3) compare differences between the defects obtained from the two methods of MRI imaging

Section 3.9 Determining Defect Detection Losses Between Spin-Echo and Echo-Planar Images

3.1.1 Data and equipment used in the research

The data for the research consists of 60 MRI scans of a black oak log taken across the longitudinal axis 10 mm apart using the spin-echo method. This method produces high-quality images and requires about 40 seconds of acquisition time per scan. These scans were recorded on 256x256 gray scale images. The same log was scanned in the same locations using the echo-planar method. The echo-planar method is quite fast. The acquisition time is about 1 second, but the quality of the image is quite poor. Because of memory space limitations at the time of scanning, the echo-planar images were recorded on 256x128 images. This slightly cropped the top and bottom edges of the log, but the scans are otherwise intact. Of the 60 scans by each method, two scan locations were located before the beginning of the log and one scan location went past the end of the log. Two more scans captured tapered ends of the log. Therefore only 55 scans by each method are useful. The data in original digital form and in film were provided by Dr. Sun Joseph Chang and the Department of Forestry at Louisiana State University. One of the spin-echo MRI scans is given in Figure 3.1. An actual photo of the equivalent sawed open cross section is given in Figure 3.2.

The use of Sun Sparc 2 computer workstations for image processing was provided by Dr. Charles Harlow and the Remote Sensing and Image Processing Center in the College of Engineering at Louisiana State University. The Khoros 1.0 software package

was used for utility purposes and prototype development. Khoros software is written in C and is widely used in academic and industrial institutions. The custom programs developed in this research were implemented in ANSI C language.



Figure 3. 1 MRI Spin-echo Scan



Figure 3. 2 Photo of Cross Section

3.1.2 Preliminary findings

As noted in previous studies (Wang and Chang 1986, Chang *et al.* 1987, Hailey and Swanson 1987, Chang *et al.* 1989, Flibotte *et al.* 1990, Chang *et al.* 1991), various internal defects become apparent on MRI scans. From Figure 3.3 come several observations:

- (1) the surrounding atmosphere is very dark due to low moisture content,
- (2) the bark does not show up which is an advantage,
- (3) knots appear as dark regions,
- (4) oak rays and growth rings are detectable though not always clearly,
- (5) the heartwood and sapwood regions are apparent,

- (6) one scar tissue area in the form of a growth ring in the middle of the sapwood region is detectable by both very dark areas and very bright areas,
- (7) a second scar tissue area near the edge of the log in the first quadrant is detectable by a dark area on the MRI scan,
- (8) the pith of the tree is detectable by both bright and dark areas, and
- (9) the clear wood areas show variation in gray levels due to overall variations of moisture in the tree and minute variations form growth rings and rays.

Based on all the MRI spin-echo scans available, one of the preliminary findings is that all the defect regions have some pixels with gray levels which deviate widely either above or below the clear wood region gray levels. This can be observed in Figures 3.3 and 3.4. Figure 3.4 gives a profile slice view of the gray levels already given in Figure 3.3 This profile runs from top to bottom passing through the log center (the pith).

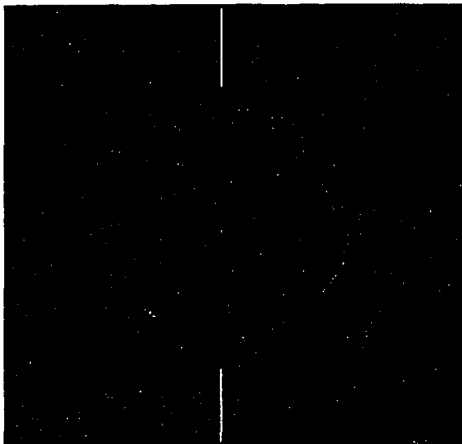


Figure 3. 3 Spin-echo Scan With Knot

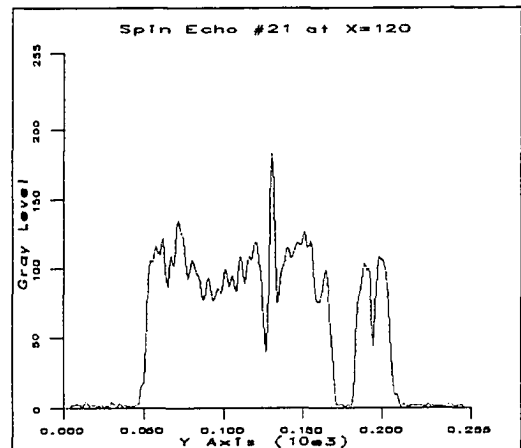


Figure 3. 4 Center Profile of Figure 3. 3

Next, Figure 3.5 and Figure 3.6 show the same views after the MRI image has been processed with a median filter (window size = 17 x 17 pixels). Finally, the deviation of the gray level from its neighborhood median is given in Figures 3.7 and 3.8.

Figures 3.7 and 3.8 show that the largest deviations are associated with defects. These large deviations can become seeds or indications for the defect regions.

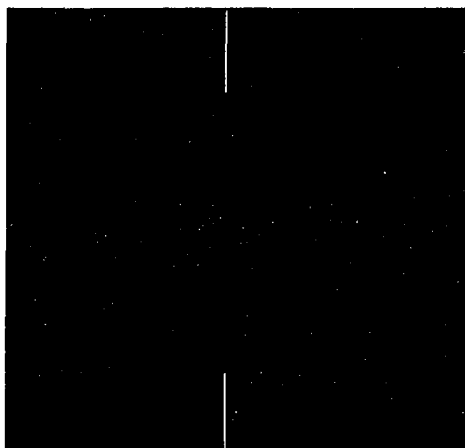


Figure 3. 5 Scan with Median Filtering

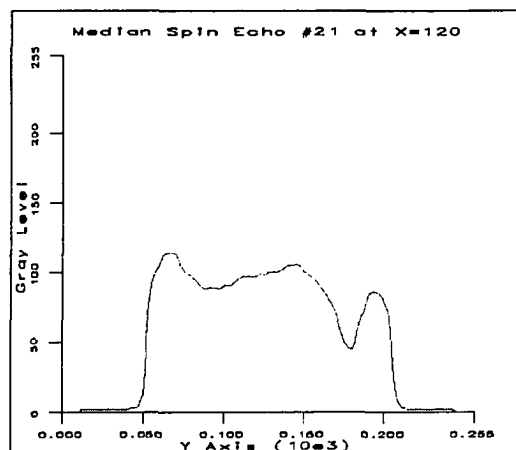


Figure 3. 6 Center Profile of Figure 3.5

By comparing Figure 3.3 with Figure 3.7, it can be seen that not all defect region gray levels deviate widely from the neighboring clear wood pixels. It can be seen from Figure 3.4 that the defect regions also contain pixels whose gray levels overlap with clear wood pixels.

Figure 3.9 shows the signed difference between a pixel's gray level and the median of its neighborhood. Figure 3.10 gives a comparison of the data from Figure 3.4 and Figure 3.9 that seems to indicate that the defect region borders coincide at the point where the pixel gray levels approach the median gray levels.

3.2 Development of Defect Detection Algorithm for Spin-echo Images

3.2.1 Overview

A program was written in ANSI C to carry out the operations to exploit the unique features of the images that were covered in Section 3.1.2. The development of the defect detection algorithm can be broken down into 4 main steps:

Section 3.2.2 Separation of foreground and background

Section 3.2.3 Determination of average gray levels of each pixel's neighborhood

Section 3.2.4 Defect region seed determination

Section 3.2.5 Growing defect regions

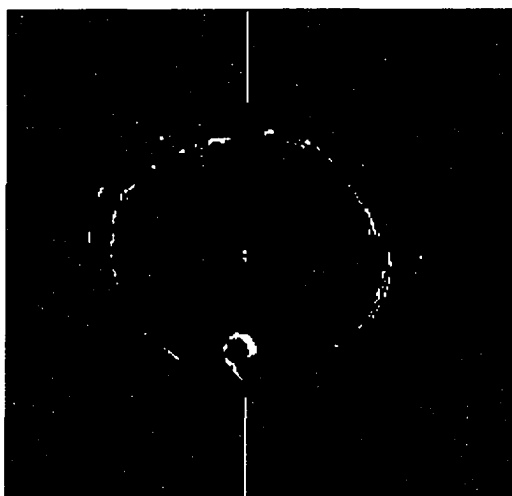


Figure 3. 7 Pixels With Absolute Difference > 40 Between Original Image And Median-Filtered Image

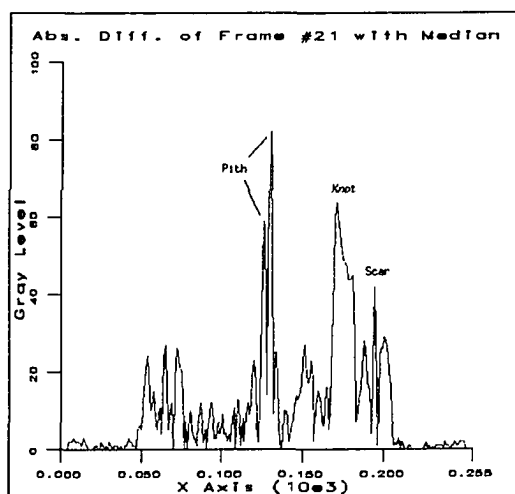


Figure 3. 8 Profile of Absolute Difference Image Between Original Image and Median-Filtered Image

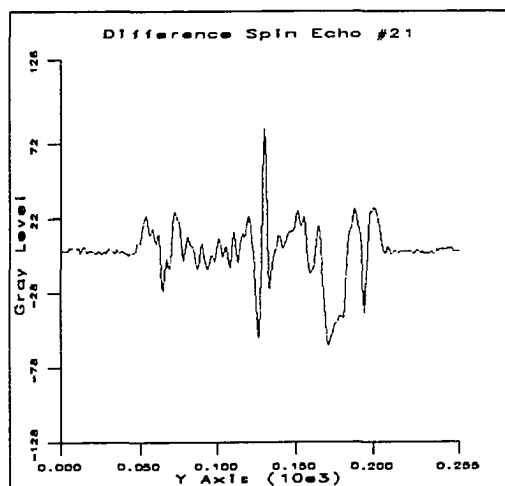


Figure 3. 9 Signed Difference Image Between Original Image and Median-Filtered Image

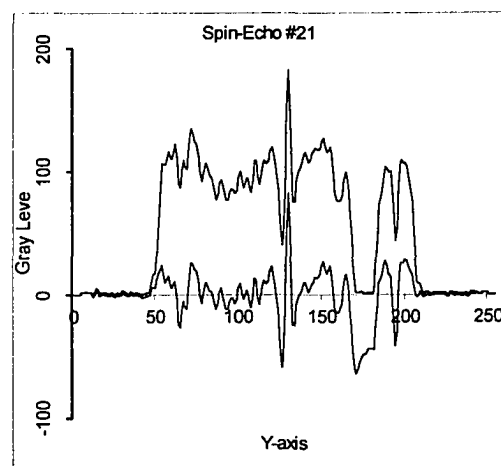


Figure 3. 10 Comparison of Original Image and Difference Image

3.2.2 Separation of foreground and background

This program first separated the foreground of the image (the log) from the background of the image (the surrounding air) automatically using the moment-preserving threshold method (Tsai, 1985). The moment-preserving thresholding method was chosen because it works automatically without iteration and search. That is important because that allows the method to be very fast. The method requires that the number of threshold levels must be known *a priori*. Since it is known that the images will be separated into a foreground and a background, the number of threshold levels will always be one and the requirement will never present a problem.

Even after the threshold that separates the dark air background from the lighter log foreground is determined, an additional step was required to insure that the dark regions such as knots inside of the log are not classified as background. Since the background is a connected region, the background was determined by starting in the top left corner, coordinates (0,0), of the image and adding all 4-connected pixels whose gray level value came under the threshold. Thus all pixels within the border of the foreground region were never considered regardless of their gray level value. A computer subroutine, `threshold.c`, was written in ANSI C to accomplish that and is given in Appendix A. Several examples of the separation of the log region from the background by the moment-preserving threshold and the `threshold.c` subroutine are given in Figures 4.2, 4.4, 4.6, 4.8, and 4.10 in Chapter 4.

3.2.3 Determination of average gray levels of each pixel's neighborhood

Once the foreground was defined, then the program generated a smoothed image by passing the original image through an averaging filter. The window size of the averaging filter was input to the program. The program was written so that the filter processed only the foreground pixels. When the averaging window approached the edges of the foreground, the window was still centered over the appropriate pixel but only the foreground pixels were used to calculate the neighborhood average. An example of the output of this routine was already given in Figure 3.5. The routine for this task, `smoothing.c`, is listed in Appendix A.

3.2.4 Defect region seed detection

Even in clear wood areas, the gray levels vary due to growth rings and to a lesser extent, random noise. The average moisture and therefore the average gray level varies even in clear wood because there can be a natural variation within a tree. For example, the heartwood moisture content can vary from the sapwood moisture content as seen from Figure 3.4. This verifies the observations by Birkeland and Holoyen (1987) that the moisture content profile of a log has high edges in the sapwood area and a bowl region in the heartwood area. Constant threshold limits would not be appropriate when the average level of gray levels in clear wood varies naturally. Threshold limits that follow the contour of the average moisture content would seem to be more effective in ignoring the natural variation due to growth rings and isolating the unusual variation associated with defect regions.

A difference image between the original image and the averaged image was calculated pixel by pixel with subroutine `subtract_image.c` listed in Appendix A. This

would, in effect, remove the influence due to the variation of the average moisture level within the log.

Next, the pixels of the difference image were passed through a positive threshold and a negative threshold (that were input by the user into the program) to capture the pixels where the deviations from the average neighborhood levels were excessive. These pixels served as seeds for the defect regions. The idea of the construction of the variable threshold limits is shown in Figure 3.11 and the application to a profile of an image is shown in Figure 3.12.

An example of the output of this routine was already given in Figure 3.7. Additionally, several examples of the output of this routine are given in Figures 4.11, 4.13, 4.15, 4.17, and 4.19. The subroutine, `spin_defects.c`, to do this task is listed in Appendix A. All defects on all spin-echo scans were then manually checked to confirm that every defect region has at least some pixels that vary from the median of its neighborhood by more than the thresholds.

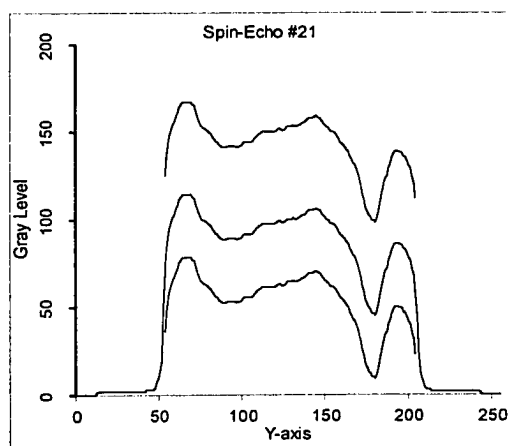


Figure 3. 11 Construction of Variable Threshold Limits

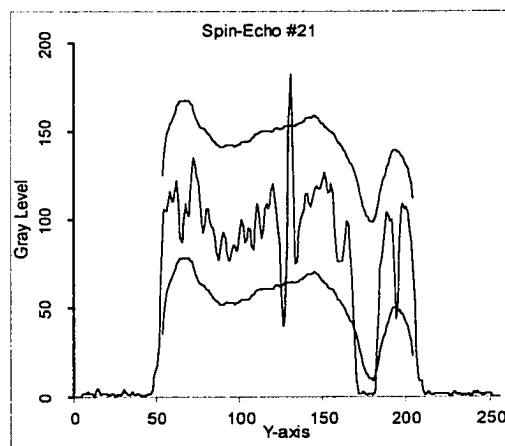


Figure 3. 12 Application of Threshold Limits to Original Image

3.2.5 Growing defect regions

Finally, using a region growing method, adjacent pixels (4-connected or 8-connected, according to the user input) were added to the defect region seeds. A stopping rule has to be specified so that the program will know when to stop adding pixels. Otherwise, the regions would continue to grow until the entire image was covered. The defect regions gradually blend into the surrounding clear wood. Thus the gray levels of the defect region approach the average gray levels of the surrounding areas. Therefore, any thresholding technique alone would clip the edges of the defect region area away from the true defect edges. At first, the proposed algorithm attempted to grow the defect regions around the seeds until the added pixel gray levels reached the average levels of the neighborhood. The method used is a variation of the conditional dilation method described by Vincent (1993). In this case the region is not predefined and the condition to be checked is that the gray level of the neighboring pixel to the region is on the same side of the median as the region seed. This region-growing step was created to recover the edges of the defect regions that were not identified by threshold methods. Using the average gray level of the neighborhood as the stopping rule for adding adjacent pixels allowed the defect region to grow too far. The stopping rule was then modified to stop the region growing process before the average gray levels were reached. The user can specify the stopping-rule limits. Several examples of the output of this routine are given in Figures 4.12, 4.14, 4.16, 4.18, and 4.20. The subroutine, `region_grow.c`, to do this task is listed in Appendix A.

3.3 Automatic Determination of Parameters for Spin-Echo Defect Detection Algorithm

3.3.1 Overview

At this point, the programming was completed to allow the defect regions to be clearly identified. However, the algorithm depended on several parameters. These parameters are:

- (1) the type of connectivity (4-connected or 8-connected) to use for region-growing,
- (2) the window size to use for the average filter,
- (3) the normal variation threshold limit above the neighborhood average,
- (4) the normal variation threshold limit below the neighborhood average,
- (5) the approach limit above the neighborhood average to stop region growing, and
- (6) the approach limit below the neighborhood average to stop region growing.

The last five parameters tended to be unique for each image. Therefore, in order to handle the variety of possible images that could be presented from natural phenomena such as a log, it would be necessary to make the algorithm more robust. The sections listed below describe the method used to make the algorithm more robust:

Section 3.3.2 Averaging window size and region-growing connectivity determination

Section 3.3.3 Manual determination of parameters for each spin-echo image

Section 3.3.4 Regression of image statistics to parameters for automatic determination of parameters

3.3.2 Averaging window size and region-growing connectivity determination

The preliminary results that were discussed in Section 3.1.2 indicated that a smoothed image was useful for removing the influence due to the variation of the

average moisture level within the log. The purpose of the smoothing operation is to allow a subsequent subtraction of the smoothed image from the original image to isolate or at least emphasize the defect regions. The choice of the average-filter window size should be made with this purpose in mind.

Very small averaging windows will follow the data too closely and not allow the defect regions to stand out. This can be demonstrated in Figure 3.13, which has an original profile with various smoothed profiles that are superimposed. Note that the smoothed profile generated by a window of size 10 follows the data very closely. This window size isolates the pith defects at coordinates 127 and 131 and the ring defect at coordinate 195 because the width of these defects are small. However, the smoothed profile follows the large knot at coordinate 175 too closely all the way down. Thus, the knot is not isolated.

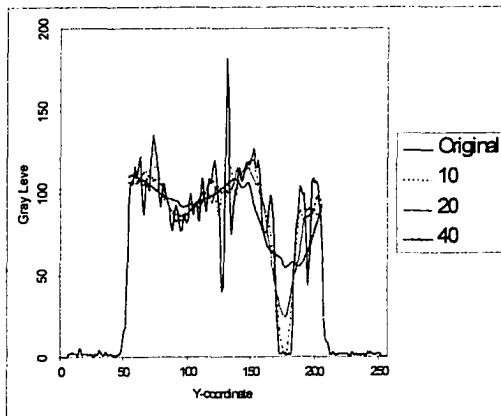


Figure 3. 13 Original Profile with Various Window Size Smoothed Profiles

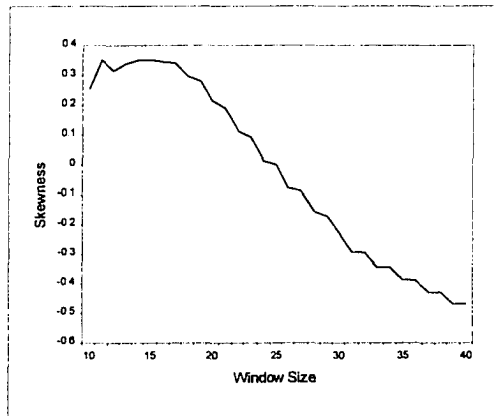


Figure 3. 14 Skewness of Residuals versus Averaging Window Size

Very large averaging windows will not follow the average moisture level properly. In Figure 3.13, the smoothed profile generated by the window size of 40 does not follow the average moisture profile properly from coordinate 140 to coordinate 200.

The smoothed profile between these two points is pulled downward due to the influence of the large knot at coordinate 175. The knot and the pith have been correctly isolated. However, the ring defect at coordinate 195 is not isolated. In addition, the normal wood at coordinate 190 between the knot and the ring defect has been emphasized as though it was a defect region.

A moderately-sized averaging window will isolate all the defects. Note, in Figure 3.13, that the smoothed profile generated by a window of size 20 follows the average moisture trend. From coordinates 140 through 165, the smoothed profile performs better than the size-40 window profile since it is not being influenced by the knot. In the knot region, the size- 20 window profile outperforms the size-10 window profile because it isolates the knot. Finally, the size-20 window profile “recovers” from the knot quickly enough so that at coordinate 195, it isolates the ring defect as well. In effect, the window size of 20 is better suited at isolating both large and small defects while not being unduly influenced by nearby large defects.

Rather than manually determining the window size, an automatic method would make the algorithm more robust. As mentioned earlier in this section, the purpose of the smoothing operation is to allow a subsequent subtraction of the smoothed image from the original image to isolate or emphasize the defect regions. In conjunction with that, it has been found that knots are the most important defect type encountered in lumber as far as grading is concerned. In addition, knots are also the most numerous type of defect (Harding *et al.* 1993).

Knots appear as dark areas in MRI images. When a smoothed image is subtracted from the original image, the knots (and some other defects) would be identified by the

extreme negative residuals as shown in Figure 3.10. A good window size would be one that maximizes the emphasis of the defects. Since trees are known to have more knot defects (dark defects) than any other kind, a good window size will generate a higher proportion of negative residuals. The resulting histogram of residuals will have a higher positive skewness because of the concentration of residuals at the lower end and a “tail” at the higher end of the histogram.

From the previous discussion on Figure 3.13, the window size of 10 was too small and the window size of 40 was too large. The skewness of the residuals from each window size from 10 to 40 is plotted in Figure 3.14. The plot indicates that a window size of 15 would be most appropriate. For the same reasons given in the previous discussion for a window size of 20, a window size of 15 does a good job of emphasizing all the defects.

Other examples also bear out the appropriateness of using the maximum skewness to indicate the best window size. Figure 3.15 illustrates a profile that features a sequence of knots at coordinates 145, 160, 170, 180 and 200. The window sizes of 10, 20 and 40 all appear to isolate most of the knots. However, the window size of 40 does a better job of isolating the knot at coordinate 200. Incidentally, the plot of skewness of residuals for the various window sizes in Figure 3.16 indicates that a window size of 40 or larger is most appropriate.

From these examples, the skewness of the residuals reflects that the window size should be adjusted according to the size of the defects encountered. The profile in Figure 3.13 calls for a moderately sized window that will not be so small that it follows that

knot and yet it is not so large that it can “recover” quickly. The profile in Figure 3.15 calls for a large window because of the numerous defects that are clumped together.

The choice of the average-filter window size will be made on the basis of emphasizing the dark defects, which are more numerous. This will be accomplished by determining which window size causes a peak in skewness of the residuals between the original image and the smoothed image. A computer program, `peak_skew.c`, was written to find the window size with the peak skewness and is listed in Appendix A.

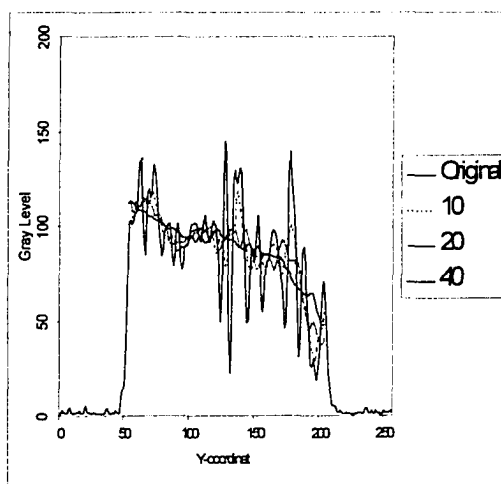


Figure 3. 15 Original Profile with Various Window Size Smoothed Profiles

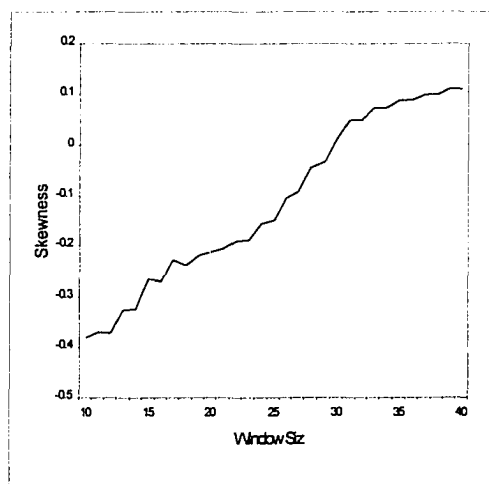


Figure 3. 16 Skewness of Residuals versus Averaging Window Size

Also, it was evident that the 8-connectedness approach to region growing allowed the defect regions to leak out into the clear wood areas. So the 4-connectedness approach was used to minimize leakage of the defect regions into the clear wood areas.

The remaining four parameters presented more of a problem because they were different for each image and the results were highly sensitive to the parameter settings. The remaining section describes how the parameters for each frame were automatically determined.

3.3.3 Manual determination of parameters for each spin-echo image

The best algorithm parameters were different for each spin-echo image. The next approach involved finding the best threshold limits for the deviation of the gray levels from the average background level so that the normal variation was ignored and the unusual variation due to defects was detected. Also, the best region-growing stopping limits were determined for each frame. In order to do this, the defect regions were first visually identified. Then, a loss function was devised to measure the effectiveness of a parameter value. For example, with the upper and lower limit parameters, the goal is to set the parameters so that as many seeds are generated in the defect regions as possible (to capture as many regions as possible) and at the same time avoid generating seeds in the clear wood areas. The function used is given below:

$$L(p) = \delta(D - i) + (1 - \delta)j, \text{ where:}$$

L = loss function

p = parameter value used in algorithm

δ = weight factor between 0 and 1

D = number of pixels in previously-identified defect area

i = number of seeds generated in defect regions

j = number of seeds generated in clear wood areas

The first term on the right side of the equation is the penalty for not identifying defect region pixels and the second term is the penalty for identifying clear wood pixels as a defects. Minimizing this loss function will allow the selection of the best parameter

value. A plot of the loss function will also provide an indication of the sensitivity of the parameter.

Because the region-growing method requires an accurate seed placement, a stiff penalty for generating seeds in clear wood areas should be in place. Thus, using a value of 0.02 for δ in the loss function would give a much higher penalty for “false alarms” compared to not identifying a defect region pixel as a seed. Since more defect region pixels will be identified later on in the pixel aggregation stage, there is not a critical need to identify all defect seeds at this point. A computer program to generate the loss function value for the seed generation, `loss_seeds.c`, is listed in Appendix A.

After the seeds are generated, the same loss function is used to measure the effectiveness (and sensitivity) of the stop values used to halt the pixel aggregation process. Because pixel aggregation is the final step, the penalty for missed defect pixels and clear wood pixels marked as defects should be about the same. Thus, at this step, a value of 0.5 for δ in the loss function would be appropriate. A computer program to generate the loss function value for the pixel aggregation, `loss_stops.c`, is listed in Appendix A.

3.3.4 Regression of image statistics to algorithm parameters for automatic determination of parameters

A manual method of setting the algorithm parameters such as the normal variation threshold limits individually for each frame would not be desirable since it would preclude the use of the defect detection algorithm in a production environment. Therefore, it would be desirable to determine whether there are characteristics of the images that would enable one to predict the best algorithm parameters for each image.

Khoros software will readily generate image statistics such as the mean of the gray level values, the standard deviation, the root mean square, the skewness, the kurtosis, the entropy and the contrast.

Before the image statistics can be obtained, it would be necessary to separate the foreground portion of the image, the log itself, from the background portion, the surrounding air, so that the image statistics would be calculated only on the relevant foreground. The moment-preserving thresholding method (Tsai, 1985) was used.

As seen in Figures 4.2, 4.4, 4.6, 4.8, and 4.10 in Chapter 4, the foreground regions were identified by a new binary image that had pixel values of zero (black) for the background and pixel values of 255 (white) for the foreground. This binary image was required by the *vstats* routine in Khoros because the statistics were to be calculated only upon the foreground (white) region of the image. The results from the *vstats* routine are given in Table B.2 in Appendix B. These results also contain the threshold value from the moment-preserving threshold and the standard deviation of the gray levels from the average-filtered image levels.

All of the image statistics in Table B.2 were used to determine if the normal-variation threshold limits could be predicted that would ignore normal variation of the deviation of the gray levels from the average background level and detect the unusual variation due to defects. Since the important prediction variables for the best threshold limit was not known *a priori*, the method used was to regress all of the available image statistics in each frame against the manually obtained normal-variation threshold limit. Then, the extraneous prediction variables were eliminated one by one using the backward elimination method (Neter *et al.*, 1996).

At each iteration of the backward elimination method, all remaining prediction variable regression parameters were evaluated by the F -test for $H_0: \beta_i = 0$. The F -test statistic is given by:

$$F_{1,n-p-1} = \frac{[SSR(full) - SSR(reduced)]/1}{MSE(full)}$$

where SSR is the sum of the squared errors due to the regression and MSE is the mean squared error. The reduced model refers to the model without the parameter in question. The regression variable with the smallest F value was eliminated from the model and the MSE for the reduced model was recalculated. If the MSE for the reduced model was smaller than the MSE of the full model, the reduced model became the new full model and another iteration was started. This process continued until the MSE could no longer be reduced. Rather than using the common method of applying a predetermined limit for the F -test statistic, the adoption of the goal of minimizing the MSE allowed the simultaneous consideration of reducing the number of predictor variable parameters while maximizing the predictive power of the regression equation.

A subset of the image statistics was found to be an excellent predictor of the proper normal-variation threshold limits for each spin-echo image. The final regression equation and its significance are given in the results in Tables 4.1 through 4.4 in Chapter 4.

3.3.5 Obtaining predicted algorithm parameters from regression

As mentioned in Section 2.6, to get an indication of how the regression model will perform, the selected data set to be used to evaluate prediction performance should be independent of the data set that was used to generate the regression relationship. When

the data set is small, the cross validation method is preferred because it utilizes the available data more efficiently (Breiman *et al.* 1984). When computational burden is not a concern, it would be preferable to use N-fold cross validation. However, the data set used in this study will not allow that. The data is obtained from MRI frames that are highly correlated with adjacent frames. The image measures from each frame that are used by the regression as predictor variables are also correlated with the same measures from adjacent frames. The N-fold cross validation method uses all but one data point for building the regression and the left-out data point for prediction. With the current data, the left-out data point would not be independent of the data points used to build the regression since there is correlation between adjacent points. Even for v-fold cross validation, where the data points are randomly assigned into v subsets, the prediction data set would not be independent of the fitting data set since points in both sets are adjacent to each other and therefore correlated.

Therefore, a modified cross validation procedure was developed to provide a prediction data set that was independent of the fitting data set. With most types of data autocorrelation, the correlation between data becomes smaller as the distance between the data points increases. The correlation of image measures between frames versus the distance between frames was calculated and plotted to find the distance where the largest portion of the correlation is gone. This distance can be used to provide a window. Now, in order to predict a point, the portion of the fitting data set that was within this distance to the prediction point is dropped out. The remaining fitted data set consists of points that are further away from the prediction point and are less correlated. This reduced set of fitting points is used to generate the regression equation for prediction the response at

the prediction point. This will achieve the effect of having the predicted data point (relatively) independent of the fitting data set. The process is repeated for each point to be predicted making it rather computationally burdensome. An example of the plot of the correlation of image measures versus the distance between images is given in Figure 4.29.

By using the program algorithm described in Section 3.2 and using the modified cross validation procedure to predict the parameters from the individual image characteristics, the defect regions were extracted from the spin-echo images. The results are given in Figures 4.114, 4.119, 4.124, 4.129, and 4.134 in Chapter 4.

3.4 Comparing the Spin-Echo Defect Detection Algorithm against Thresholding Methods

3.4.1 Overview

The goal of this section is to compare the results from the spin-echo defect detection algorithm against the common method of ordinary thresholding to detect defects. This test is important since ordinary thresholding is the common method used today and because ordinary thresholding would be much faster and simpler to implement.

To compare the two methods, the following steps are taken:

Section 3.4.2 Finding the equivalent number of defect pixels with thresholding

Section 3.4.3 Comparing defect results between the spin-echo defect detection algorithm and thresholding

Section 3.4.4 Seeking better threshold levels for the thresholding method

3.4.2 Finding equivalent number of defect pixels with thresholding

The Khoros routine, `vsubstit`, can be used to replace the pixels of a particular gray level value with another gray level value. Also, the Khoros routine, `vstats`, can be used to count the number of positive gray level pixels and negative gray level pixels in an image. Thus, combinations of these Khoros routines can be used to count the pixels at any gray level.

In fact, those Khoros routines were used to count the number of defect pixels of wet areas (high gray levels) and the number of defect pixels of dry areas (low gray levels) from the defect images produced by the spin-echo defect detection algorithm. That data was tabulated in Table B.4 in Appendix B. Then, for each original spin-echo image, ordinary thresholds were applied using the Khoros routine, `vtresh`, to obtain the same number of wet defect pixels and dry defect pixels. The thresholds were manually adjusted until the number of pixels matched the spin-echo defect images. Typical results are given in Figures 4.42 through 4.46 in Chapter 4.

3.4.3 Comparing defect results between the spin-echo defect detection algorithm and thresholding

Figures 4.42 through 4.46 in Chapter 4 allowed the visual comparison of the thresholding results when an attempt was made to capture as many defect pixels with thresholding as was obtained with the spin-echo defect detection algorithm. Visual comparison was adequate for comparison purposes. However, the number of undetected pixels and the number of false alarm pixels generated by the ordinary thresholding method were counted for each image and tabulated in Table B.4 in Appendix B.

3.4.4 Seeking better threshold levels for the thresholding method

The results from the comparison as described in Section 3.4.3 indicated that the ordinary thresholding method could not find as many defect pixels as the proposed method without generating a large number of false alarms. It was of interest to determine if the thresholding method could generate good results as any threshold level. The method was applied again but this time the goal of obtaining the same number of defect pixels was abandoned. The method was applied to a single image several times with different threshold levels. The results of that trial were given in Figures 4.47 through 4.51 in Chapter 4.

3.5 Development of Defect Detection Algorithm for Echo-planar Images

3.5.1 Overview

The echo-planar images were taken from the same log at the same locations as the spin-echo images. The spin-echo images were stored in 256 by 256 arrays. Because of memory limitations in the equipment at the time that the images were acquired, the echo-planar images were stored in pixel arrays that were only 128 pixels high and 256 pixels wide. Also, the echo-planar images were at a very low intensity.

3.5.2 Cropping and cleanup of echo-planar images

In order to increase the brightness and the contrast, the gray levels of each pixel were all multiplied by 2. The echo-planar image after the gray levels were increased for the same spin-echo image given in Figure 3.3 is shown in Figure 3.17. In order to work with the echo-planar images, the top and bottom edges of the image were cleaned up and the image size was increased to 256 by 256. An example of the echo-planar image after these processing steps is shown in Figure 3.18.

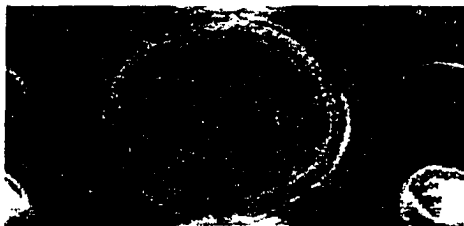


Figure 3. 17 Contrast Enhanced Echo-Planar Image (x2)

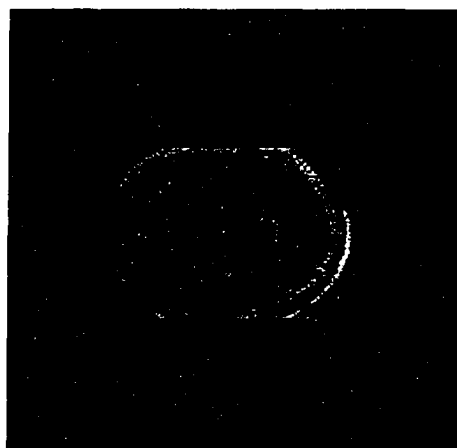


Figure 3. 18 Echo-Planar Image After Cropping and Padding

3.5.3 Registration of echo-planar images to spin-echo images

One final step remained before the echo-planar images could be compared to the spin-echo images. There is an inherent tendency in echo-planar images for warping in the y-direction. This is caused by an anisotropic resolution in the image domain due to the different weighting of T_2 decay in directions x and y. However, researchers are proposing new methods to partially alleviate this drawback (Iwaoka *et al.* 1984, Bendel 1985, Ahn *et al.* 1986).

To compare the echo-planar images directly with the spin-echo images, it was necessary to unwarp the echo-planar images so that the images would register, or line up with the spin-echo images. This was done manually one frame at a time using the Khoros routine *warpimage*. This routine allows one to interactively select any number of corresponding points on both images. Then the routine performs a least-squares fit regression on a second-order polynomial equation that will translate the locations of the first set of points to the locations of the second set of points. The resulting equation is

then used to warp, or translate, the first image to the shape of the second image. In industrial practice, since the main axis of distortion is in the y-direction, it would only be necessary to choose points on the perimeter of the log at fixed x-axis locations and compare the y-dimensions of these same points with the y-dimensions obtained from an optical profile scanner of the type that are currently in use (Lee *et al.* 1991, Maness and Donald 1994). Thus, with a program like Khoros warpimage, the image could be routinely transformed to match the physical dimensions. An example of the unwarping of the echo-planar image in Figure 3.18 to register with its respective spin-echo image is given in Figures 3.19 and Figure 3.20. Note that the unwarped image has slightly larger measurements in the y-directions. There was a tendency for the echo-planar images to be compressed in the y-dimension compared to the spin-echo images.

3.5.4 Separation of foreground and background

After the echo-planar images were cleaned up and registered with the spin-echo images, the spin-echo defect detection algorithm was first applied to the echo-planar images directly without any further preprocessing of the echo-planar images. The purpose of this step was to begin a search for the least amount of preprocessing of echo-planar images that is required to provide useful defect information. However, when the moment-preserving thresholding step to separate the log foreground from the air background was applied, the severe amount of noise in the image generated a rough border between the log and the background. The border itself also encroached upon the interior of the log. An example of an image that was separated into foreground and background using the same approach for spin-echo images is given in Figure 3.21. It became necessary to adjust the threshold percentile that came from the moment-

preserving threshold algorithm by reducing the percentile 3%. A program, `echo_thresh.c`, was written in ANSI C to accomplish that and it is listed in Appendix A.

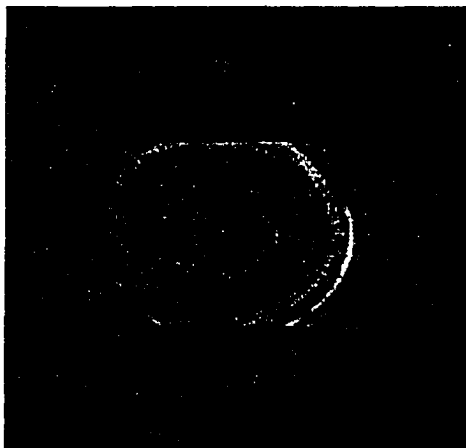


Figure 3. 19 Warped Echo-Planar Image That Registers With Spin-Echo Image in Figure 3. 20

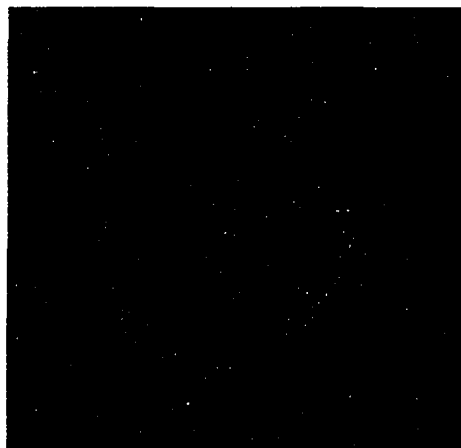


Figure 3. 20 Spin-Echo Image Taken at Same Location as Echo-Planar Image in Figure 3. 19

To clean up the roughness of border, it was necessary to apply a morphological smoothing step. A morphological element in the shape of a ball was used because a natural border can have an infinite variety of orientations. An example of the thresholded image after the extra steps of adjusting the moment-preserving threshold percentile and applying morphological smoothing to the result is given in Figure 3.22.

3.5.5 Application of spin-echo defect detection algorithm to echo-planar images

At this point, the programming was completed to allow the defect regions to be clearly identified. The echo-planar images represent a different population from the spin-echo images because the noise in the echo-planar images is much higher and the defect regions did not stand out as much as they did in the spin-echo images. Therefore, the same regressions used by the spin-echo detection system could not be used again.

However, the same approach (described in Section 3.3) used for the spin-echo images was applied to the echo-planar images. The parameters for each echo-planar image are given in Table B.5 in Appendix B.

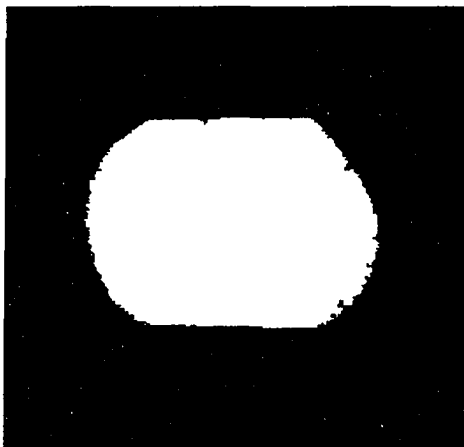


Figure 3. 21 Foreground of Echo-Planar Image From Moment-Preserving Threshold

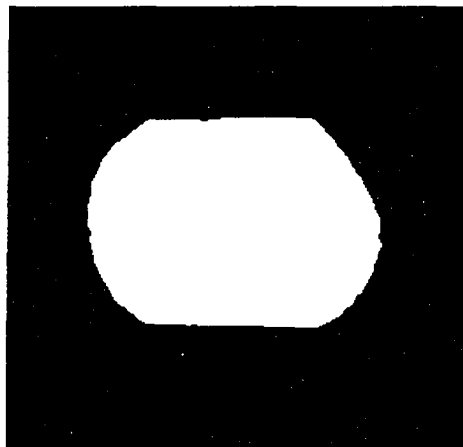


Figure 3. 22 Foreground of Echo-Planar Image With Additional Morphological Smoothing Step

All of the image statistics in Table B.6 in Appendix B were used to determine if the parameters could be predicted. The extraneous prediction variables were eliminated one by one using the backward elimination method (Neter *et al.*, 1996) until the *MSE* could no longer be reduced.

A subset of the image statistics was found to be an excellent predictor of the proper parameters for the echo-planar images. The final regression equation and its significance are given in Tables 4.5 through 4.8 in Chapter 4.

3.5.6 Finding pith defect in echo-planar images

Because of the extra noise in the echo-planar images compared to the spin-echo images, the normal-variation threshold limits were forced to be higher to avoid

generating an excessive number of false alarms. The results being that often the pith area was not detected. Given that the pith is always present in any cross section and that the pith is located close to the geographic centroid of the cross section, an extra processing step could be added to the echo-planar algorithm to detect this defect.

The centroid of the foreground was calculated using the thresholded image and the following equations:

$$\begin{aligned} \text{x - coordinate of centroid} &= \frac{\sum_{\text{foreground pixels}} \text{x - coordinate of pixel}}{\text{number of foreground pixels}} \\ \text{y - coordinate of centroid} &= \frac{\sum_{\text{foreground pixels}} \text{y - coordinate of pixel}}{\text{number of foreground pixels}} \end{aligned}$$

Next the defect detection program was amended to search for defect regions in a 20-pixel square area centered on the foreground centroid after the normal algorithm was allowed to grow defect areas. If defects were found in the center of the foreground, then the program assumed that the pith area had been detected. If no defect region was detected in the centroid area, then the defect detection program applied more severe normal-variation threshold limits to the center area to obtain the pith defect seeds. The program then grows the defect regions around the seeds in the center area as before but using a different set of stopping limits.

The best pith defect threshold limits and pith region stopping limits for each echo-planar image were different. So, as with the other algorithm parameters, all of the image statistics in Table B.6 were used to determine if the pith region threshold limit and pith region-growing stopping limit could be predicted from the image characteristics themselves.

Using the backward elimination method, a subset of the image statistics was found to be an excellent predictor of the proper pith region threshold limit and pith region-growing stopping limit for each spin-echo image. The final regression equation and its significance are given in Tables 4.9 and 4.10 in Chapter 4.

Again, using the modified cross validation method, a distance between frames was determined that would allow the major portion of correlation between the fitting data and the prediction data set to be removed. Using the predicted parameters for each image in the algorithm, the defect regions were extracted from the echo-planar images. The results are given in Figures 4.113, 4.118, 4.123, 4.128, and 4.133 in Chapter 4.

3.6 Finding the Best Type of General Image Enhancement Method for Echo-Planar Images

3.6.1 Overview

The number of image enhancement methods is large and is growing as evidenced by the review of the image enhancement methods in Chapter 2, Section 2.5. The Khoros image processing software package contains utility programs that implement the most common image enhancement and noise elimination routines. These include spatial filters such as the median, speckle removal, and 2-D convolution which allows a number of smoothing filters such as an averaging window. These three methods of image enhancement, the median filter, speckle removal and averaging filter represent three broad areas of general image enhancement methods.

It was of interest to determine which method would improve the appearance of the echo-planar images. There are two reasons for this. Improving the appearance of the rough-looking echo-planar image could possibly improve the acceptability of the method

by industry. The second reason is that enhancement of the image could possibly improve the effectiveness of the defect detection algorithm. Each of the three main general image-enhancement routines was tested on the echo-planar images to determine which routine performed best.

The median filter was applied over the foreground using the Khoros routine, `vhmed`, and used a 3 x 3 window for minimal disturbance of major features. The speckle removal routine from Khoros, `vspeckle`, uses the Crimmins routine. The average filter was applied using the Khoros 2-D convolution routine, `vconvolve`, and it used the 3 x 3 window again for minimal disturbance of major features.

3.6.2 Measure of the best image enhancement

In order to compare the success of the three methods, each of the echo-planar images were processed by the three methods. The defect image was not generated at this point since the enhancement of the echo-planar image before defect detection is the interest of this section. The signal-noise ratio of each of the processed images was computed using the original spin-echo image as the base of comparison. Before computing the signal-noise ratio using two images of the same object, there are two precautions that must be taken. A physical registration between the two images must be performed. This was already done as described in Section 3.5.3. Also, a “gray-level” registration must be performed. That is, the gray levels of both images should be “normalized” before we start looking for noise in one of the images. If there were two images whose only difference was that one image’s gray levels were a multiple of the other image’s gray levels, then all of the pixels would appear to be noise. Thus, the pixels need to be standardized before the comparisons are made. Otherwise, the signal-

noise ratio would be comparing the illumination response curve of two vision systems rather than comparing the noise-removal effectiveness of an image enhancement method. A sure way of standardizing the pixel gray levels for each image is to employ histogram equalization. Recall that the histogram of a digital image in L gray levels in the range $[0, L-1]$ is a discrete function given by $p(r_k) = n_k/n$, where r_k is the k th gray level, n_k is the number of pixels in the image with the k th gray level and n is the total number of pixels in the image. Histogram equalization refers to a monotonic remapping of the pixel gray levels so that the histogram of the image is flat and the gray levels follow a uniform distribution (i.e., $p(r_k) = 1/L$). A computer program to equalize the histograms of the images, `fore_hist_eq.c`, was written in ANSI C and is listed in Appendix A.

After both the spin-echo and the processed echo-planar images were equalized, the signal-noise ratio was computed using the following equation:

$$SNR = \log_{10} \frac{\text{noise free power}}{\text{noise power}} = \log_{10} \frac{\sum_{x=0}^n \sum_{y=0}^m f(x,y)^2}{\sum_{x=0}^n \sum_{y=0}^m [g(x,y) - f(x,y)]^2}$$

where $f(x,y)$ is the gray level at pixel location (x,y) in the “good” spin-echo image of size $n \times m$ pixels and $g(x,y)$ is the gray level at pixel location (x,y) in the echo-planar image of size $n \times m$. Thus, the processing method used on the echo-planar image that would come closest to reproducing the gray levels of the spin-echo image would produce the highest signal-noise ratio. The SNR results are given in Tables B.8 in Appendix B.

3.6.3 Experimental design for testing differences in image enhancement

In order to determine which image enhancement method, if any, was superior to the rest, the statistical experimental design was first determined. The three image enhancement methods were considered as the treatments. Since these three treatments were the only ones of interest, they were considered as fixed treatments. The echo-planar images were considered to be the experimental units. Because it was known that each echo-planar image was unique, to take the variability of the images out of consideration during the comparison, the experimental design should test each method on each of the images. Thus, the randomized complete block design was used. That meant that the images were also considered as blocks. Since the images were but one set from the population of black oak trees, the block effect was considered a random effect. The statistical model for this design is:

$$y_{ij} = \mu + \tau_i + \beta_j + \varepsilon_{ij} \begin{cases} i = 1, 2, \dots, a \\ j = 1, 2, \dots, b \end{cases}$$

There are a treatments and b blocks and y_{ij} is the experimental result from the i^{th} treatment and the j^{th} block, τ_i is the effect of the i^{th} treatment, β_j is the effect of the j^{th} block, and ε_{ij} is the random error term which is assumed normal and independently distributed with a 0 mean and a variance of σ^2 . It can be shown (Montgomery, 1997) that

$$\begin{aligned} E(MS_{\text{Blocks}}) &= \sigma^2 + a\sigma_\beta^2 b \sum_{i=1}^a \tau_i^2 \\ E(MS_E) &= \sigma^2 \\ E(MS_{\text{Blocks}}) &= \sigma^2 + a\sigma_\beta^2 \\ E(MS_E) &= \sigma^2 \end{aligned}$$

Thus, if the treatment effects were equal meaning that the individual treatment effects, τ_i , were all 0, then the test statistic of equality would be

$$F_0 = \frac{MS_{\text{Treatments}}}{MS_E}$$

which is distributed as $F_{a-1, (a-1)(b-1)}$ if the null hypothesis is true. The critical region is the upper tail of the F distribution, and we would reject H_0 if $F_0 > F_{0.95, a-1, (a-1)(b-1)}$

The analysis of variance table for the randomized complete block design is given in Table 3.1.

Because there was found to be a significant difference between the SNR of the three image enhancement methods, Tukey's test was used to determine which method was significantly different from the others. This test is based upon the studentized range statistic and the standard error of the treatment mean. The standard error of a treatment mean is calculated as $S\bar{y}_i = \sqrt{\frac{MS_E}{b}}$. The studentized range statistic is denoted by $q_\alpha(a, f)$ where α is the significance level, a is the number of treatments and the degree of freedom of the error, f , is $(a-1)(b-1)$. Tukey's test declares two means significantly different if the absolute value of their sample differences exceeds $T_\alpha = q_\alpha(a, f)S\bar{y}_i$. (Montgomery, 1997).

The results of these comparisons between the unprocessed method and the three general image enhancement methods on echo-planar images are given in Table 4.11 in Chapter 4.

In addition, the three general enhancement methods applied to the echo-planar images were judged visually to determine which method produced “good” results. A discussion of that evaluation is given in Chapter 4, Section 4.7.

Table 3. 1 Analysis of Variance for a Randomized Complete Block Design

Source of Variation	Sum of Squares	Degrees of Freedom	Mean Square	F_0
Treatments	$\sum \frac{y_{i.}^2}{b} - \frac{y_{..}^2}{N}$	$a - 1$	$\frac{SS_{Treatments}}{a - 1}$	$\frac{MS_{Treatments}}{MS_E}$
Blocks	$\sum \frac{y_{.j}^2}{a} - \frac{y_{..}^2}{N}$	$b - 1$	$\frac{SS_{Blocks}}{b - 1}$	
Error	$SS_E = SS_T - SS_{TRT} - SS_{BLK}$	$(a - 1)(b - 1)$	$\frac{SS_E}{(a - 1)(b - 1)}$	
Total	$\sum \sum y_{ij}^2 - \frac{y_{..}^2}{N}$	$N - 1$		

3.7 Use adjacent slices to eliminate noise in echo-planar images

3.7.1 Overview

The adjacent-slice smoothing method was specifically designed to be used on the noisy echo-planar image. The trend in recent years in image smoothing has been toward adaptive filters where the goal is to smooth homogeneous areas and avoid smoothing across the image structures (Rank and Unbehaven 1992, Kundu and Zhou 1992, Itagaki 1993, Hussian and Reed 1994, Chen *et al.* 1994, Hanke *et al.* 1994, Hwang and Haddad 1995, Karaman *et al.* 1995, Li and Ramsingh 1995). Soltanian-Zadeh *et al.* (1995) note that the noise in an MRI scene sequence is characterized by additional zero-mean white Gaussian noise field that is uncorrelated between different frames. Because of the nature of trees, the image structure between two adjacent cross-sectional MRI slices of a log are very similar, especially at 10 mm apart.

The two proposed methods of smoothing along the z-axis (between like pixels in adjacent MRI slices) accomplished the dual goals of smoothing along homogeneous directions while avoiding the smoothing across image structures which are also the goals of adaptive smoothing filters. Since the smoothing takes place without the condition checking used by adaptive filters, the computational burden of z-axis smoothing is minimized compared to adaptive filters. Gaussian smoothing or median smoothing along the z-axis are designed to eliminate many of the random noise values.

3.7.2 Gaussian smoothing between frames

Gaussian smoothing refers to a specific method of computing a weighted average of the neighborhood gray levels. The height of the Gaussian curve is used to determine the relative weights of the gray levels in the averaging window. The idea is to give the pixels that are closest to the window center more weight in determining the neighborhood average. The only parameter required is the standard deviation of the Gaussian distribution, σ . Since 99.73% of the Gaussian curve area falls within $\pm 3\sigma$, the effective width of the window used in Gaussian smoothing is 6σ .

Since the best Gaussian smoothing parameter, σ , was not known *a priori*, three values of σ were tried: $\sigma = 5$ mm, $\sigma = 10$ mm, and $\sigma = 15$ mm. Recall that the slices were 10 mm apart. The computation of the weights used in the Gaussian smoothing is given in Tables 3.2 through 3.4.

The Gaussian smoothing process was implemented on the echo-planar images using the Khoros routine, `vmul`, to factor the images using the weights given in Tables 3.2 through 3.4. The Khoros routine, `vadd`, was used to combine the factored images into

one. After the resulting images went through the histogram equalization process, the SNR of each resulting image was computed as noted before by using the spin-echo images as the base of comparison. The results are given in Table B.9 in Appendix B.

Table 3. 2 Gaussian Smoothing Weights for $\sigma = 5$ mm

Slice Location in Window	Distance from Window Center	Z-value (Distance/ σ)	$f(z) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}z^2}$	Relative Weight
-1 (Previous)	10 mm	2	0.0540	0.1065
0 (Center)	0 mm	0	0.3989	0.7870
+1 (Subsequent)	10 mm	2	0.0540	0.1065

Table 3. 3 Gaussian Smoothing Weights for $\sigma = 10$ mm

Slice Location in Window	Distance from Window Center	Z-value (Distance/ σ)	$f(z) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}z^2}$	Relative Weight
-2 (Previous)	20 mm	2	0.0540	0.0545
-1 (Previous)	10 mm	1	0.2420	0.2442
0 (Center)	0 mm	0	0.3989	0.4026
+1 (Subsequent)	10 mm	1	0.2420	0.2442
+2 (Subsequent)	20 mm	2	0.0540	0.0545

Table 3. 4 Gaussian Smoothing Weights for $\sigma = 15$ mm

Slice Location in Window	Distance from Window Center	Z-value (Distance/ σ)	$f(z) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}z^2}$	Relative Weight
-3 (Previous)	30 mm	2	0.0540	0.0366
-2 (Previous)	20 mm	1.33	0.1640	0.1113
-1 (Previous)	10 mm	0.67	0.3194	0.2167
0 (Center)	0 mm	0	0.3989	0.2708
+1 (Subsequent)	10 mm	0.67	0.3194	0.2167
+2 (Subsequent)	20 mm	1.33	0.1640	0.1113
+3 (Subsequent)	30 mm	2	0.0540	0.0366

3.7.3 Median smoothing between frames

The other method of z-axis smoothing involved applying the median filter across slices. It was noted that the noise is uncorrelated between different frames in an MRI

sequence (Soltanian-Zadeh *et al.* 1995). Therefore, the median filter would only need a small window of size 3 since the likelihood of noise occurring more than once in the same pixel location in three separate frames would be very small. The smallest window size of 3 would cause the minimum disturbance of pixels that were not noise. Because of the uniqueness of this approach, it was necessary to write a routine to perform this z-axis median filtering. The routine, `zmed3.c`, was written in ANSI C and is listed in Appendix A.

As with the Gaussian-smoothed images, the *SNR* of each resulting z-axis median smoothed image was computed as noted before by using the spin-echo images as the base of comparison. The results are given in Table B.9 in Appendix B.

3.7.4 Comparing adjacent slice smoothing methods

The recommended method for z-axis smoothing was determined to be the method that gave the best (highest) *SNR*. As with the comparison of the general image enhancement methods, the statistical experimental design was first determined. The five image enhancement methods (unprocessed, $\sigma = 5$ mm, $\sigma = 10$ mm, $\sigma = 15$ mm, and z-axis median) were considered as fixed treatments. The randomized complete block design was used to remove the variability of the slices from the experimental error. Thus, the images were again considered as random effect blocks.

The analysis of variance table for the randomized complete block design is given in Table 3.1. Because there was found to be a significant difference between the *SNR* of the five image enhancement methods, Tukey's test was used to determine which method was significantly different from the others.

The results of these comparisons between the unprocessed method and the four proposed image enhancement methods on echo-planar images are given in Table 4.12 in Chapter 4.

In addition, the four proposed enhancement methods applied to the echo-planar images were judged visually to determine which method produced “good” results. A discussion of that evaluation is given in Chapter 4, Section 4.8.

3.8 Comparing Defect Detection Capability of Processed Echo-Planar Images

3.8.1 Overview

At this point, the best image enhancement for the echo-planar image has been explored from the standpoint of appearance. It is more important to determine which image enhancement method improves the echo-planar images from the standpoint of detecting defect regions correctly. It is this comparison of the adjacent-slice-smoothing methods to the best of the general-purpose image enhancement methods that will ultimately determine the method most suitable for use with echo-planar images.

3.8.2 Generating defect images from pre-processed echo-planar images

As with the spin-echo and unprocessed echo-planar images, the same process described in Section 3.3 was applied to the pre-processed echo-planar images. This was done so that all comparisons across methods were made with predicted images, not manually fitted images.

3.8.3 Measures of best defect detection

The clear wood areas in defect images are denoted by white (gray level = 255) pixels. Defect areas are denoted by gray (gray level = 128) pixels and the background is

denoted by black (gray level = 0) pixels. The defect data from the spin-echo images was used as a base to determine whether the echo-planar images were correct.

There are five measures that were used to judge the defect detection ability of the final four pre-processing methods (which used manually-set parameters for each frame):

- (1) *SNR* of defect images,
- (2) number of undetected defect pixels,
- (3) number of undetected defect regions,
- (4) number of false alarm defect pixels, and
- (5) number of false alarm defect regions.

When using *SNR* on defect images, whether a pixel was classified as an undetected defect or as a false alarm, it would contribute $(255-128)^2$ toward the noise power. Thus, the *SNR* would give equal consideration to undetected defects and false alarms. That is why the other measures were introduced so that different aspects of the defect detection ability could be judged.

The *SNR* calculation was aided by using a combination of Khoros routines for the subtraction and multiplication steps of the image pixels. The number of undetected pixels and the number of false alarm pixels were also calculated using a Khoros routine that subtracts one image from another and then the Khoros routine, *vstats*, to determine how many positive pixels and how many negative pixels were found in the difference image. The number of undetected defect regions and the number of false alarm regions were manually counted. A spin-echo defect image region was counted as detected by the echo-planar defect region if there was at least some pixels of the region that were detected. If no pixels at all showed up, then the region was counted as undetected.

Likewise, even if a small region appeared on the echo-planar defect image that did not appear on the spin-echo image, that region was counted as a false alarm region.

3.8.4 Comparing defect images

As with the comparison of the general image enhancement methods, the statistical experimental design was first determined. The four image enhancement methods (no preprocessing, z-axis Gaussian smoothing with $\sigma = 5$ mm, z-axis median smoothing, and xy-median smoothing) were considered as fixed treatments. The randomized complete block design was used to remove the variability of the slices from the experimental error. Thus, the images were again considered as random effect blocks.

The analysis of variance table for the randomized complete block design is given in Table 3.1. Because there was found to be a significant difference between the *SNR* of the four image enhancement methods, Tukey's test was used to determine which method was significantly different from the others.

The results of the *SNR* comparisons between the four image enhancement methods on echo-planar images are given in Table 4.19 in Chapter 4. The results of the undetected defect pixel comparisons are given in Table 4.20 in Chapter 4. The results of the undetected defect region comparisons are given in Table 4.21 in Chapter 4. The results of the false alarm pixel comparisons are given in Table 4.22 in Chapter 4 and the results of the false alarm region comparisons are given in Table 4.23 in Chapter 4.

In addition, the four enhancement methods applied to the echo-planar images were judged visually against the spin-echo defect images to determine which method

produced “good” results. A discussion of that evaluation is given in Chapter 4, Section 4.10.

3.9 Determine Defect Detection Losses Between Spin-Echo and Echo-Planar Images

Based upon comparisons of the 55 slices, a general indication of the defect detection differences between the spin-echo and the echo-planar processed by the z-axis median filter was found. The average numbers of undetected defect pixels and regions and the number of false alarm pixels and regions on the defect images of the z-axis median enhanced echo-planar images were calculated along with 95% confidence limits on the averages using the equation:

$$\bar{x} - t_{\alpha/2, n-1} S/\sqrt{n} \leq \mu \leq \bar{x} + t_{\alpha/2, n-1} S/\sqrt{n}$$

A summary of those averages is given in Table 4.24 in Chapter 4.

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Overview

The detailed results for each particular image are found in the tables in Appendix B. Summaries of the results and statistical analyses are given in this chapter. The results given in this chapter are listed in the same order as the chronology of the methodology given in Chapter 3.

4.2 Results of Spin-Echo Defect Detection Algorithm

4.2.1 Separation of foreground from background

The application of the modified moment-preserving thresholding to separate the log foreground from the surrounding air background is demonstrated in Figures 4.1 through 4.10. The figures are grouped in pairs to present the original image followed by the respective binary image which maps the foreground and background.



Figure 4. 1 Spin-Echo Frame # 15

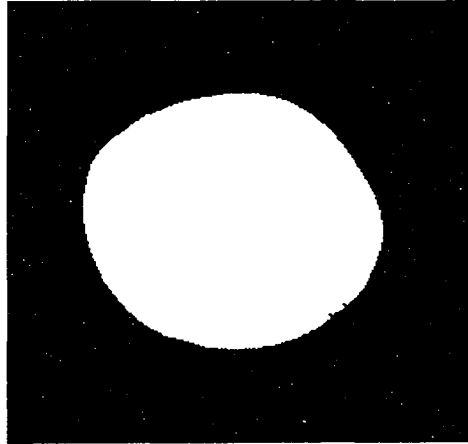


Figure 4. 2 Mask Image for Frame #15

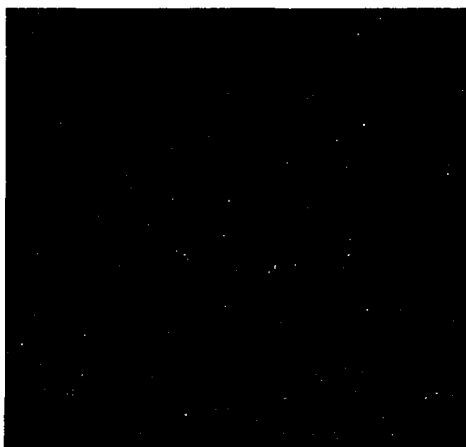


Figure 4. 3 Spin-Echo Frame # 20

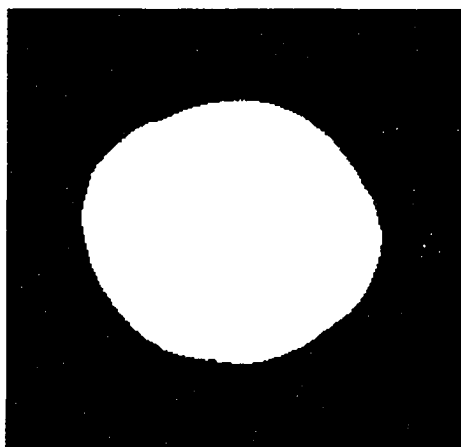


Figure 4. 4 Mask for Frame # 20



Figure 4. 5 Spin-Echo Frame # 25

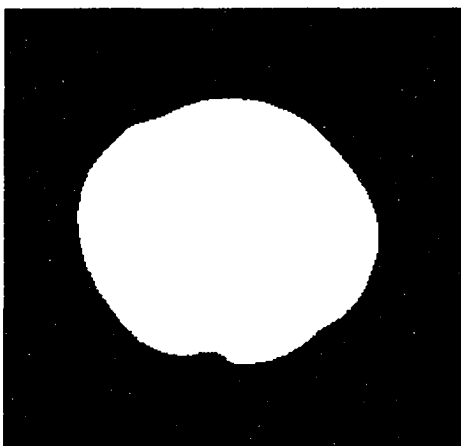


Figure 4. 6 Mask for Frame # 25



Figure 4. 7 Spin-Echo Frame # 30

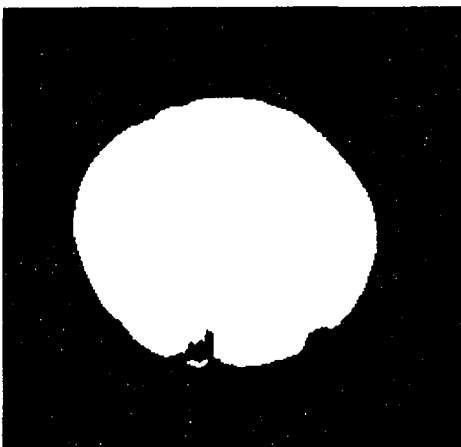


Figure 4. 8 Mask for Frame # 30



Figure 4. 9 Spin-Echo Frame # 50

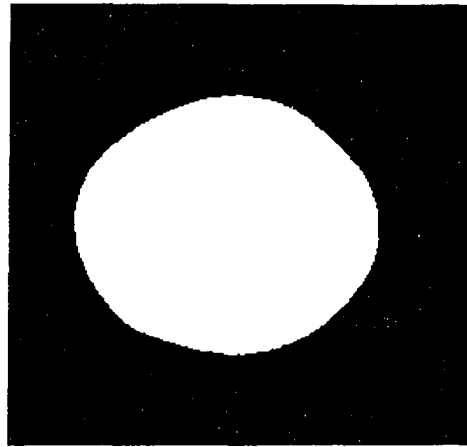


Figure 4. 10 Mask for Frame # 50

These thresholded images were used to define the foreground for the rest of the image processing steps. After the images were passed through an average filter, the result was subtracted from the original image to obtain the pure deviations in gray levels.

4.2.2 Image results of spin-echo defect detection algorithm

Next, the difference image, which contained positive and negative values, was passed through a positive threshold to capture unusually high (very moist) areas. Also, the image was passed through a negative threshold to capture unusually low (very dry) areas. These unusual areas are always associated with defect areas and will serve as seeds for the defect regions. Some examples of these seeds are given in Figures 4.11, 4.13, 4.15, 4.17 and 4.19. The defect regions were grown from these seeds using pixel aggregation based upon gray-level deviation as described in Section 3.2.5 and stopped when the gray levels deviations approached the deviations of surrounding clear wood. The respective defect images for the previously given seed images are given in Figures 4.12, 4.14, 4.16, 4.18 and 4.20.

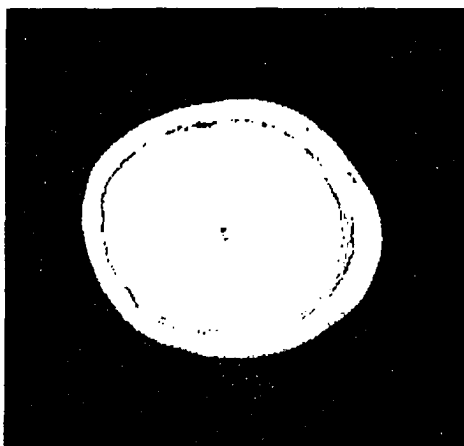


Figure 4.11 Seeds for Frame # 15

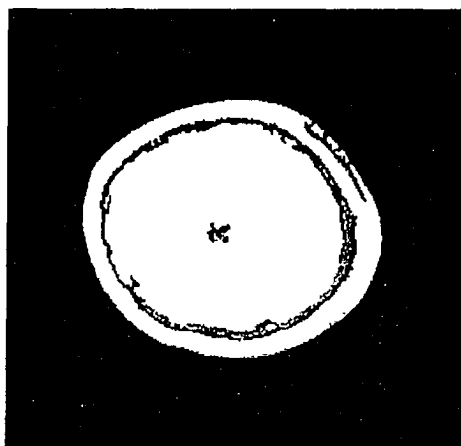


Figure 4.12 Defects for Frame # 15

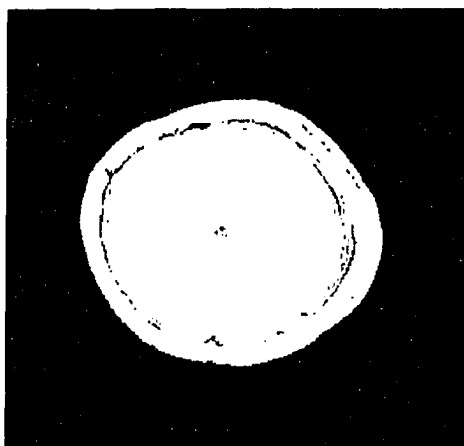


Figure 4.13 Seeds for Frame # 20

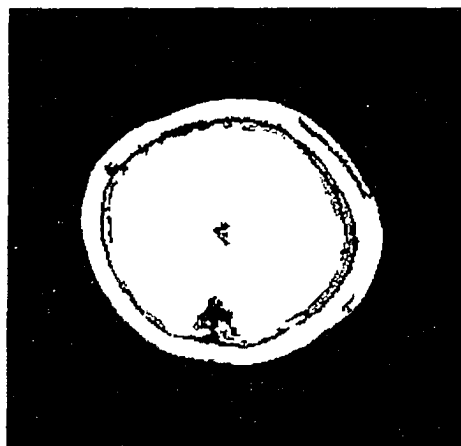


Figure 4.14 Seeds for Frame # 20

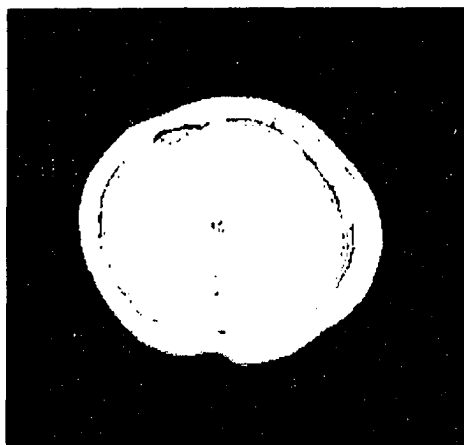


Figure 4.15 Seeds for Frame # 25

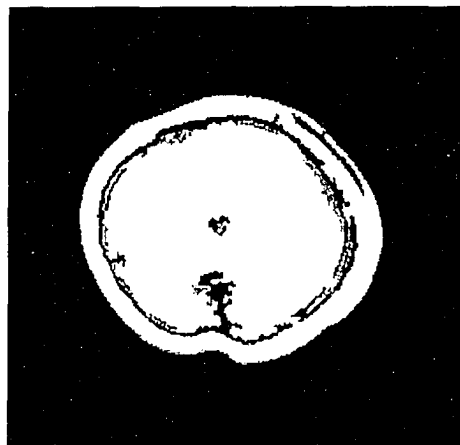


Figure 4.16 Defects for Frame # 25

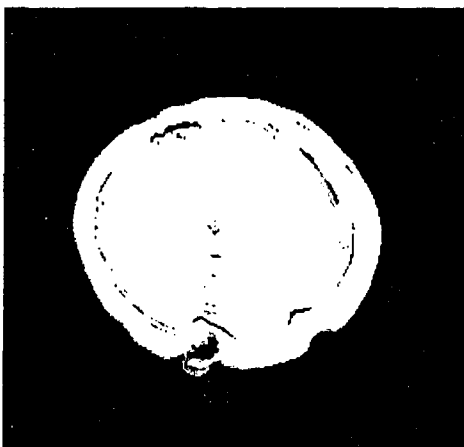


Figure 4. 17 Seeds for Frame # 30

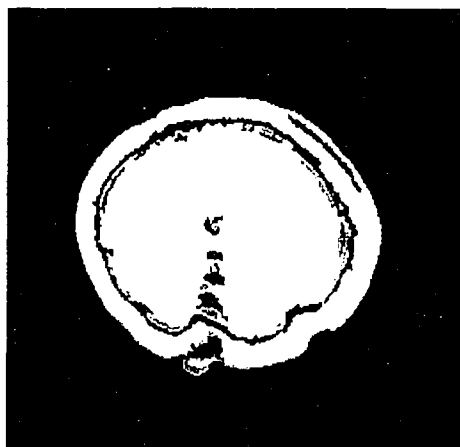


Figure 4. 18 Defects for Frame # 30

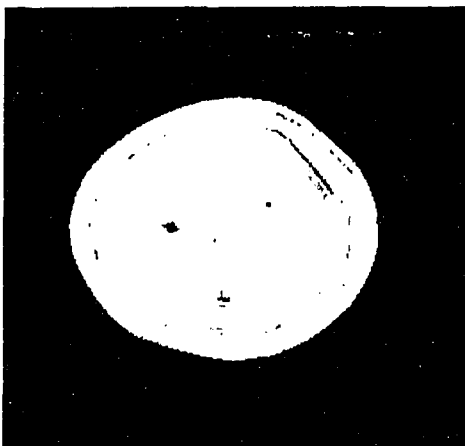


Figure 4. 19 Seeds for Frame # 50



Figure 4. 20 Defects for Frame # 50

Comparison of the defect images in Figures 4.12, 4.14, 4.16, 4.18 and 4.20 with the respective original spin-echo images found in Figures 4.1, 4.3, 4.5, 4.7 and 4.9 indicate that the spin-echo defect detection algorithm with the proper parameters produces excellent results. Again, the black areas indicate the air background, the white areas indicate clear wood and the gray areas indicate defect regions. The dark gray areas indicate defect regions associated with unusually dry wood such as knots. The light gray areas indicate defect regions associated with unusually wet areas such as rot.

4.3 Automatic Determination of Parameters for Spin-Echo Defect Detection Algorithm

In automatically determining the parameters to use, the first step was to find the appropriate window size for the smoothing operation. As explained in Section 3.3.2, the window size that maximized the skewness of the residuals was selected. The results for each spin-echo frame are shown in Figures 4.21 through 4.26.

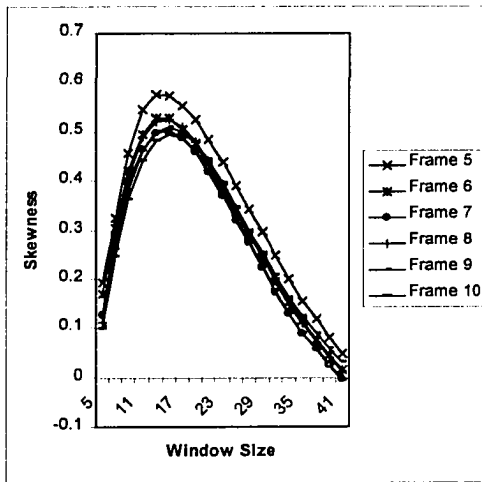


Figure 4. 21 Skewness for Frames 5-10

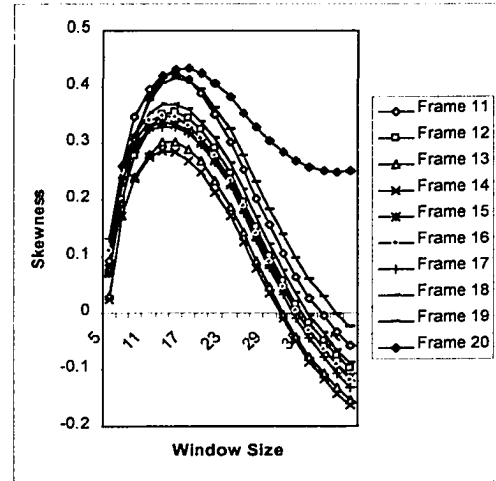


Figure 4. 22 Skewness for Frames 11-20

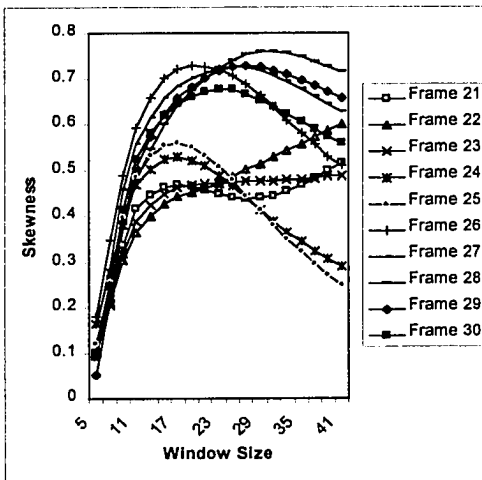


Figure 4. 23 Skewness for Frames 21-30

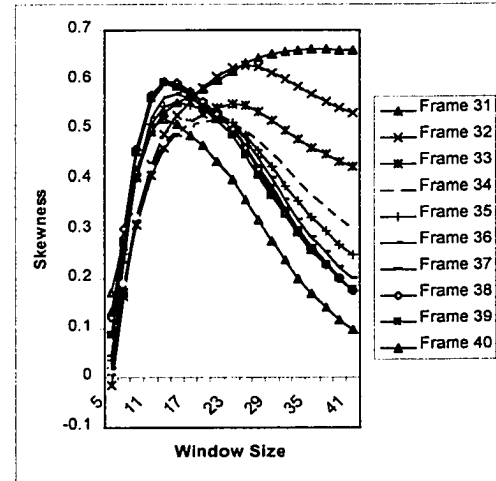


Figure 4. 24 Skewness for Frames 31-40

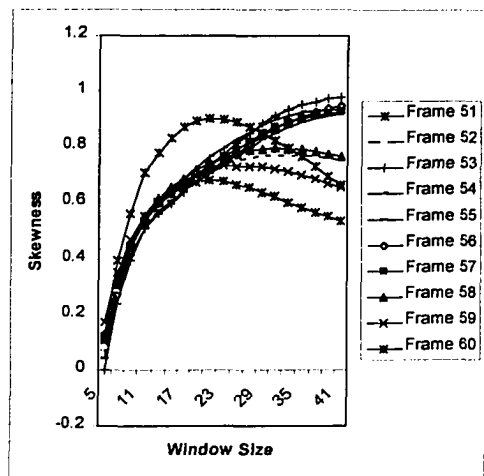
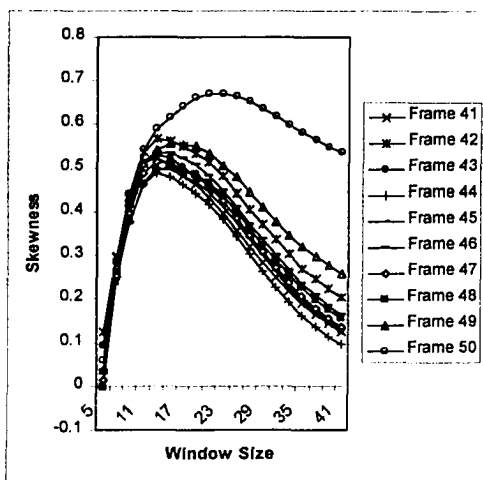


Figure 4. 25 Skewness for Frames 41-50 Figure 4. 26 Skewness for Frames 51-60

Using the appropriate window size for each frame and using the previously manually-obtained spin-echo defect images as the goal, the loss function as described in Section 3.3.3 was used to determine the optimum parameters to use on each of the spin-echo images. The manually-obtained parameters from the loss function that were used for the regression are given in Table B.2 in Appendix B. A couple of the loss-function profiles for the parameters are given in Figures 4.27 and 4.28. These profiles also give an indication of the sensitivity of the parameter.

The image statistics for the spin-echo images are given in Table B.3 in Appendix B. In Table B.3, the threshold is defined as the gray level obtained from the moment-preserving thresholding method that was used to separate the foreground from the background. The variance in Table B.3 is defined as the variance of the original-image pixel gray levels from the smoothed-image pixel gray levels (or the variance of the difference image). The mean, of course, is the mean gray level of the foreground pixels. Std. dev. in Table B.3 is the standard deviation of the gray levels of the foreground

pixels. RMS in Table B.3 is the root-mean-square of the gray levels of the foreground pixels and is calculated by squaring the gray levels of every pixel in the foreground. The mean value of the squares is calculated and the RMS is the square root of this mean value.

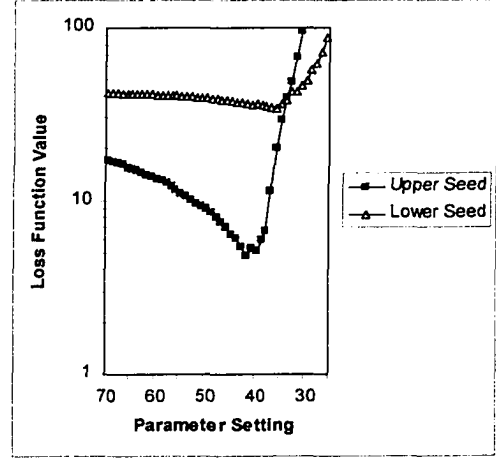
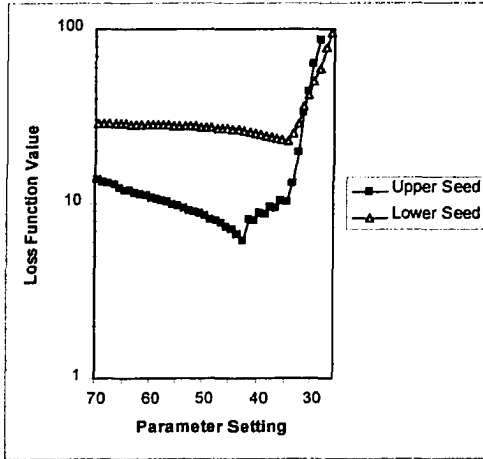


Figure 4. 27 Loss Function for Frame 10 Figure 4. 28 Loss Function for Frame 30

Skewness in Table B.3 is a measure of the tendency of the deviations from the mean to be larger in one direction than in the other, or, in other words, the asymmetry of a distribution about its mean. A positive skew value signifies a distribution whose tail extends out more in the positive direction, and a negative skew value signifies a distribution whose tail extends out more in the negative direction. The Khoros vstats routine computes the sample skewness as $\frac{1}{S^3 N} \sum_{i=0}^{N-1} (x_i - \bar{x})^3$ where N is the number of pixels, S is sample standard deviation and x_i is the gray level of the i^{th} pixel.

Kurtosis characterizes the relative peakedness or flatness of a distribution compared with the Gaussian distribution. Positive kurtosis indicates a relatively peaked

distribution. Negative kurtosis indicates a relatively flat distribution. The Khoros vstats

routine computes the sample kurtosis as $\frac{1}{S^4 N} \sum_{i=0}^{N-1} (x_i - \bar{x})^4 - 3$.

Entropy in Table B.3 is a measure of uncertainty in an information source, or said another way, the average number of choices a message source has in producing

information. Entropy is calculated in the Khoros vstats routine by $-\sum_{k=0}^{L-1} \frac{n_k}{N} \log_2 \left(\frac{n_k}{N} \right)$

where L is the number of gray levels, k is the gray level value, n_k is the number of pixels with gray level k and N is the total number of pixels.

Contrast in Table B.3 is a measure of the differences in the intensity (gray levels)

in an image and is calculated by the Khoros vstats routine by $\sum_{k=0}^{L-1} k^2 \frac{n_k}{N}$ where L is the

number of gray levels, k is the gray level value, n_k is the number of pixels with gray level k and N is the total number of pixels.

To find the regression equations that could predict the parameters, all the parameters were regressed against the set of image statistics. Backward elimination was used to find the set of image statistics that minimized the *MSE* for the prediction equation.

In order to obtain the prediction mean squared error, the correlation between the image statistics in adjacent frames was plotted against the distance between images (frames). The plot is given in Figure 4.29. The correlation between image statistics drops quickly as the distance increases to 7 frames. Beyond 7 frames, the correlation tends to level out. For that reason, no data was used in fitting the regression equation for a

distance of 7 frames on either side of a predicted data point. Except at the ends, each of the data points was predicted from a regression equation that used 55-15 or 40 of the points that were furthestmost away. The sum of the squared differences of the predictions and the actual values were calculated for comparison to the regression *MSE*.

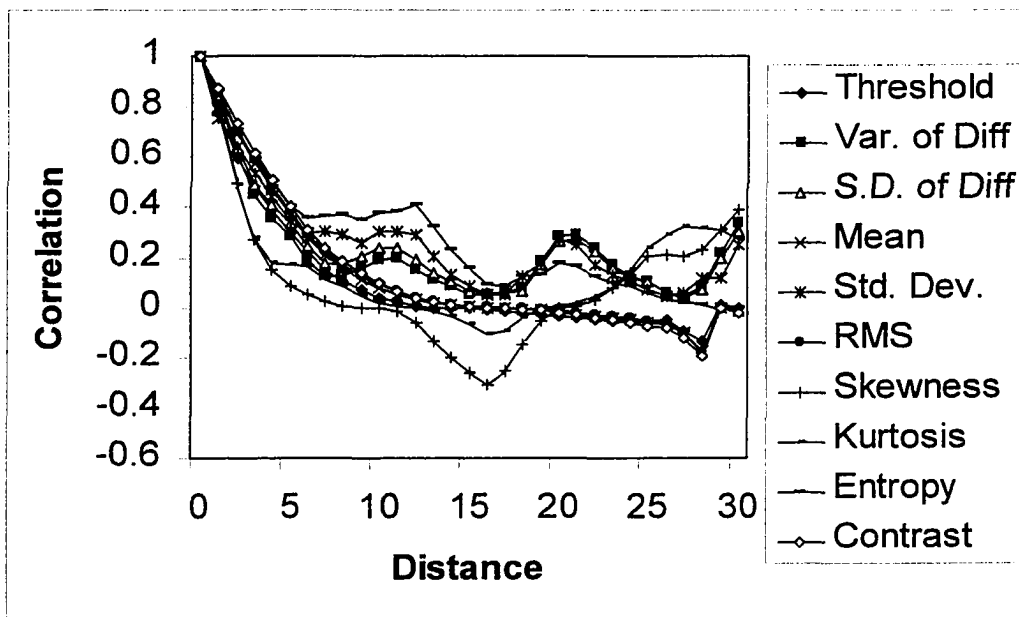


Figure 4. 29 Correlation of Image Statistics Between Frames

The results of the regression for each parameter are given in Tables 4.1 through 4.4.

Table 4. 1 ANOVA of Regression of Spin-Echo Upper Normal Variation Limit

<i>Regression Statistics</i>	
Multiple R	0.86121
R Square	0.741683
Adjusted R Square	0.696758
Standard Error	2.976192
Observations	55

table continued on next page

ANOVA				
	<i>df</i>	<i>SS</i>	<i>MS</i>	<i>F</i>
Regression	8	1169.891	146.2363	16.50948
Residual	46	407.4549	8.857716	
Total	54	1577.345		
<i>Significance F</i>	3.34E-11			
	<i>Coefficients</i>	<i>Standard Error</i>	<i>t Stat</i>	<i>P-value</i>
Intercept	383.4056	139.0296	2.757727	0.008321
Threshold	1.970321	0.884989	2.22638	0.030927
S.D. from back	4.681052	1.05981	4.416878	6.03E-05
Mean	-254.418	72.55963	-3.50633	0.001026
Standard Deviation	-44.2312	15.27105	-2.89641	0.00576
RMS	255.1553	73.43198	3.474716	0.001126
Skewness	11.49642	4.213371	2.728556	0.00898
Entropy	-55.1356	30.4937	-1.8081	0.077131
Contrast	0.009122	0.008317	1.096798	0.278438
Durbin Watson (P-1=8, N=55)	2.408671			
Prediction Mean Squared Error	19.44515			

The regression equation for the upper normal-variation limit uses 8 image statistics and an intercept value to estimate the parameter. Table 4.1 indicates that this set of image statistics can explain 74% of the variation of the algorithm parameter since R^2 is 0.74. The F -statistic of the entire regression equation is highly significant. Also, the prediction mean squared error obtained from the modified cross validation method is 19.44 which is over twice the regression mean squared error of 8.86.

Table 4. 2 ANOVA of Regression of Spin-Echo Lower Normal Variation Limit

<i>Regression Statistics</i>	
Multiple R	0.807382
R Square	0.651865
Adjusted R Square	0.616341
Standard Error	3.279419
Observations	55

table continued on next page

ANOVA

	<i>df</i>	<i>SS</i>	<i>MS</i>	<i>F</i>
Regression	5	986.7342	197.3468	18.35001
Residual	49	526.9749	10.75459	
Total	54	1513.709		
<i>Significance F</i>	3.16E-10			
	<i>Coefficients</i>	<i>Standard Error</i>	<i>t Stat</i>	<i>P-value</i>
Intercept	401.7412	95.78235	4.194314	0.000114
Threshold	1.573197	0.937865	1.677424	0.099828
S.D. from back	3.288742	0.457276	7.192031	3.35E-09
RMS	-5.76148	2.214775	-2.60139	0.012243
Entropy	-27.8568	9.699991	-2.87184	0.006014
Contrast	0.02186	0.008177	2.673321	0.010174
Durbin Watson (P-1=5, N=55)	1.686607			
Prediction Mean Squared Error	17.67488			

The regression equation for the lower normal-variation limit uses 5 image statistics and an intercept value to estimate the parameter. Table 4.2 indicates that this set of image statistics can explain 65% of the variation of the algorithm parameter. The *F*-statistic of the entire regression equation is highly significant. Also, the prediction mean squared error obtained from the modified cross validation method is 17.67 compared to the regression mean squared error of 10.75.

The regression equation for the upper region stopping limit uses 8 image statistics and no intercept value to estimate the parameter. Table 4.3 indicates that this set of image statistics can explain 94% of the variation of the algorithm parameter. The *F*-statistic of the entire regression equation is highly significant. Also, the prediction mean squared error obtained from the modified cross validation method is 15.58 compared to the regression mean squared error of 5.40.

Table 4. 3 ANOVA of Regression of Spin-Echo Upper Region Stopping Limit

<i>Regression Statistics</i>				
Multiple R	0.970197			
R Square	0.941281			
Adjusted R Square	0.911259			
Standard Error	2.322994			
Observations	55			
<i>ANOVA</i>				
	<i>df</i>	<i>SS</i>	<i>MS</i>	<i>F</i>
Regression	8	4065.719	508.2149	94.17836
Residual	47	253.6262	5.396303	
Total	55	4319.345		
<i>Significance F</i>	6.62E-26			
	<i>Coefficients</i>	<i>Standard Error</i>	<i>t Stat</i>	<i>P-value</i>
Intercept	0			
Threshold	1.985319	0.712894	2.784874	0.007694
Variance	0.195572	0.077711	2.516653	0.015319
S.D. from back	-3.74779	2.695356	-1.39046	0.170938
Mean	-7.26695	1.61753	-4.49262	4.57E-05
Standard Deviation	-5.12622	1.327194	-3.86245	0.000342
Skewness	-13.9966	2.932888	-4.7723	1.81E-05
Entropy	68.08716	14.79988	4.600521	3.2E-05
Contrast	0.026807	0.005818	4.607309	3.13E-05
Durbin Watson (P- l=8, N=55)	2.277319			
Prediction Mean Squared Error	15.57838			

The regression equation for the lower region stopping limit uses 8 image statistics and an intercept value to estimate the parameter. Table 4.4 indicates that this set of image statistics can explain 57% of the variation of the algorithm parameter. The F -statistic of the entire regression equation is highly significant. Also, the prediction mean squared error obtained from the modified cross validation method is 5.88 compared to the regression mean squared error of 3.17.

To visually verify the effectiveness of the parameter prediction, all of the parameters for each image were calculated and given in Table B.3 in Appendix B along

with any difference between the modified cross validation prediction and the manually-set parameters. The worst predictions appear to be associated with frame #5 (whose predicted upper normal limit of variation is off by 21) and frame #54 (whose predicted upper region stopping limit is off by 12).

Table 4. 4 ANOVA of Regression of Spin-Echo Lower Region Stopping Limit

<i>Regression Statistics</i>				
Multiple R	0.757125			
R Square	0.573238			
Adjusted R Square	0.499018			
Standard Error	1.780596			
Observations	55			

ANOVA				
	<i>df</i>	<i>SS</i>	<i>MS</i>	<i>F</i>
Regression	8	195.9014	24.48767	7.723544
Residual	46	145.8441	3.170523	
Total	54	341.7455		
<i>Significance F</i>	1.68E-06			
	<i>Coefficients</i>	<i>Standard Error</i>	<i>t Stat</i>	<i>P-value</i>
Intercept	381.3666	108.0767	3.528666	0.00096
Threshold	-1.76543	0.570151	-3.09644	0.003332
Variance	-0.28877	0.085554	-3.3753	0.001506
S.D. from back	10.19911	3.045293	3.349139	0.001625
Mean	-27.7021	7.884657	-3.51342	0.001005
RMS	28.46647	8.194667	3.47378	0.001129
Skewness	7.054011	2.90703	2.426535	0.019221
Entropy	-82.2655	22.51293	-3.65414	0.00066
Contrast	0.005512	0.004906	1.123588	0.267018
Durbin Watson (P-1=8, N=55)	2.508533			
Prediction Mean Squared Error	5.87969			

The original spin-echo image for frame #5 is given in Figure 4.30. The defect image with manually set parameters is given in Figure 4.31 and the defect image with predicted parameters is given in Figure 4.32. The original spin-echo image for frame #54



Figure 4.30 Spin-Echo Frame #5

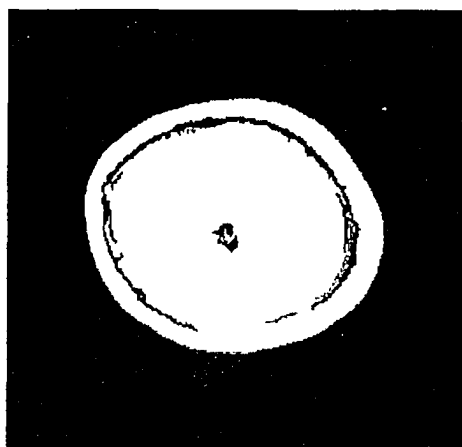


Figure 4.31 Manual Parameters #5

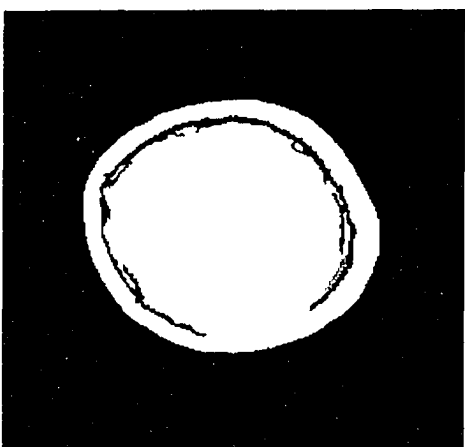


Figure 4.32 Predicted Parameters #5



Figure 4.33 Spin-Echo Frame #54



Figure 4.34 Manual Parameters #54

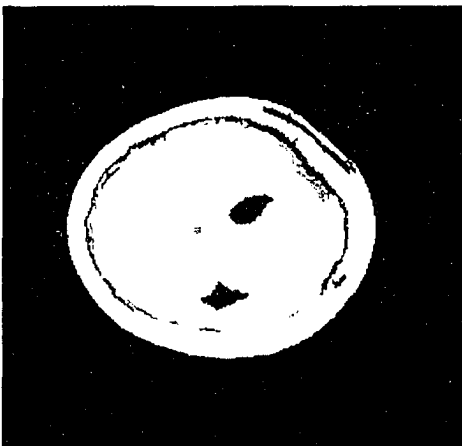


Figure 4.35 Predicted Parameters #54

is given in Figure 4.33. The defect image with manually set parameters is given in Figure 4.34 and the defect image with predicted parameters is given in Figure 4.35.

The worst case represented in Figure 4.32 was caused by an error on the upper normal variation limit which did not allow for enough region seeds to detect the “wet” (light) defects at the 5 o’clock position. Also, the pith defect was missed. The worst case represented in Figure 4.35 was caused by an error on the upper region stopping limit which did not actually have much effect. There was a false “wet” (light) defect in the pith area and the ring defect was not completely detected in the 4 to 7 o’clock position.

A couple of average cases can be illustrated with frames #36 and #43. The original spin-echo image for frame #36 is presented in Figure 4.36. The respective defect image with manually set parameters is in Figure 4.37. The defect image with predicted parameters is given in Figure 4.38. The original spin-echo image for frame #43 is given in Figure 4.39. The respective defect image with manually set parameters is given in Figure 4.40 and the defect image with predicted parameters is given in Figure 4.41.



Figure 4. 36 Spin-Echo Frame #36

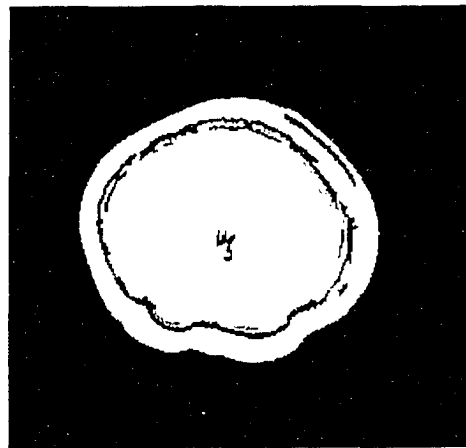


Figure 4. 37 Manual Parameters #36

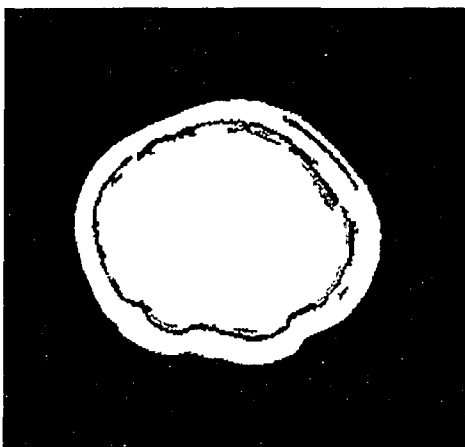


Figure 4. 38 Predicted Parameters #36

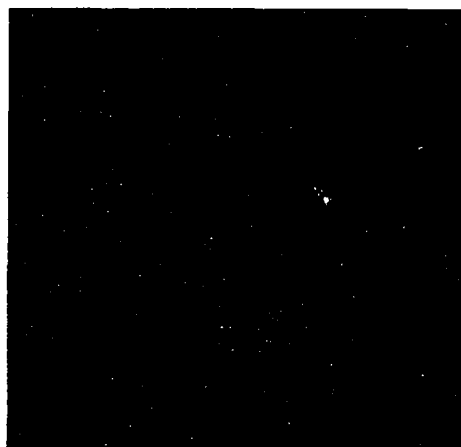


Figure 4. 39 Spin-Echo Frame #43

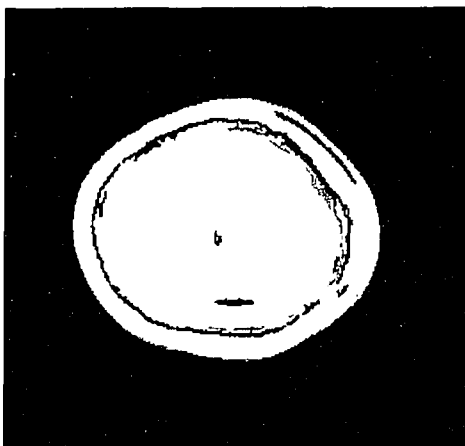


Figure 4. 40 Manual Parameters #43

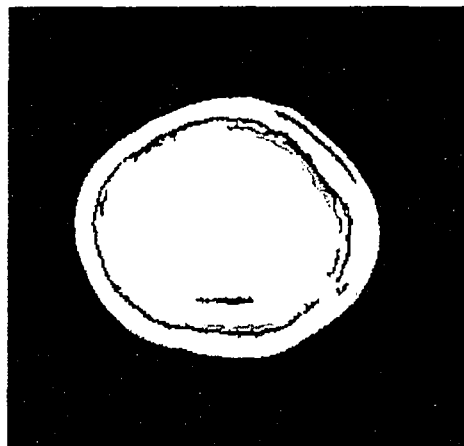


Figure 4. 41 Predicted Parameters #43

An average case of predicting the algorithm parameters as shown in Figure 4.38 indicates that the defect image is only slightly different from the defect image where the parameters were set manually. The main difference being the missed pith area. Another average case as shown in Figure 4.41 indicates more differences. The main difference is that the pith defect was missing and the knot area that was in the 6 o'clock position had "bled" out to the left. In any event, the remaining differences are very slight. The results from all four regressions and the visual examination of the worst cases and the average

cases indicate that the algorithm parameters can be automatically adjusted to the individual image based upon the image statistics.

4.4 Comparison of Spin-Echo Defect Detection Algorithm to Ordinary Thresholding

4.4.1 Overview

Two methods of comparison between the proposed spin-echo defect algorithm and ordinary thresholding are presented below. The first method adjusts the upper and lower thresholds of the spin-echo images until the ordinary (flat) threshold method generates the same number of wet and dry defect pixels as the spin-echo defect detection algorithm. The ordinary thresholds used are given in Table B.4 in Appendix B. The images are compared visually and by a count of undetected defect pixels and false alarm pixels to check the results of the ordinary threshold method.

4.4.2 Using thresholding to find equivalent numbers of defect pixels

For ordinary thresholding, the threshold was adjusted until the count of the defect pixels was equal to the number of defect pixels obtained by the spin-echo defect detection method. Some typical results of the ordinary thresholding method are given in Figures 4.42, through 4.46. For ease of comparison, Figures 4.12, 4.14, 4.16, 4.18 and 4.20 are repeated again next to the respective ordinary threshold defect images.

By comparison of Figures 4.42 through 4.46 (ordinary thresholding results) with Figures 4.12, 4.14, 4.16, 4.18, and 4.20 (spin-echo defect detection algorithm results), it is self-evident that the ordinary thresholding method cannot find as many properly-positioned defect pixels without generating a large number of false alarms.

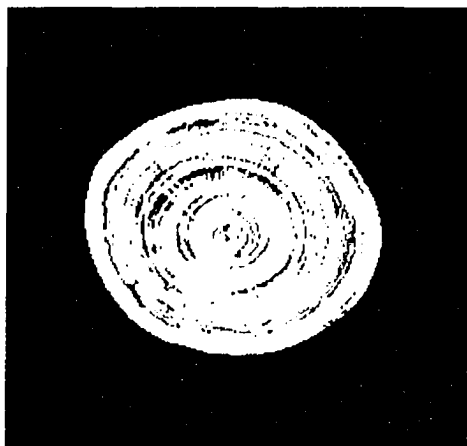
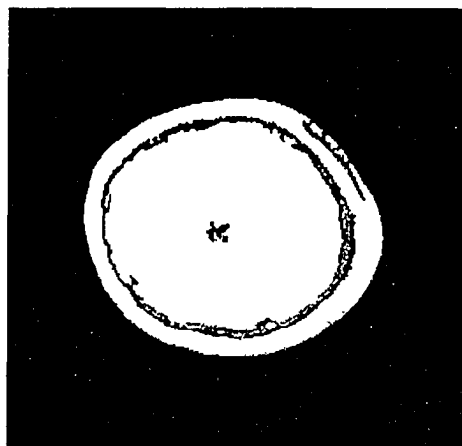


Figure 4.42 Threshold Defects # 15



(Figure 4.12 Defects for Frame # 15)

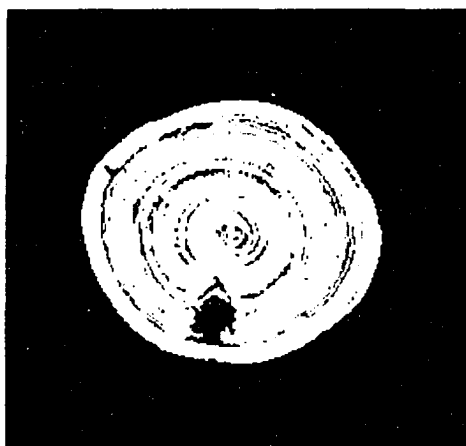
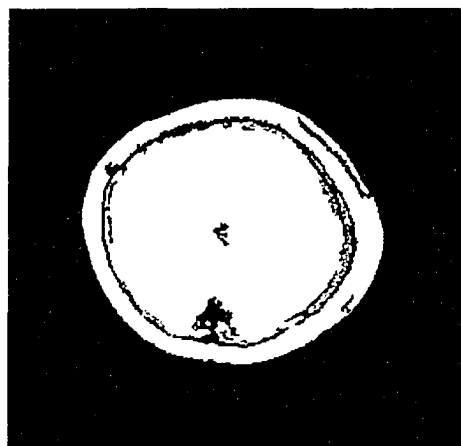


Figure 4.43 Threshold Defects # 20



(Figure 4.14 Defects for Frame # 20)

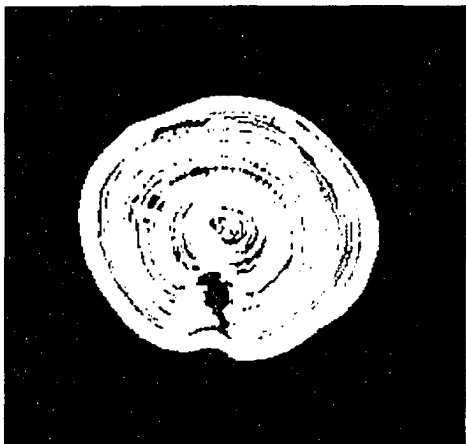
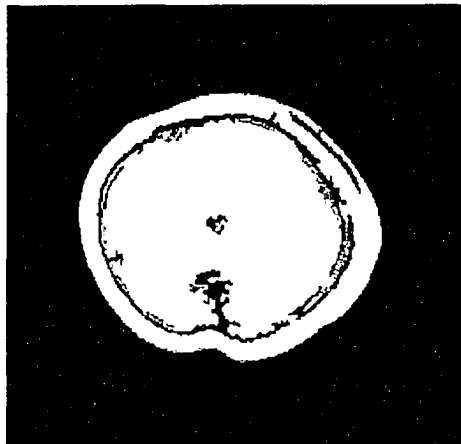


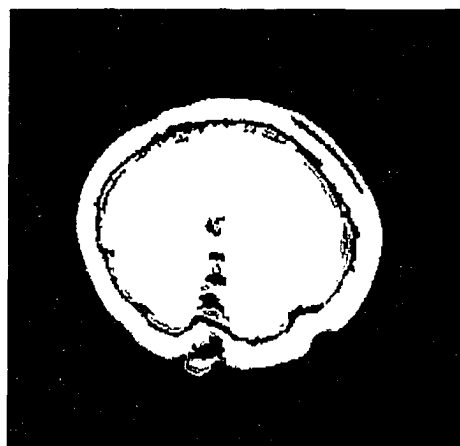
Figure 4.44 Threshold Defects # 25



(Figure 4.16 Defects for Frame # 25)



Figure 4.45 Threshold Defects # 30



(Figure 4.18 Defects for Frame # 30)

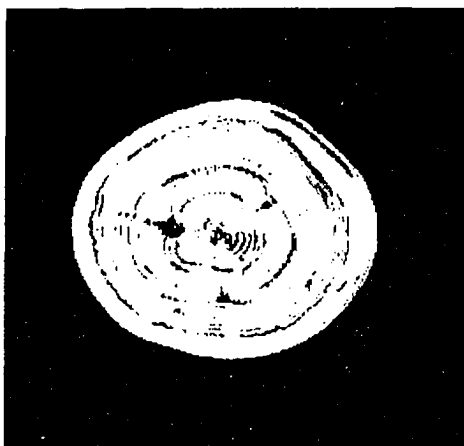


Figure 4.46 Threshold Defects # 50



(Figure 4.20 Defects for Frame # 50)

4.4.3 Testing various threshold levels for defect detection

Because of the large number of false alarms in Figures 4.42 through 4.46, it was of interest to determine if the ordinary thresholds could be adjusted for less false-alarm defect regions. This would entail abandoning the goal of seeking to detect the equivalent number of pixels as the proposed algorithm had detected. The lower ordinary threshold was set at 5 different levels (gray level < 40, 50, 60, 70 and 80) for the same frame, #20, in Figures 4.47 through 4.51.

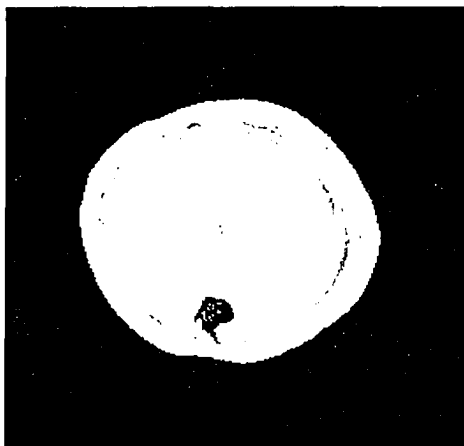


Figure 4. 47 Lower Threshold at 40

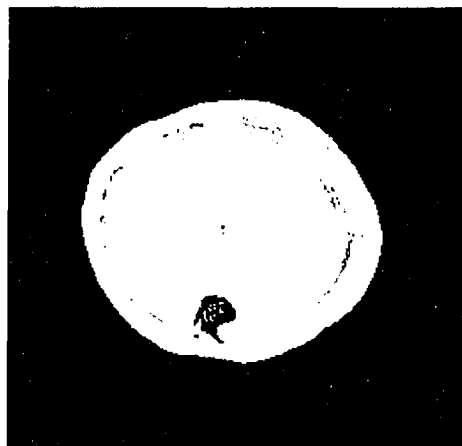


Figure 4. 48 Lower Threshold at 50

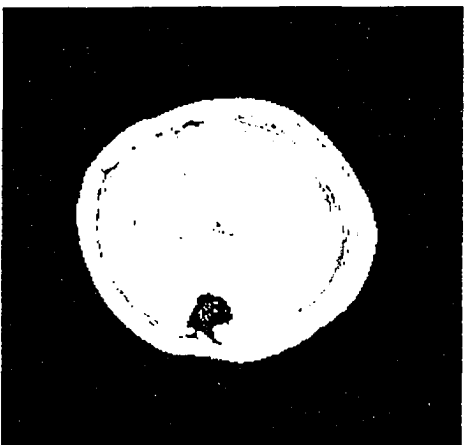


Figure 4. 49 Lower Threshold at 60



Figure 4. 50 Lower Threshold at 70

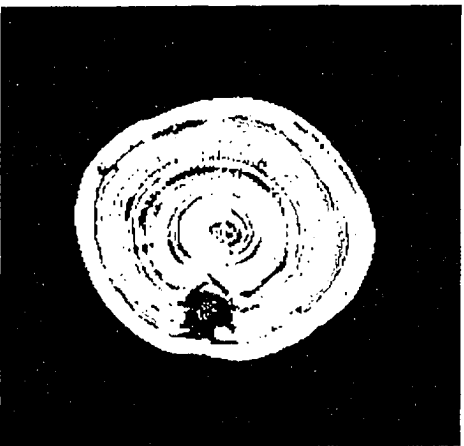
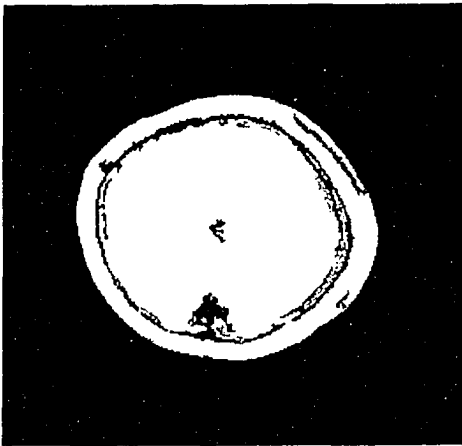


Figure 4. 51 Lower Threshold at 80



(Figure 4. 14 Defects for Frame # 20)

From Figure 4.47, the ordinary thresholding method with a threshold of 40 begins to detect some of the more extreme defect pixels. Each figure has the threshold moved 10 gray levels higher to detect more defects. From Figure 4.49, it can be seen that more defect pixels are found. In fact, the knot has been fully detected. There are just a very small number of false alarms that are beginning to appear but they are quite small. However, compared to Figure 4.14, the thresholding method has yet to detect any amount of the scar tissue in the 2 o'clock position nor much of the remaining defect area. The threshold is raised by 10 gray levels again in Figure 4.50 where a number of false alarms are beginning to appear. Figure 4.51 shows clearly that any attempt to capture more defect area quickly generates numerous false alarm defect regions.

This result is rather important because many research efforts are being put into more sophisticated methods of determining the proper threshold. Yet any threshold is a compromise of undetected defect pixels and false alarms. Also, these results indicate that defect detection methods that simultaneously consider adjacency relationships and gray level perform better than threshold methods that consider only gray level.

4.5 Results of Echo-Planar Defect Detection Algorithm

4.5.1 Separation of foreground and background

The modified moment-preserving thresholding method was used to separate the log foreground from the surrounding air background. In addition, the result was morphologically smoothed with an 11-pixel diameter ball. Some of the results are demonstrated in Figures 4.52 through 4.61. The respective pre-processed echo-planar image is also presented for comparison purposes. These thresholded images were used as a mask image to define the foreground for the rest of the image processing steps.

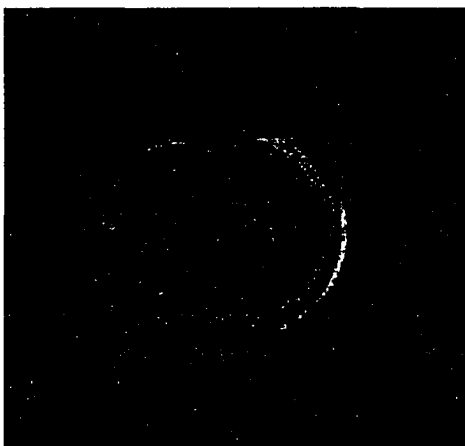


Figure 4. 52 Echo-Planar Frame #15

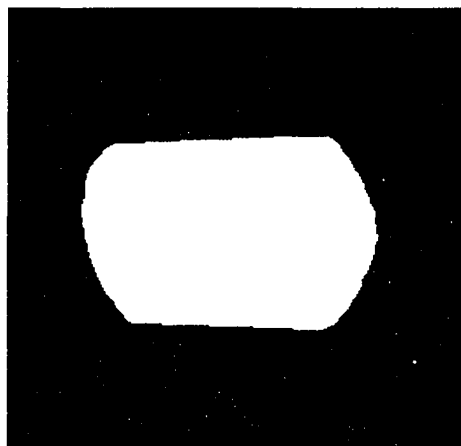


Figure 4. 53 Mask for Frame #15



Figure 4. 54 Echo-Planar Frame #20

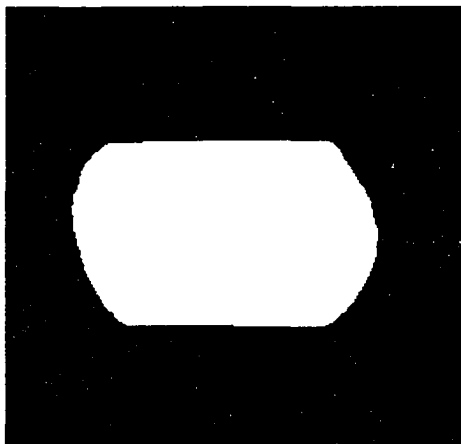


Figure 4. 55 Mask for Frame #20



Figure 4. 56 Echo-Planar Frame #25

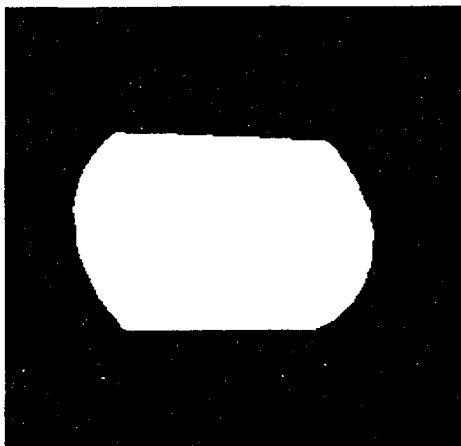


Figure 4. 57 Mask for Frame #25



Figure 4. 58 Echo-Planar Frame #30

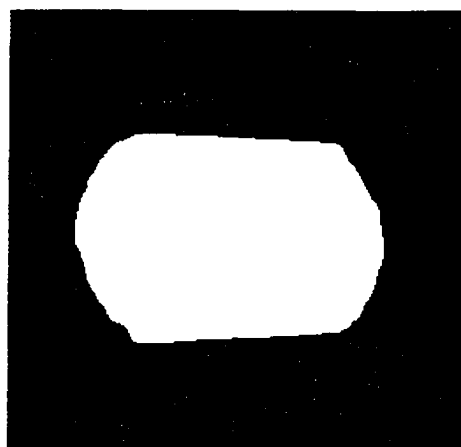


Figure 4. 59 Mask for Frame #30

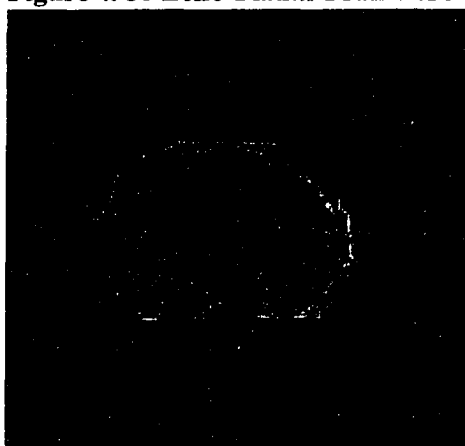


Figure 4. 60 Echo-Planar Frame #50

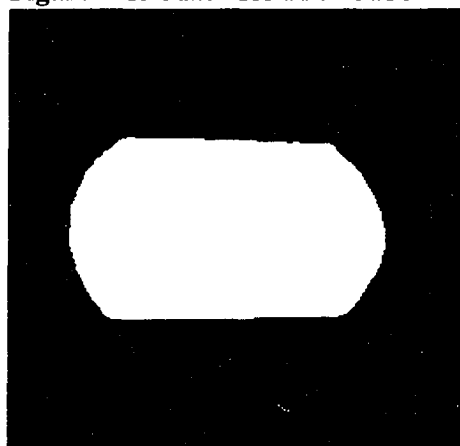


Figure 4. 61 Mask for Frame #50

4.5.2 Image results of unprocessed echo-planar defect detection algorithm

After the images were passed through an average filter, the result was subtracted from the original image to obtain the deviations in gray levels. Next, the difference image was passed through a positive threshold to capture high (very moist) areas and through a negative threshold to capture low (very dry) areas. The defect regions were grown from these seeds using pixel aggregation as described in Section 3.2.5 and stopped when the gray-level deviation approached the deviation of the surrounding clear wood. Some examples of these defect images are given in Figures 4.62 through 4.66.

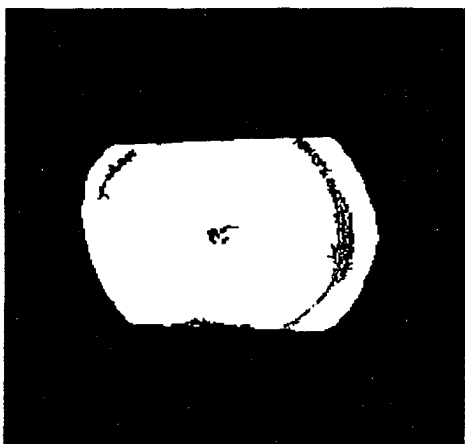


Figure 4. 62 Defects - Echo-Planar #15

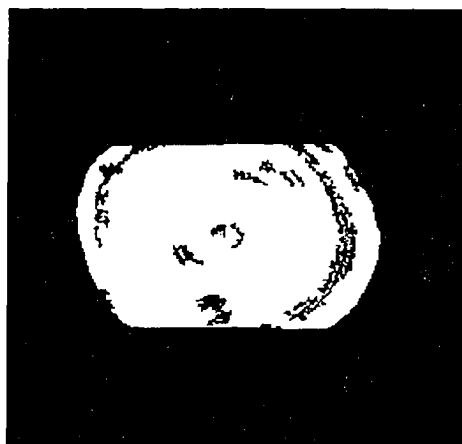


Figure 4. 63 Defects - Echo-Planar #20

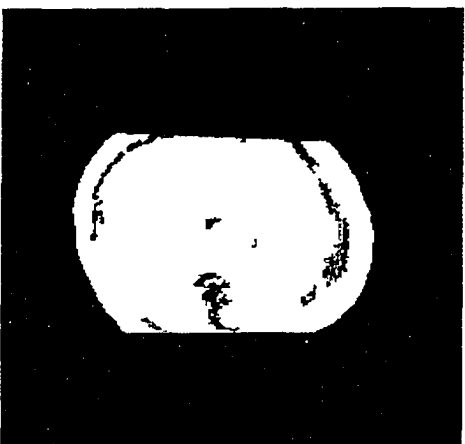


Figure 4. 64 Defects - Echo-Planar #25

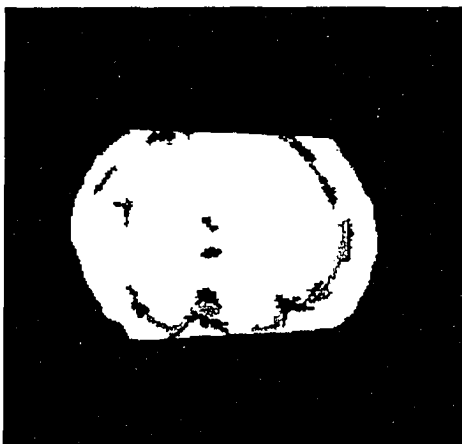


Figure 4. 65 Defects - Echo-Planar #30

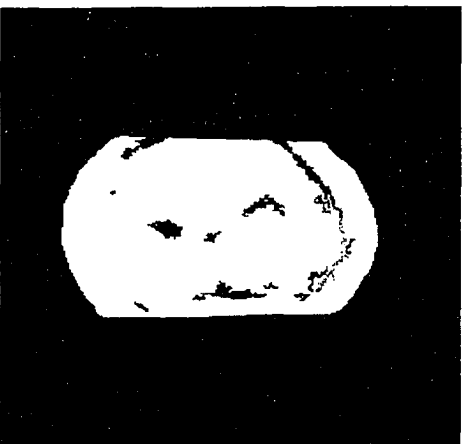


Figure 4. 66 Defects - Echo-Planar #50

From Figures 4.62 through 4.66, it can be seen that most of the defects found on the spin-echo images can also be found from the unprocessed echo-planar images. The figures show that the ring defect is usually missing in the 7 o'clock to 8 o'clock areas. Also, the scar tissue at the 1 o'clock to 2 o'clock position is missing in 4 of the 5 representative images. The major knots and the pith are always found. However, Figures 4.63 and 4.65 have a couple of false alarms.

4.6 Automatic Determination of Parameters for Echo-Planar Defect Detection Algorithm

In automatically determining the parameters to use, the first step was to find the appropriate window size for the smoothing operation. As explained in Section 3.3.2, the window size that maximized the skewness of the residuals was selected. The results for each echo-planar frame are shown in Figures 4.67 through 72.

Using the appropriate window size for each frame and using the previously manually-obtained spin-echo defect images as the goal, the loss function as described in Section 3.3.3 was used to determine the optimum parameters to use on each of the spin-echo images. The manually-set parameters that gave the results as typified in Figures 4.62 through 4.66 are given in Table B.5 in Appendix B. The image statistics for the echo-planar images are listed in Table B.6 in Appendix B.

To find the regression equations that could predict the parameters, all the parameters were regressed against the set of image statistics. Backward elimination was used to find the set of image statistics that minimized the *MSE* for the prediction equation.

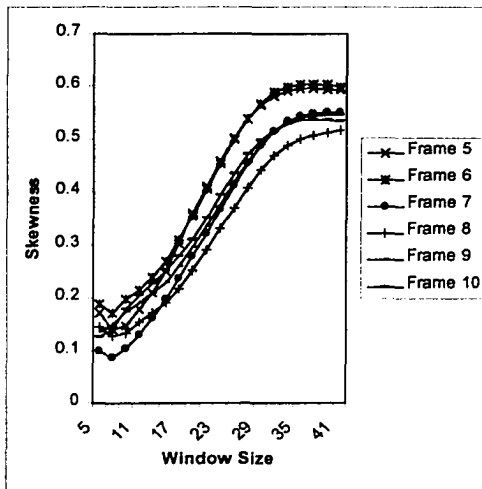


Figure 4.67 Skewness for Frames 5-10

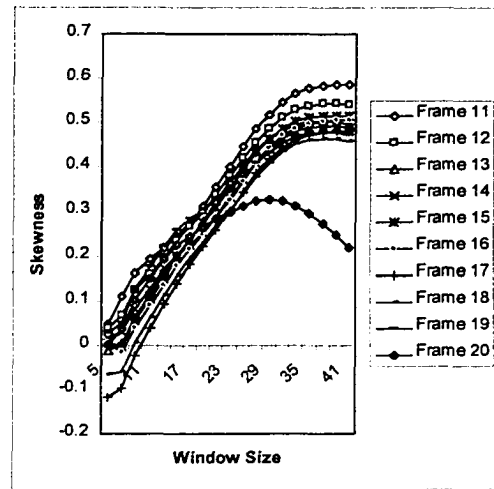


Figure 4.68 Skewness for Frames 11-20

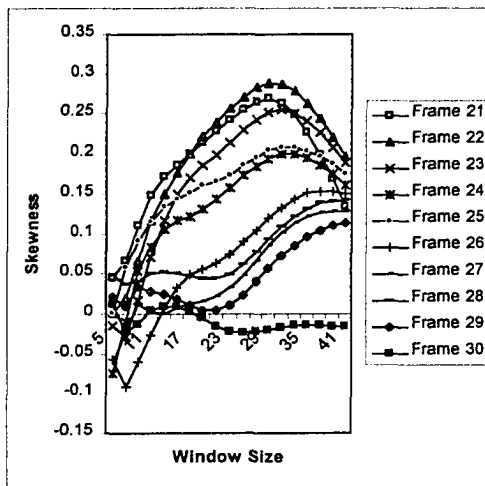


Figure 4.69 Skewness for Frames 21-30

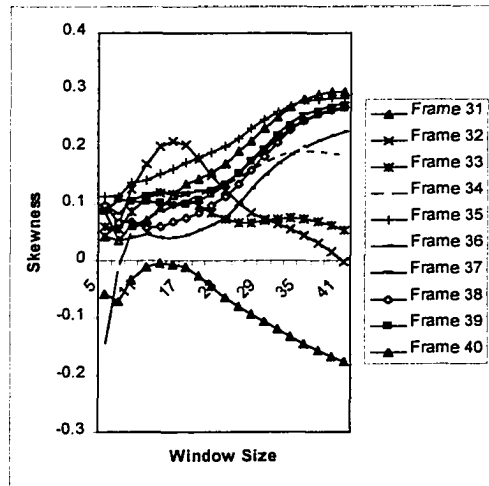


Figure 4.70 Skewness for Frames 31-40

In order to obtain the prediction mean squared error, the correlation between the image statistics in adjacent frames was plotted against the distance between images (frames). The plot is given in Figure 4.73. The correlation between image statistics drops quickly as the distance increases to 10 frames. Beyond 10 frames, the correlation tends to level out. For that reason, no data was used in fitting the regression equation for a distance of 10 frames on either side of a predicted data point. Except at the ends, each of

the data points was predicted from a regression equation that used 55-21 or 34 of the points that were furthestmost away. The sum of the squared differences of the predictions and the actual values were calculated for comparison to the regression MSE.

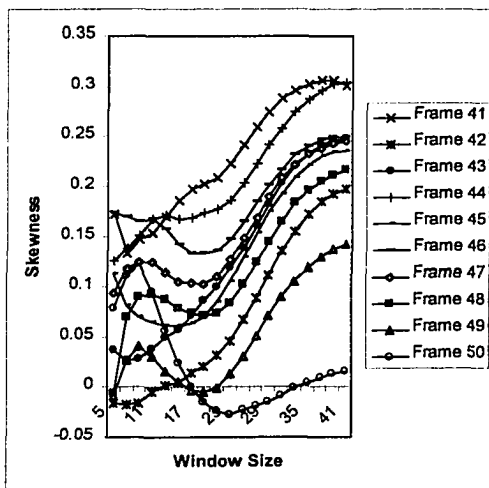


Figure 4.71 Skewness for Frames 41-50

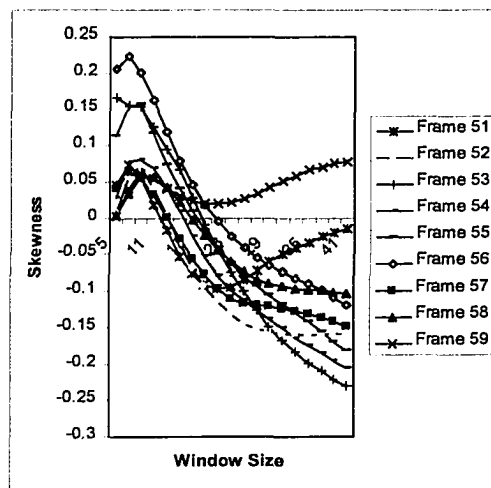


Figure 4.72 Skewness for Frames 51-60

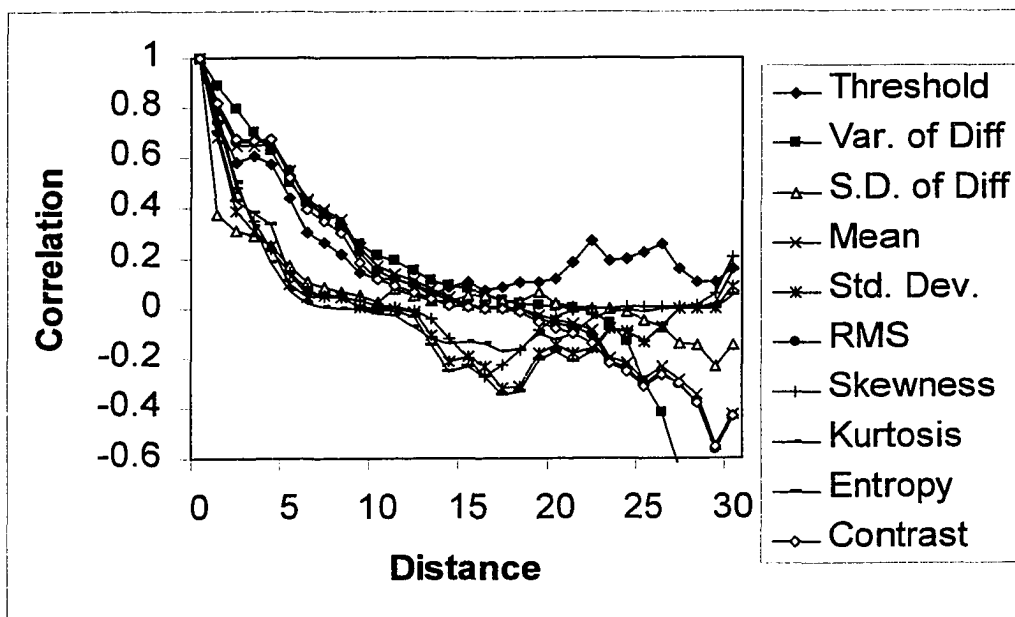


Figure 4.73 Correlation of Image Statistics Between Frames

The results of the regression for each echo-planar parameter are given in Tables 4.5 through 4.10. The regression equation for the upper normal-variation limit uses 8 image statistics and no intercept value to estimate the parameter. Table 4.5 indicates that this set of image statistics can explain 39% of the variation of the algorithm parameter. The F -statistic of the entire regression equation is highly significant. Also, the prediction mean squared error obtained from the modified cross validation method is 47.93 which is three times the regression mean squared error of 16.97.

Table 4. 5 ANOVA of Regression of Echo-Planar Upper Normal Variation Limit

<i>Regression Statistics</i>				
Multiple R	0.624925			
R Square	0.390532			
Adjusted R Square	0.278483			
Standard Error	4.119105			
Observations	55			
ANOVA				
	<i>df</i>	<i>SS</i>	<i>MS</i>	<i>F</i>
Regression	8	510.986	63.87325	3.764551
Residual	47	797.4504	16.96703	
Total	55	1308.436		
<i>Significance F</i>	0.001825			
	<i>Coefficients</i>	<i>Standard Error</i>	<i>t Stat</i>	<i>P-value</i>
Intercept	0			
Threshold	-0.56596	0.245123	-2.30888	0.025395
Variance	-0.77037	0.213932	-3.60101	0.000761
S.D. from back	28.76293	7.984144	3.602507	0.000758
Mean	163.4424	45.6518	3.580197	0.000811
Standard Deviation	45.78419	12.41097	3.689009	0.000583
RMS	-168.281	47.10034	-3.57282	0.000829
Skewness	23.30405	7.258692	3.210503	0.002391
Entropy	-45.043	16.99951	-2.64967	0.010941
Durbin Watson (P-1=8, N=55)	1.483966			
Prediction Mean Squared Error	47.93235			

The regression equation for the lower normal-variation limit uses 4 image statistics and an intercept value to estimate the parameter. Table 4.6 indicates that this set of image statistics can explain 56% of the variation of the algorithm parameter. The F -statistic of the entire regression equation is highly significant. Also, the prediction mean squared error obtained from the modified cross validation method is 31.70 compared to the regression mean squared error of 20.62.

Table 4. 6 ANOVA of Regression of Echo-Planar Lower Normal Variation Limit

<i>Regression Statistics</i>				
Multiple R	0.746142			
R Square	0.556728			
Adjusted R Square	0.521266			
Standard Error	4.540968			
Observations	55			
ANOVA				
	<i>df</i>	<i>SS</i>	<i>MS</i>	<i>F</i>
Regression	4	1294.908	323.7269	15.69936
Residual	50	1031.019	20.62039	
Total	54	2325.927		
<i>Significance F</i>	2.19E-08			
	<i>Coefficients</i>	<i>Standard Error</i>	<i>t Stat</i>	<i>P-value</i>
Intercept	54.80867	11.37522	4.818251	1.39E-05
Threshold	-0.39247	0.148878	-2.6362	0.011138
Variance	0.096317	0.025816	3.730844	0.000488
Standard Deviation	-1.34145	0.354761	-3.78128	0.000417
RMS	0.23813	0.16627	1.432186	0.158313
Durbin Watson (P- l=4, N=55)	1.548528			
Prediction Mean Squared Error	31.70223			

The regression equation for the upper region stopping limit uses 5 image statistics and an intercept value to estimate the parameter. Table 4.7 indicates that this set of image statistics can explain only 19% of the variation of the algorithm parameter. The F -statistic of the entire regression equation is not significant. Despite the poor regression

results, the prediction errors are small. The prediction mean squared error obtained from the modified cross validation method is 15.69 compared to the regression mean squared error of 9.33.

Table 4. 7 ANOVA of Regression of Echo-Planar Upper Region-Stopping Limit

<i>Regression Statistics</i>				
Multiple R	0.440709			
R Square	0.194224			
Adjusted R Square	0.112002			
Standard Error	3.054449			
Observations	55			
ANOVA				
	<i>df</i>	<i>SS</i>	<i>MS</i>	<i>F</i>
Regression	5	110.1922	22.03844	2.362192
Residual	49	457.1533	9.329658	
Total	54	567.3455		
<i>Significance F</i>	0.053484			
	<i>Coefficients</i>	<i>Standard Error</i>	<i>t Stat</i>	<i>P-value</i>
Intercept	353.7563	261.1175	1.354778	0.181702
Mean	1.534906	1.055529	1.454158	0.152278
Standard Deviation	4.390321	1.62929	2.694622	0.009625
Kurtosis	-2.58907	1.503351	-1.7222	0.091341
Entropy	-80.1925	40.05126	-2.00225	0.050809
Contrast	-0.00645	0.004265	-1.51278	0.13676
Durbin Watson (P-1=5, N=55)	1.87854			
Prediction Mean Squared Error	15.69356			

The regression equation for the lower region stopping limit uses 7 image statistics and an intercept value to estimate the parameter. Table 4.8 indicates that this set of image statistics can explain 47% of the variation of the algorithm parameter. The *F*-statistic of the entire regression equation is highly significant. Also, the prediction mean squared error obtained from the modified cross validation method is 12.68 compared to the regression mean squared error of 4.63.

Table 4. 8 ANOVA of Regression of Echo-Planar Lower Region-Stopping Limit

<i>Regression Statistics</i>				
Multiple R	0.684712			
R Square	0.46883			
Adjusted R Square	0.38972			
Standard Error	2.150863			
Observations	55			
<i>ANOVA</i>				
	<i>df</i>	<i>SS</i>	<i>MS</i>	<i>F</i>
Regression	7	191.9135	27.41622	5.926278
Residual	47	217.4319	4.626212	
Total	54	409.3455		
<i>Significance F</i>	5.69E-05			
	<i>Coefficients</i>	<i>Standard Error</i>	<i>t Stat</i>	<i>P-value</i>
Intercept	125.7414	51.46152	2.443406	0.01836
Threshold	-0.1942	0.125774	-1.54403	0.129287
Variance	-0.02986	0.016117	-1.85277	0.0702
Mean	75.95978	28.72831	2.644074	0.011099
Standard Deviation	21.64333	7.558859	2.863306	0.006245
RMS	-80.5866	29.68334	-2.71488	0.009243
Skewness	17.73292	4.37713	4.051266	0.00019
Contrast	0.007546	0.002925	2.580108	0.013062
Durbin Watson (P-1=7, N=55)	1.949991			
Prediction Mean Squared Error	12.67714			

The regression equation for the pith-area lower normal variation limit uses 7 image statistics and an intercept value to estimate the parameter. Table 4.9 indicates that this set of image statistics can explain 62% of the variation of the algorithm parameter. The *F*-statistic of the entire regression equation is highly significant. Also, the prediction mean squared error obtained from the modified cross validation method is 12.68 compared to the regression mean squared error of 4.63.

The regression equation for the pith-area lower region stopping limit uses 7 image statistics and an intercept value to estimate the parameter. Table 4.10 indicates that this

set of image statistics can explain 62% of the variation of the algorithm parameter. The F -statistic of the entire regression equation is highly significant. Also, the prediction mean squared error obtained from the modified cross validation method is 66.12 which is twice the regression mean squared error of 33.12.

Table 4. 9 ANOVA of Regression of Echo-Planar Pith-Area Lower Normal-Variation Limit

<i>Regression Statistics</i>				
Multiple R	0.785223			
R Square	0.616575			
Adjusted R Square	0.559469			
Standard Error	5.755427			
Observations	55			
ANOVA				
	<i>df</i>	<i>SS</i>	<i>MS</i>	<i>F</i>
Regression	7	2503.564	357.652	10.79706
Residual	47	1556.872	33.12494	
Total	54	4060.436		
<i>Significance F</i>	5.04E-08			
	<i>Coefficients</i>	<i>Standard Error</i>	<i>t Stat</i>	<i>P-value</i>
Intercept	439.4524	145.3935	3.022504	0.00405
Threshold	0.701847	0.326065	2.152479	0.036526
Mean	-82.7401	66.71929	-1.24012	0.221086
Standard Deviation	-22.5463	17.60376	-1.28076	0.206561
RMS	78.98599	69.11398	1.142837	0.258895
Skewness	-12.6826	9.261915	-1.36933	0.177404
Kurtosis	11.22981	2.45792	4.568827	3.55E-05
Contrast	0.025906	0.008346	3.104138	0.003228
Durbin Watson (P-1=7, N=55)	1.88056			
Prediction Mean Squared Error	65.11735			

The regression equation for the pith-area lower region stopping limit uses 8 image statistics and an intercept value to estimate the parameter. Table 4.10 indicates that this set of image statistics can explain 34% of the variation of the algorithm parameter. The F -statistic of the entire regression equation is highly significant. Also,

the prediction mean squared error obtained from the modified cross validation method is 15.99 which is almost three times the regression mean squared error of 5.73.

Table 4. 10 ANOVA of Regression of Echo-Planar Pith-Area Lower Region-Growing Stopping Limit

<i>Regression Statistics</i>				
Multiple R	0.587034			
R Square	0.344609			
Adjusted R Square	0.230628			
Standard Error	2.393554			
Observations	55			
ANOVA				
	<i>df</i>	<i>SS</i>	<i>MS</i>	<i>F</i>
Regression	8	138.5705	17.32131	3.02339
Residual	46	263.5386	5.729101	
Total	54	402.1091		
<i>Significance F</i>	0.001394			
	<i>Coefficients</i>	<i>Standard Error</i>	<i>t Stat</i>	<i>P-value</i>
Intercept	271.8657	239.3804	1.135705	0.261963
Threshold	-0.43052	0.138171	-3.11587	0.003156
Mean	57.15899	27.74707	2.060001	0.045082
Standard Deviation	18.01836	7.479527	2.409025	0.020057
RMS	-57.5862	28.74313	-2.00348	0.051037
Skewness	6.532284	4.001627	1.632407	0.109422
Kurtosis	-5.91875	1.466922	-4.03481	0.000204
Entropy	-60.7969	39.12609	-1.55387	0.127069
Contrast	-0.00533	0.003515	-1.51762	0.135954
Durbin Watson (P-1=8, N=55)	2.459397			
Prediction Mean Squared Error	15.9932			

To visually verify the effectiveness of the parameter prediction for the echo-planar images, all of the parameters for each image were calculated and given in Table B.7 in Appendix B along with any difference between the modified cross validation prediction and the manually-set parameters. The worst predictions (the ones where the defect images had the worst *SNR*) appear to be associated with frame #26 (whose predicted

lower region stopping limit is off by 7) and frame #33 (whose predicted lower region stopping limit is off by 5). The original echo-planar image for frame #26 is given in Figure 4.74. The defect image with manually set parameters is given in Figure 4.75 and the defect image with predicted parameters is given in Figure 4.76. The original echo-planar image for frame #33 is given in Figure 4.77. The defect image with manually set parameters is given in Figure 4.78 and the defect image with predicted parameters is given in Figure 4.79.

The worst case represented in Figure 4.76 was caused by an error on the upper normal variation limit which was set too low. This allowed seeds for false “wet” (light) defects to develop. Also, there was an error on the upper region stopping limit which allowed the stopping limit to be set too low. This caused the “wet” (light) regions (which were already false alarms) to grow too far. The worst case represented in Figure 4.79 was caused by errors on both the upper and lower region stopping limits. Again, these were set too low and they caused the identified defect regions (both light and dark) to grow too far.



Figure 4. 74 Echo-Planar Frame #26

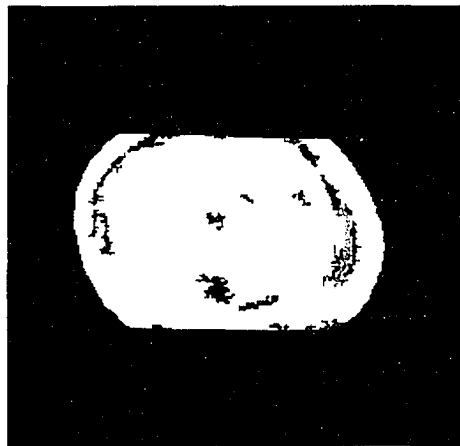


Figure 4. 75 Manual Parameters #26

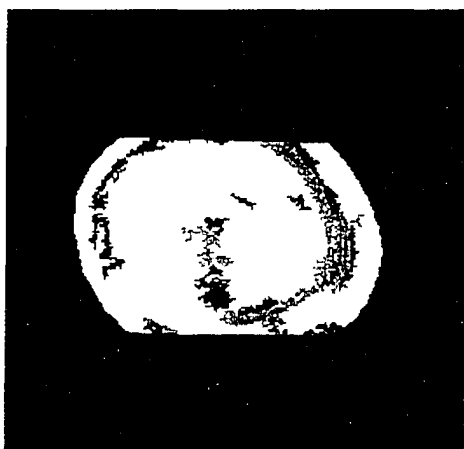


Figure 4. 76 Predicted Parameters #26

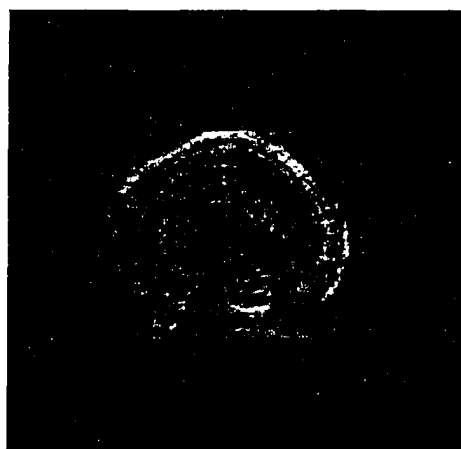


Figure 4. 77 Echo-Planar Frame #33

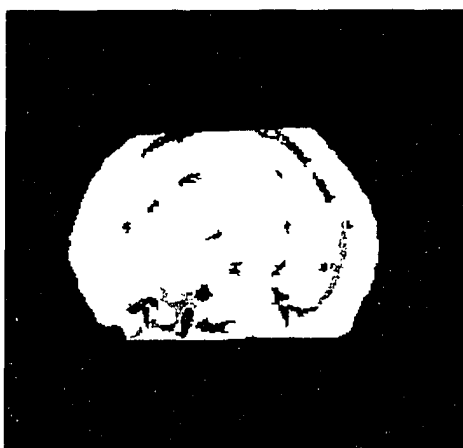


Figure 4. 78 Manual Parameters #33

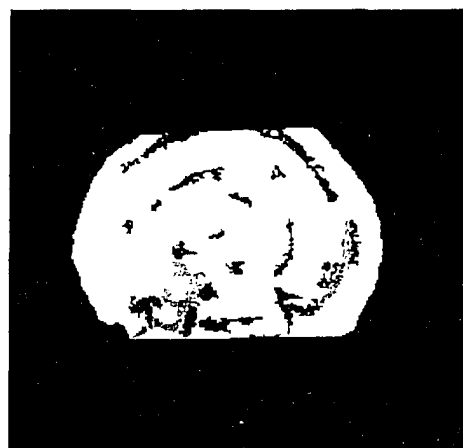


Figure 4. 79 Predicted Parameters #33

Based upon the *SNR* of all the echo-planar predicted defect images, a couple of average cases can be illustrated with frame #34 and frame #53. The original echo-planar image for frame #34 is presented in Figure 4.80. The respective defect image with manually set parameters is presented in Figure 4.81. The defect image with predicted parameters is given in Figure 4.82. The original echo-planar image for frame #53 is given in Figure 4.83. The respective defect image with manually set parameters is given in Figure 4.84 and the defect image with predicted parameters is given in Figure 4.85.



Figure 4. 80 Echo-Planar Frame #34

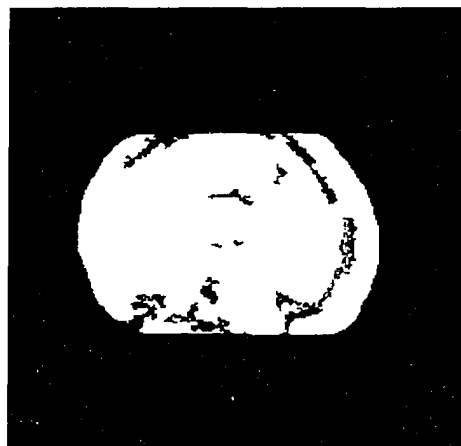


Figure 4. 81 Manual Parameters #34

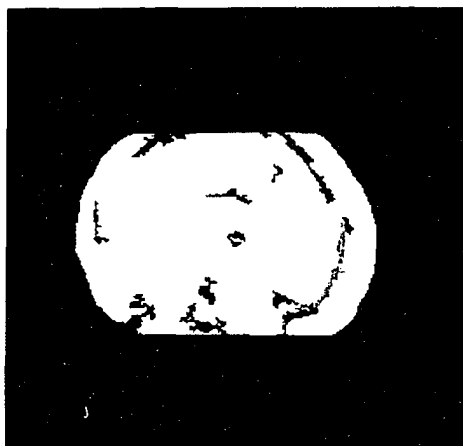


Figure 4. 82 Predicted Parameters #34

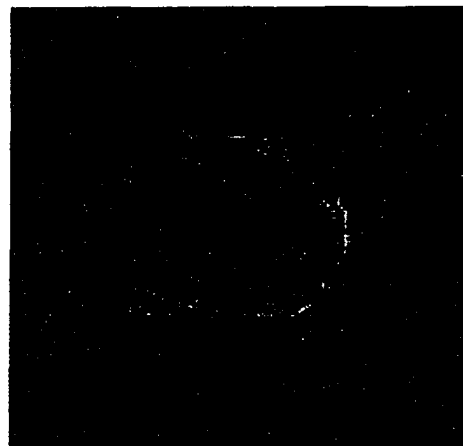


Figure 4. 83 Echo-Planar Frame#53



Figure 4. 84 Manual Parameters #53

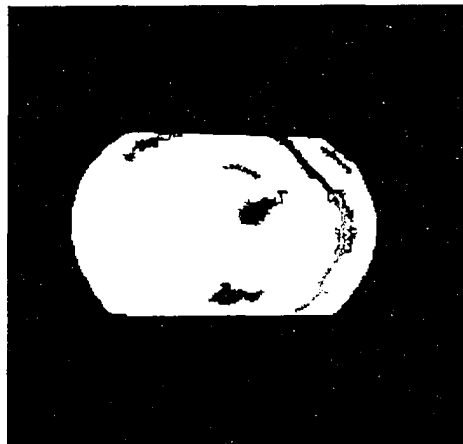


Figure 4. 85 Predicted Parameters #53

An average case of predicting the algorithm parameters as shown in Figure 4.82 indicates that the defect image is virtually the same as the defect image where the parameters were set manually. Another average case as shown in Figure 4.85 indicates a couple of differences. The main difference is that the pith defect was missed.

In all, the echo-planar image statistics were not as good predictors of the algorithm parameters as they were for the spin-echo images because the prediction mean squared errors of the regressions were larger. Also, for the upper region growing stopping limit, the overall regression was marginally not significant. Strangely enough, for that particular parameter, the prediction capability was not impaired.

There may be more image statistics that could reduce the error or there may be polynomial forms of the prediction equation that could be better. Although the results were not as good as the spin-echo case, the results of the regression and the visual inspection of the worst and average cases still indicate that the algorithm parameters can be automatically adjusted to the individual image based upon image statistics.

4.7 Comparison of General Image Enhancement Methods on Echo-Planar Images

In this section, the echo-planar images are enhanced by three general types of preprocessing methods in an attempt to improve the appearance of the original image. The signal-noise ratio of the unprocessed echo-planar images using the spin-echo images as the noise-free basis for comparison is given in Table B.8 in Appendix B along with the signal-noise ratio of the echo-planar images after three general image enhancement methods (median, despeckle algorithm, and average window) were applied.

The ANOVA table for the comparison of four sets of data is given in Table 4.11. Recall that the experiment design was given in Chapter 3, Section 3.6.3. The treatments

refer to the four methods of presenting the echo-planar images: unprocessed, median filtered, despeckled, and average filtered. The blocks refer to the individual frames. The table indicates that blocking was a good idea since it captured the largest portion (.698/.750) of the variation. The first F -statistic is the ratio of the mean square of the treatments and the mean square error and is used to test whether the treatments have a significant effect on the SNR of the echo-planar images. The F -statistic of 23.16 is very significant as indicated by the extremely low P -value of 1.34×10^{-12} . We can conclude that the choice of methods of processing echo-planar images has an effect on the quality of the image as measured by the SNR . The second F -statistic is the ratio of the mean square of the blocks and the mean square error and is given although a strict F -test can not be performed on the blocking factor. However, large ratios would indicate that the blocking factor has a large effect and the noise reduction obtained by blocking was helpful (Montgomery, 1997).

In Table 4.11, Tukey's Test procedure was used to tell us that the standard error of the treatment means were 0.001963 and that any two treatments that were less than 0.007183 apart would not be significantly different. In Table 4.11, the treatment averages were arranged in order and the treatments that were not significantly different were connected with the heavy bars to indicate similarity of results. Thus, the median filter (window size: 3x3) and the unprocessed images produced similar results. The despeckled images and the average-filtered images were actually worse than the unprocessed images based upon the SNR measure (using the spin-echo image as the noise-free basis).

Table 4. 11 ANOVA of SNR of Echo-Planar Images (Unprocessed and General Processing Methods)

<i>Source of Variation</i>	<i>Sum of Squares</i>	<i>Degrees of Freedom</i>	<i>Mean Square</i>	<i>F₀</i>	<i>Significance of F</i>
<i>Treatments</i>	0.015257	3	0.005086	23.16464	1.33935E-12
<i>Blocks</i>	0.697969	56	0.012464	56.77134	
<i>Error</i>	0.036883	168	0.00022		
<i>Total</i>	0.750109	227			

Tukey's Test

$S\bar{y}_i$	0.001963		
q.05(4,168)	3.66	T.05	0.007183
q.01(4,168)	4.45	T.01	0.008733

<i>Treatment</i>	<i>Unprocessed</i>	<i>XYMedian</i>	<i>Despeckled</i>	<i>Average</i>
<i>Treatment Average</i>	0.494977	0.4899	0.487218	0.472937

Before dismissing the general image enhancement methods, the images were compared visually. Some of the images after the three general image enhancements were applied are given in Figures 4.86 through 4.94. The respective, unprocessed echo-planar figures, Figure 4.54, Figure 4.58 and Figure 4.60 are also reproduced for ease of comparison. The figures for frame #20, frame #30 and frame #50 are grouped together.

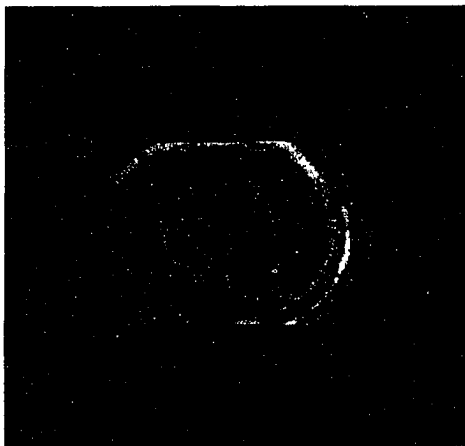
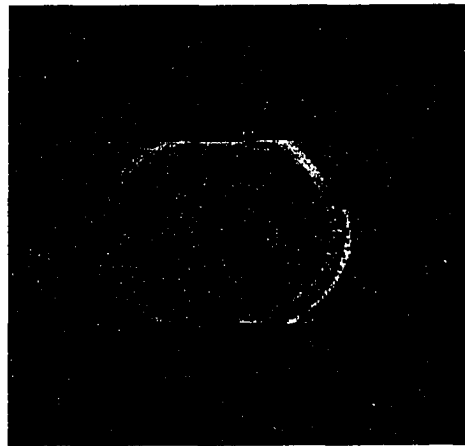
**Figure 4. 86 Median Echo-planar #20****Figure 4. 87 Despeckle Echo-planar #20**



Figure 4.88 Average Echo-planar #20



(Figure 4.54 Echo-planar Frame #20)

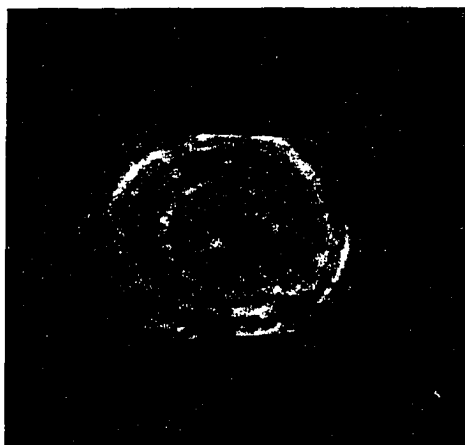


Figure 4.89 Median Echo-planar #30



Figure 4.90 Despeckle Echo-planar #30

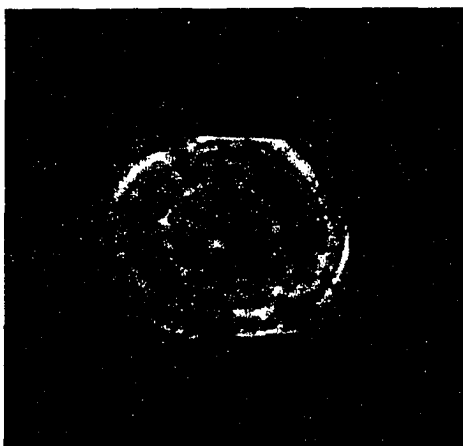


Figure 4.91 Average Echo-planar #30



(Figure 4.58 Echo-planar Frame #30)



Figure 4.92 Median Echo-planar #50

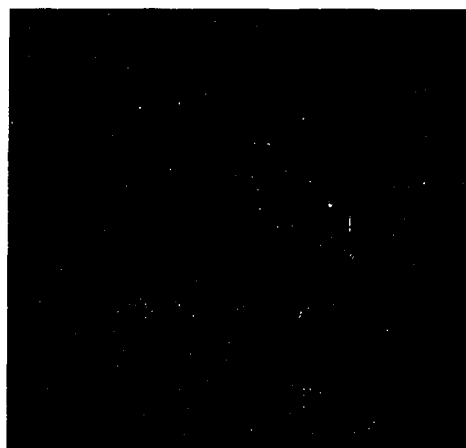


Figure 4.93 Despeckle Echo-planar #50



Figure 4.94 Average Echo-planar #50



(Figure 4.60 Echo-planar Frame #50)

Visual comparison indicates that the speckle noise is attenuated. However, the clarity and sharpness of the original image is lacking in the results from all three methods. By comparison with the unprocessed images, it can be noted that one of the effects of the general processing procedures is the tendency to smear the image. This is because there is no discrimination during the processing. In an effort to reduce noise, all pixels are smoothed whether they need it or not. Visual inspection confirms the *SNR* conclusion that the common general image processing methods do not enhance the image.

4.8 Comparison of Proposed Image Enhancement Methods on Echo-Planar Images

As the literature review in Chapter 2, Section 2.5.4 indicates, current research in image smoothing is heading toward discriminate smoothing techniques that smooth only along homogenous areas to preserve image structures. This is because general image enhancements via smoothing tend to smear image structures such as lines and borders. Because MRI images are acquired across the log, indiscriminate image smoothing on the individual image frame is equivalent to smoothing across the image structures such as growth rings. Therefore, image enhancement methods were designed to take advantage of the structure of the tree. Because adjacent frames at 10 mm apart in a log are very similar, smoothing between the same pixel locations in adjacent frames accomplishes the goal of smoothing along homogenous areas and not smoothing across image structures. Two methods are proposed: Gaussian smoothing between frames and median smoothing between frames. Because Gaussian smoothing requires the parameter, σ , to be specified and the proper σ was not known *a priori*, four proposed methods were tested: Gaussian smoothing with $\sigma = 5$ mm, Gaussian smoothing with $\sigma = 10$ mm, Gaussian smoothing with $\sigma = 15$ mm and median smoothing between frames with a window of 3.

The signal-noise ratio of the echo-planar images compared to the spin-echo images is given in Table B.9 in Appendix B along with the signal-noise ratio of the echo-planar images after the four proposed image enhancement methods were applied.

The ANOVA table for the comparison of five sets of data is given in Table 4.12. The experiment design was given in Chapter 3, Section 3.6.3. The treatments refer to the five methods of presenting the echo-planar images: unprocessed, and the four proposed

image enhancements. The blocks refer to the individual frames. The table indicates that blocking was a good idea since it captured the largest portion (.90/.95) of the variation. The first F -statistic is the ratio of the mean square of the treatments and the mean square error and is used to test whether the treatments have a significant effect on the SNR of the echo-planar images. The F -statistic of 2.19 is not significant at 5% as indicated by the P -value of 0.071. However, it would be considered significant at 10%. Thus, depending on the level of significance, we have a borderline situation where we don't have enough evidence to conclude that the choice of methods of processing echo-planar images has an effect on the quality of the image as measured by the SNR .

In Table 4.12, Tukey's Test procedure was used to investigate further. The treatment averages were arranged in order and the treatments that were not significantly different were connected with the heavy bars to indicate similarity of results. Thus, the Gaussian filtered images at all levels and the unprocessed images produced similar results. Also, the z-axis median-filter images produced similar results with the unprocessed images based upon the SNR measure (using the spin-echo image as the noise-free basis).

The overall ANOVA table did not find significant differences in all the treatments at 5% and Tukey's test did not find differences with the treatments from the unprocessed images based upon the SNR measure. Before dismissing the proposed preprocessing methods, the images were compared visually. Some of the images after the four proposed image enhancements were applied are given in Figures 4.95 through 4.106. The respective, original, unprocessed echo-planar figures, Figure 4.54, Figure 4.58 and

Figure 4.60 are also reproduced for ease of comparison. The figures for frame #20, frame #30 and frame #50 are grouped together.

Table 4. 12 ANOVA of SNR of Echo-Planar Images (Unprocessed and Proposed Processing Methods)

<i>Source of Variation</i>	<i>Sum of Squares</i>	<i>Degrees of Freedom</i>	<i>Mean Square</i>	<i>F₀</i>	<i>Significance of F</i>
<i>Treatments</i>	0.001854	4	0.000463	2.189534	0.071060481
<i>Blocks</i>	0.900839	56	0.016086	76.00929	
<i>Error</i>	0.047407	224	0.000212		
<i>Total</i>	0.950099	284			

Tukey's Test

$\bar{S}y_i$	0.001927		
$q_{.05}(4,224)$	3.64	$T_{.05}$	0.007014
$q_{.01}(4,224)$	4.42	$T_{.01}$	0.008517

<i>Treatment</i>	<i>Zmedian</i>	$\sigma = 5mm$	$\sigma = 10mm$	<i>Unprocessed</i>	$\sigma = 15mm$
<i>Treatment Average</i>	0.501702	0.497367	0.497263	0.494977	0.494501



Figure 4. 95 Gaussian ($\sigma = 5mm$) #20

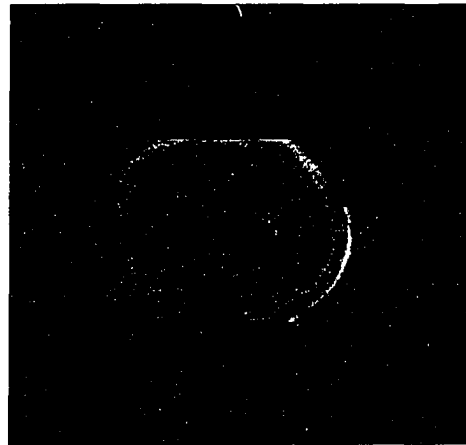


Figure 4. 96 Gaussian ($\sigma = 10mm$) #20



Figure 4.97 Gaussian ($\sigma = 15\text{mm}$) #20

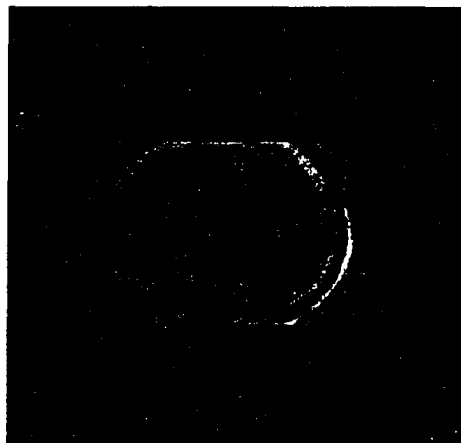
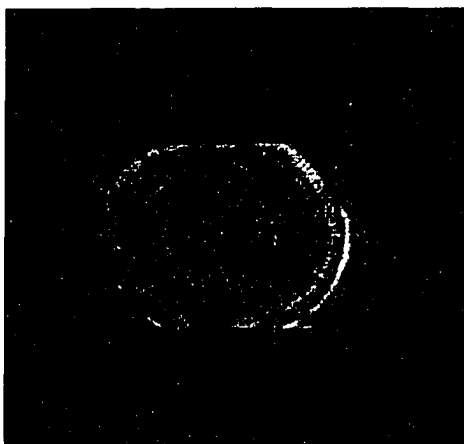


Figure 4.98 Z-axis median #20



(Figure 4.54 Echo-Planar Frame #20)



Figure 4.99 Gaussian ($\sigma = 5\text{mm}$) #30

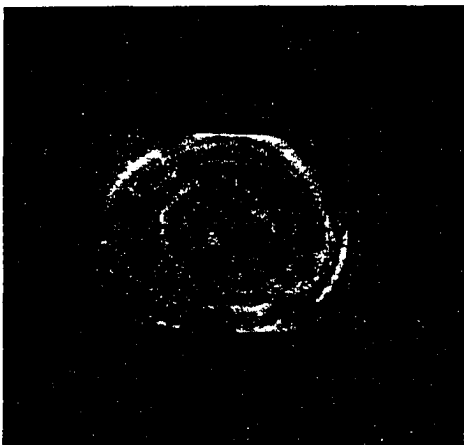


Figure 4.100 Gaussian ($\sigma = 10\text{mm}$) #30

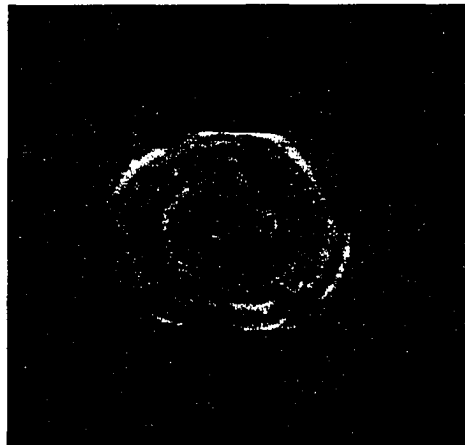


Figure 4.101 Gaussian ($\sigma = 15\text{mm}$) #30

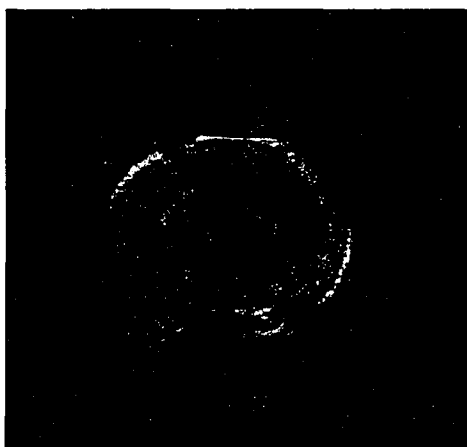


Figure 4. 102 Z-axis Median #30



(Figure 4. 58 Echo-Planar Frame #30)



Figure 4. 103 Gaussian ($\sigma = 5\text{mm}$) #50



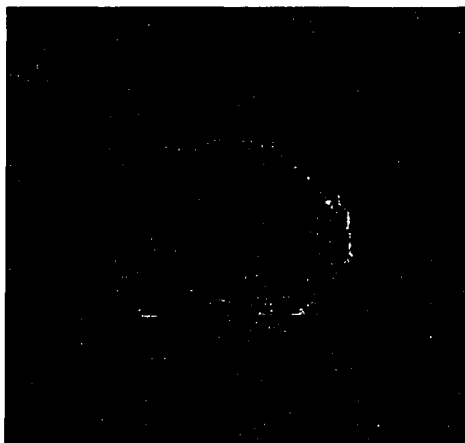
Figure 4. 104 Gaussian ($\sigma = 10\text{mm}$) #50



Figure 4. 105 Gaussian ($\sigma = 15\text{mm}$) #50



Figure 4. 106 Z-axis Median #50



(Figure 4. 60 Echo-Planar Frame #50)

Visual comparison indicates that all of the proposed process methods have succeeded in suppressing the speckle noise. The graininess of the original is also suppressed. For the Gaussian-smoothed images, the graininess becomes more suppressed as σ is increased as would be expected. There is also a slightly detectable decrease in sharpness as σ is increased, but compared to Figures 4.86 through 4.94, the decrease in sharpness is very small next to the general enhancement methods.

The sharpness of the z-axis median is indistinguishable from the original, yet, the speckle noise has been removed. Thus, visual inspection indicates, where the *SNR* analysis does not, that the proposed image smoothing methods especially the z-axis median smoothing method enhance the echo-planar images.

4.9 Comparison of Defect Detection Capability of Best Enhancement Methods on Echo-Planar Images

4.9.1 Comparison of defect images using *SNR*

Given that the proposed methods enhance the appearance of the echo-planar images, it is of interest to know if the enhancement actually improves the ability to detect defects. To determine that, the echo-planar images were processed by the best

general-purpose enhancement method (median), the z-axis Gaussian smoothing method ($\sigma = 5$ mm), and the z-axis median smoothing method.

Following the procedure in Section 3.3.2, the window size was determined by the size that maximized skewness of the residuals. Then, using the original spin-echo images for the basis, the loss-functions as described in Section 3.3.3 were calculated to determine the best parameters to use on each frame. Image statistics were gathered on each preprocessed frame. For an automatic determination of parameters, regressions were performed on each parameter against the image statistics using backward elimination as described in Section 3.3.4. A summary of these regressions is given in Tables 4.13 through 4.18.

Table 4. 13 Regressions of Echo-Planar Algorithm Upper Limit Parameters Against Image Statistics of Unprocessed and Processed Echo-Planar Images

<i>Upper Limit</i>	<i>Unprocessed</i>	<i>Zmedian</i>	<i>$\sigma=5mm$</i>	<i>XYMedian</i>
r^2	0.390532	0.437854	0.158169	0.787743
Significance of F	0.001825	0.000861	0.066927	1.44E-14
No. of Coefficients	8	9	4	6
MSE	16.96703	21.21046	36.96895	12.26555
Prediction SE	47.93235	72.3985	45.40091	57.62321

Table 4. 14 Regressions of Echo-Planar Algorithm Lower Limit Parameters Against Image Statistics of Unprocessed and Processed Echo-Planar Images

<i>Lower Limit</i>	<i>Unprocessed</i>	<i>Zmedian</i>	<i>$\sigma=5mm$</i>	<i>XYMedian</i>
r^2	0.556728	0.409894	0.56596	0.462107
Significance of F	2.19E-08	2.11E-05	2.39E-06	2.62E-05
No. of Coefficients	4	4	8	6
MSE	20.62039	23.86516	23.51092	9.869515
Prediction SE	31.70223	37.3322	78.39608	21.33506

Table 4. 15 Regressions of Echo-Planar Algorithm Upper Stop Parameters Against Image Statistics of Unprocessed and Processed Echo-Planar Images

<i>Upper Stop</i>	<i>Unprocessed</i>	<i>Zmedian</i>	$\sigma=5mm$	<i>XYMedian</i>
r^2	0.194224	0.341978	0.218673	0.500378
Significance of F	0.053484	0.008786	0.005321	0.000111
No. of Coefficients	5	8	3	9
MSE	9.329658	17.41964	10.84554	7.346771
Prediction SE	15.69356	28.61656	12.08321	29.12803

Table 4. 16 Regressions of Echo-Planar Algorithm Lower Stop Parameters Against Image Statistics of Unprocessed and Processed Echo-Planar Images

<i>Lower Stop</i>	<i>Unprocessed</i>	<i>Zmedian</i>	$\sigma=5mm$	<i>XYMedian</i>
r^2	0.46883	0.574798	0.485751	0.578619
Significance of F	5.69E-05	4.87E-07	7.91E-07	4E-07
No. of Coefficients	7	7	4	7
MSE	4.626212	4.340183	3.928111	3.144466
Prediction SE	12.67714	12.05041	6.685792	4.807313

Table 4. 17 Regressions of Echo-Planar Algorithm Pith Lower Limit Parameters Against Image Statistics of Unprocessed and Processed Echo-Planar Images

<i>Upper Limit</i>	<i>Unprocessed</i>	<i>Zmedian</i>	$\sigma=5mm$	<i>XYMedian</i>
r^2	0.616575	0.526488	0.5575	0.489348
Significance of F	5.04E-08	4.39E-07	9.05E-08	6.68E-07
No. of Coefficients	7	5	5	4
MSE	33.12494	40.14923	36.59857	33.64994
Prediction SE	65.11735	68.54833	100.714	49.21802

Table 4. 18 Regressions of Echo-Planar Algorithm Pith Lower Stop Parameters Against Image Statistics of Unprocessed and Processed Echo-Planar Images

<i>Upper Limit</i>	<i>Unprocessed</i>	<i>Zmedian</i>	$\sigma=5mm$	<i>XYMedian</i>
r^2	0.344609	0.401519	0.187064	0.307611
Significance of F	0.008175	2.44E-05	0.06345	0.000887
No. of Coefficients	8	4	5	4
MSE	5.729101	11.72937	10.73076	3.956437
Prediction SE	15.9932	11.1299	26.80584	4.572804

To obtain predicted parameters, the modified cross validation procedure, described in Section 3.3.5, was used. The plots of the correlation of image statistics versus frame distance for each type of preprocessed image is given in Figures 4.107 through 4.109.

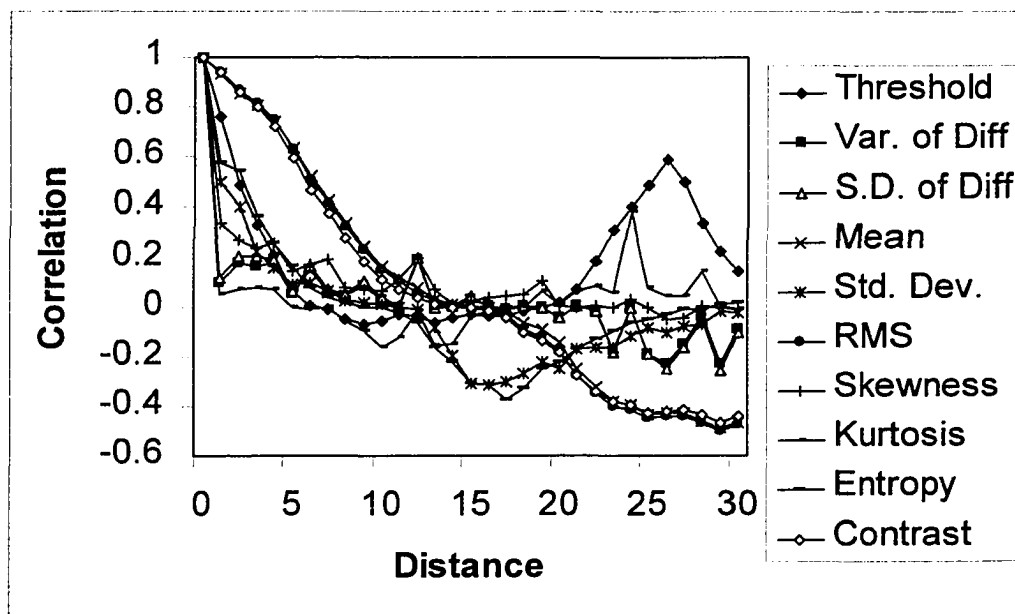


Figure 4. 107 Correlation of Image Statistics Between Zmedian Frames

After obtaining the predicted parameters, the processed images were then sent through the echo-planar defect detection algorithm. Then the defect images from the three enhancements and the defect images of the unprocessed echo-planar images were all compared to the defect images produced from the spin-echo method to determine the *SNR* for each processing method. The results are given in Table B.9 in Appendix B.

The ANOVA table for the comparison of four sets of data is given in Table 4.19. The experiment design was given in Chapter 3, Section 3.6.3. The treatments refer to the four methods of presenting the echo-planar defect images: unprocessed, and the three proposed image preprocessing enhancements. The blocks refer to the individual frames.

The table indicates that blocking was a good idea since it captured a large portion (1.56/4.04) of the variation. The first F -statistic is the ratio of the mean square of the treatments and the mean square error and is used to test whether the treatments have a significant effect on the SNR of the echo-planar images. The F -statistic of 2.68 is barely significant as indicated by the P -value of 0.049. Because of the barely significant level, Tukey's test was used to determine if any two of the treatment mean differences were significant. No pair was significantly different. So, the conclusion is that the choice of methods of preprocessing the echo-planar images before sending them through the defect detection algorithm does not affect the quality of the defect detection as measured by the SNR .

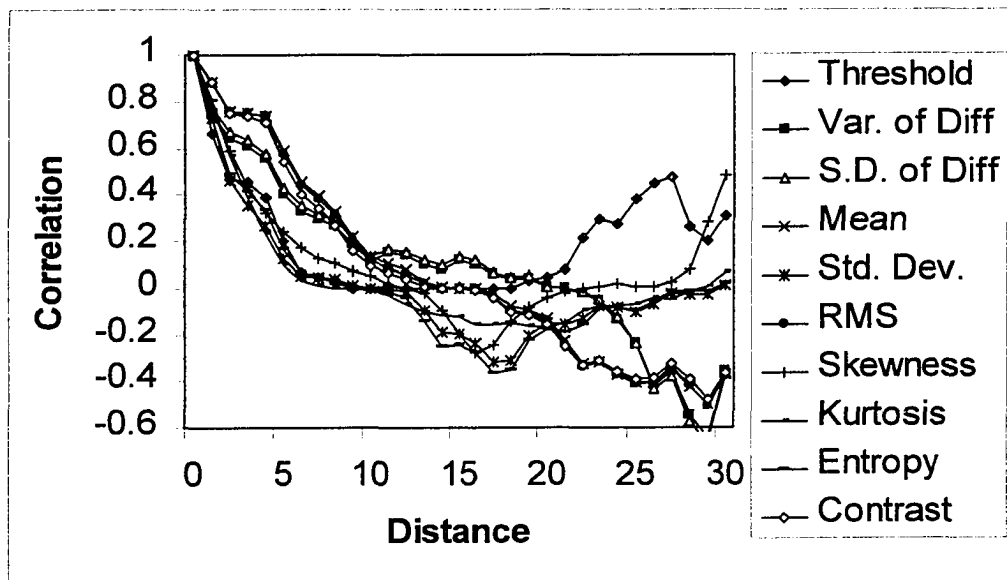


Figure 4.108 Correlation of Image Statistics Between Gaussian Frames

Finally, because SNR is only one indicator of the effectiveness of the defect region recognition ability, other statistics on the defect regions from the four methods were gathered. They were the number of undetected defect pixels, the number of undetected

defect regions, the number of false alarm pixels, and the number of false alarm defect regions.

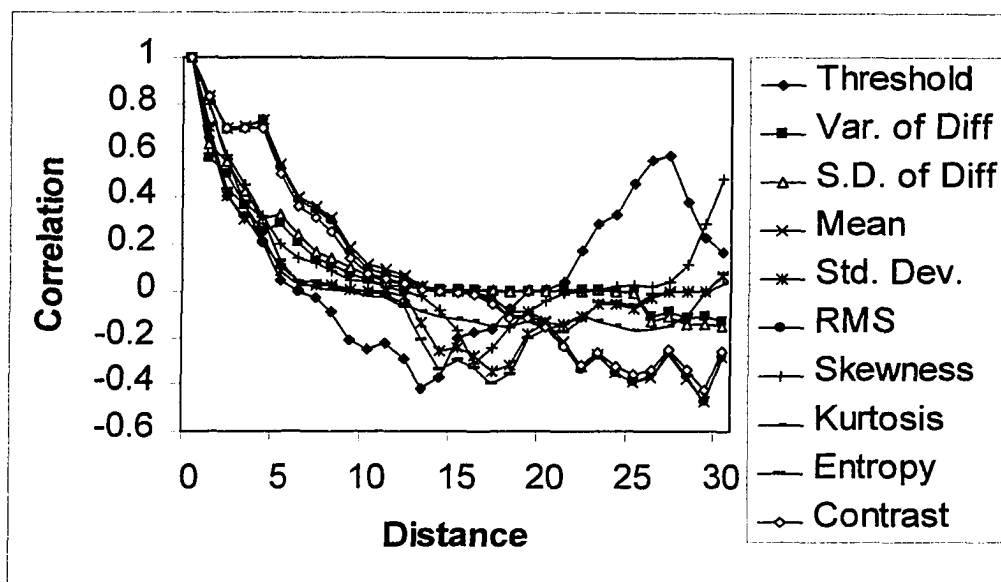


Figure 4.109 Correlation of Image Statistics Between XY Median Frames

Table 4.19 ANOVA of SNR of Defect Images from Unprocessed and Processed Echo-Planar Images

Source of Variation	Sum of Squares	Degrees of Freedom	Mean Square	F ₀	Significance of F
Treatments	0.117262	3	0.039087	2.681613	0.048655
Blocks	1.559021	54	0.028871	1.9807	0.000553
Error	2.361318	162	0.014576		
Total	4.037601	219			

Tukey's Test

$S\bar{y}_i$	0.016279		
$q_{.05}(4,162)$	3.66	$T_{.05}$	0.059583
$q_{.01}(4,162)$	4.45	$T_{.01}$	0.072443

Treatment	XYMedian	Zmedian	$\sigma = 5mm$	Unprocessed
Treatment Average	1.540315	1.541977	1.576017	1.594722

4.9.2 Compare defect images using undetected defect pixels

The results of the undetected defect pixels are given in Table B.11 in Appendix B. The ANOVA table for the comparison of the four preprocessing methods based on the measure of the number of undetected defect pixels is given in Table 4.20.

The F -statistic of 2.18 is not significant at 5% as indicated by the P -value of 0.093. In Table 4.20, Tukey's Test procedure also verified that no treatments were significantly different. So, the conclusion is that the choice of methods of preprocessing the echo-planar images before sending them through the defect detection algorithm does not affect the quality of the defect detection as measured by the number of undetected defect pixels.

Table 4. 20 ANOVA of Undetected Defect Pixels from Defect Images of Unprocessed and Preprocessed Echo-Planar Images

<i>Source of Variation</i>	<i>Sum of Squares</i>	<i>Degrees of Freedom</i>	<i>Mean Square</i>	F_0	<i>Significance of F</i>
<i>Treatments</i>	93894.8	3	31298.27	2.175101	0.092996
<i>Blocks</i>	8404719.3	54	155643	10.81655	
<i>Error</i>	2331072.7	162	14389.34		
<i>Total</i>	10829686	219			

Tukey's Test

$S\bar{y}_i$	16.17480		
$q_{.05}(4,162)$	3.66	$T_{.05}$	59.19979
$q_{.01}(4,162)$	4.45	$T_{.01}$	71.97788

<i>Treatment</i>	<i>Zmedian</i>	<i>XYMedian</i>	$\sigma = 5mm$	<i>Unprocessed</i>
Treatment Average	808.255	824.1273	855.7818	856.1273

4.9.3 Compare defect images using undetected defect regions

The results of the undetected defect regions are given in Table B.12 in Appendix B. The ANOVA table for the comparison of the four sets of the four preprocessing

methods based on the measure of the number of undetected defect regions is given in Table 4.21. The F -statistic of 0.752 is not significant at all as indicated by the P -value of 0.52. Thus, the conclusion is that the choice of methods of preprocessing echo-planar images has no effect on the defect detection ability as measured by the number of undetected defect regions.

Table 4. 21 ANOVA of Undetected Defect Regions from Defect Images of Unprocessed and Preprocessed Echo-Planar Images

<i>Source of Variation</i>	<i>Sum of Squares</i>	<i>Degrees of Freedom</i>	<i>Mean Square</i>	F_0	<i>Significance of F</i>
<i>Treatments</i>	1.509091	3	0.50303	0.751734	0.522841
<i>Blocks</i>	79.22727	54	1.467172	2.192556	8.35E-05
<i>Error</i>	108.404	162	0.66916		
<i>Total</i>	189.1404	219			

Tukey's Test

$S\bar{y}_i$	0.110302		
q.05(4,162)	3.66	T.05	0.403706
q.01(4,162)	4.45	T.01	0.490844

<i>Treatment</i>	<i>Zmedian</i>	<i>XYMedian</i>	$\sigma=5mm$	<i>Unprocessed</i>
<i>Treatment Average</i>	1.236364	1.236364	1.381818	1.418182

4.9.4 Compare defect images using false alarm pixels

The results of the false alarm pixel count is given in Table B.13 in Appendix B. The ANOVA table for the comparison of the four preprocessing methods based on the measure of the number of false alarm pixels is given in Table 4.22. The F -statistic of 3.10 is significant at 5% as indicated by the P -value of 0.028.

In Table 4.22, Tukey's Test procedure was used to investigate further. The treatment averages were arranged in order and the treatments that were not significantly different were connected with the heavy bars to indicate similarity of results. Thus, we

can conclude that preprocessing with a median filter along the x-y plane (within a frame) actually causes an increase in false alarm pixels compared to no preprocessing.

Table 4. 22 ANOVA of False Alarm Defect Pixels from Defect Images of Unprocessed and Processed Echo-Planar Images

<i>Source of Variation</i>	<i>Sum of Squares</i>	<i>Degrees of Freedom</i>	<i>Mean Square</i>	<i>F₀</i>	<i>Significance of F</i>
<i>Treatments</i>	7809738.6	3	2603246	3.099126	0.028399
<i>Blocks</i>	54805453	54	1014916	1.208242	
<i>Error</i>	136078961	162	839993.6		
<i>Total</i>	198694153	219			

Tukey's Test

$S\bar{y}_i$	123.5824		
$q_{.05}(4,162)$	3.66	$T_{.05}$	452.3116
$q_{.01}(4,162)$	4.45	$T_{.01}$	549.9417

<i>Treatment</i>	<i>Unprocessed $\sigma = 5mm$</i>	<i>Zmedian</i>	<i>XYMedian</i>
<i>Treatment Average</i>	737.96364	819.7091	1004.618
			1227.673

4.9.5 Compare defect images using false alarm regions

The results of the false alarm defect regions count is given in Table B.14 in Appendix B. The ANOVA table for the comparison of the four preprocessing methods based on the measure of the number of false alarm regions is given in Table 4.23. The *F*-statistic of 4.94 is significant as indicated by the *P*-value of 0.0026. Thus, the conclusion is that the choice of methods of preprocessing echo-planar images has an effect on the defect detection ability as measured by the number of false defect regions.

In Table 4.23, Tukey's Test procedure was used to investigate further. The treatment averages were arranged in order and the treatments that were not significantly different were connected with the heavy bars to indicate similarity of results. Both

median preprocessing treatments gave better results than no preprocessing based upon the number of false alarm regions.

Table 4. 23 ANOVA of False Alarm Defect Regions from Defect Images of Unprocessed and Preprocessed Echo-Planar Images

<i>Source of Variation</i>	<i>Sum of Squares</i>	<i>Degrees of Freedom</i>	<i>Mean Square</i>	<i>F₀</i>	<i>Significance of F</i>
<i>Treatments</i>	68.18182	3	22.72727	4.939928	0.002616
<i>Blocks</i>	480.6091	54	8.900168	1.934512	0.000827
<i>Error</i>	745.3182	162	4.60073		
<i>Total</i>	1294.109	219			

Tukey's Test

$\bar{S}\bar{y}_i$	0.289222		
$q_{.05}(4,162)$	3.66	$T_{.05}$	1.058554
$q_{.01}(4,162)$	4.45	$T_{.01}$	1.28704

<i>Treatment</i>	<i>Zmedian</i>	<i>XYMedian</i>	$\sigma = 5\text{ mm}$	<i>Unprocessed</i>
<i>Treatment Average</i>	1.545455	1.672727	2.690909	2.745455

The next table, Table 4.24, summarizes the defect detection losses that occur when echo-planar images are used versus spin-echo images. Table 4.24 indicates that on average just over 1.23 defect regions are missed and about 1.54 false alarm defect regions are generated when Z-axis median smoothing method is applied.

Table 4. 24 Defect Detection Losses by Preprocessing on Echo-Planar Images

<i>Undetected Pixels (% of Foreground Pixels)</i>				
<i>Frame</i>	<i>Unprocessed</i>	$\sigma = 5\text{ mm}$	<i>ZMedian</i>	<i>XYMedian</i>
<i>Average</i>	5.1614	5.1579	4.8693	4.9657
<i>lower 95%</i>	4.8291	4.7994	4.4628	4.5015
<i>upper 95%</i>	5.4937	5.5163	5.2759	5.4300
<i>Undetected Regions</i>				
<i>Frame</i>	<i>Unprocessed</i>	$\sigma = 5\text{ mm}$	<i>ZMedian</i>	<i>XYMedian</i>
<i>Average</i>	1.4181	1.3818	1.2363	1.2363

table continued on next page

<i>Frame</i>	<i>Unprocessed</i>	$\sigma = 5 \text{ mm}$	<i>ZMedian</i>	<i>XYMedian</i>
<i>lower 95%</i>	1.1888	1.0857	0.9690	0.9199
<i>upper 95%</i>	1.6475	1.6779	1.5036	1.5527
<i>False Alarm Pixels (% of Foreground Pixels)</i>				
<i>Frame</i>	<i>Unprocessed</i>	$\sigma = 5 \text{ mm}$	<i>ZMedian</i>	<i>XYMedian</i>
<i>Average</i>	4.4373	4.9289	6.0703	7.4915
<i>lower 95%</i>	3.7484	4.1283	5.1629	4.1850
<i>upper 95%</i>	5.1262	5.7295	6.9777	10.7981
<i>False Alarm Regions</i>				
<i>Frame</i>	<i>Unprocessed</i>	$\sigma = 5 \text{ mm}$	<i>ZMedian</i>	<i>XYMedian</i>
<i>Average</i>	2.7454	2.6909	1.5454	1.6727
<i>lower 95%</i>	1.8988	1.8138	1.0788	0.9722
<i>upper 95%</i>	3.5920	3.5679	2.0120	2.3731

4.10 Visual Comparison of Defect Images

Examples of the defect images after the three image enhancements are applied (Z-axis median, Z-axis Gaussian smoothing with $\sigma = 5 \text{ mm}$, and ordinary median smoothing) are given in Figures 4.110 through 4.134. Also, the respective defect image from the unprocessed echo-planar image and the original spin-echo image are given again to allow visual comparison of the defect detection ability of each preprocessing method.

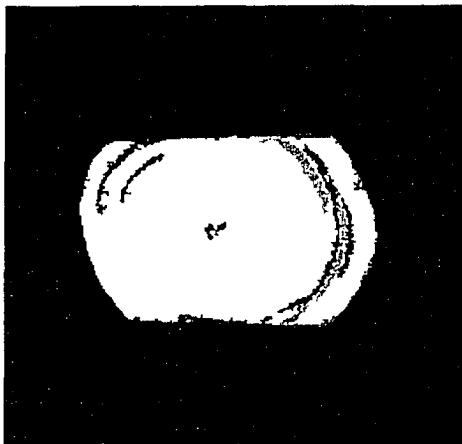


Figure 4. 110 Z-axis Median Defects #15

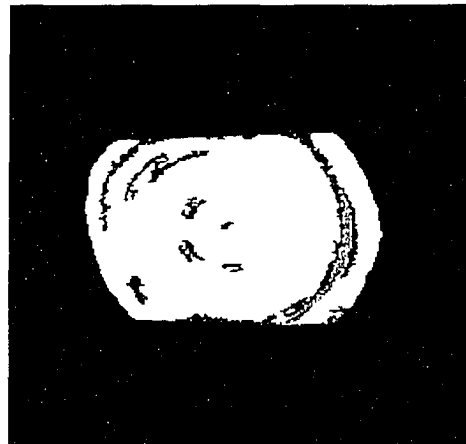


Figure 4. 111 Gaussian Defects #15

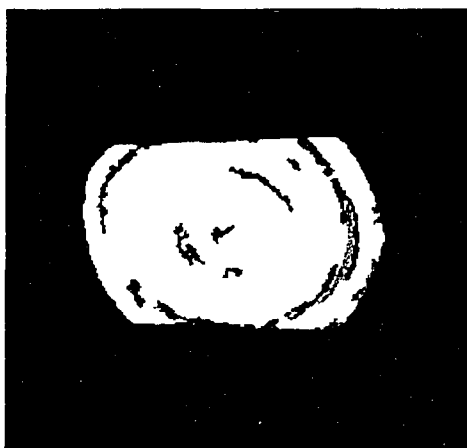


Figure 4.112 XYMedian Defects #15

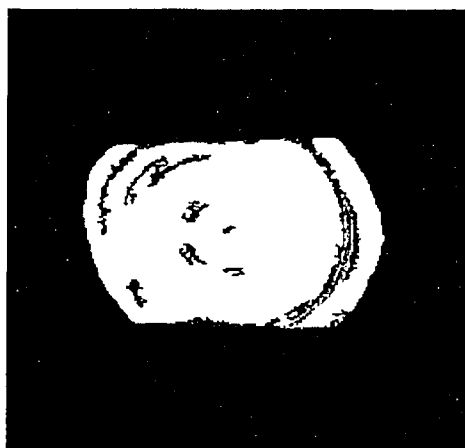


Figure 4.113 Unprocessed Defects #15

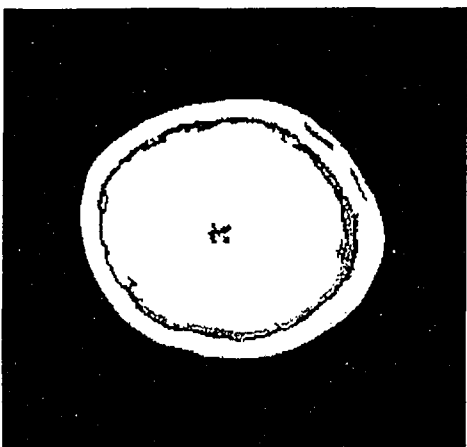


Figure 4.114 Spin-Echo Defects #15



(Figure 4.1 Spin-Echo Frame #15)



Figure 4.115 Z-axis Median Defects #20

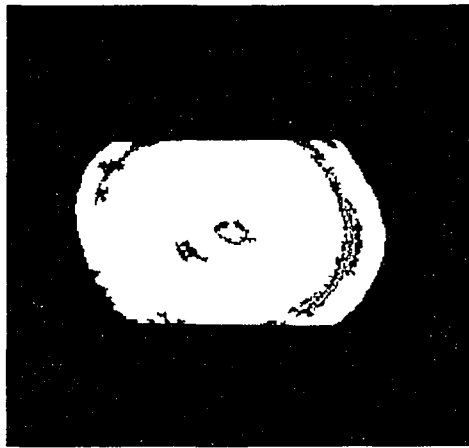


Figure 4.116 Gaussian Defects #20

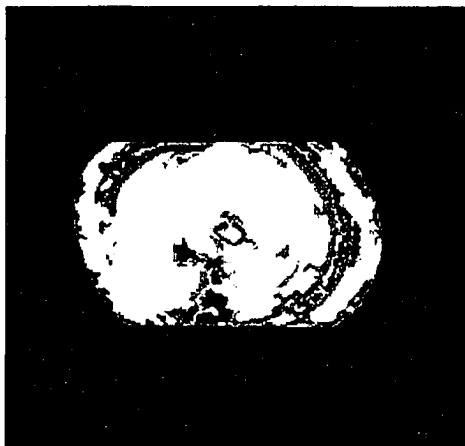


Figure 4.117 XYMedian Defects #20

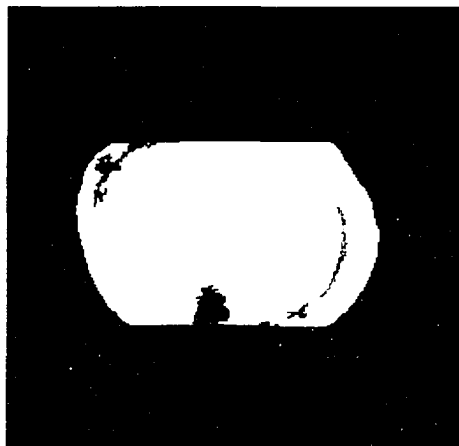


Figure 4.118 Unprocessed Defects #20

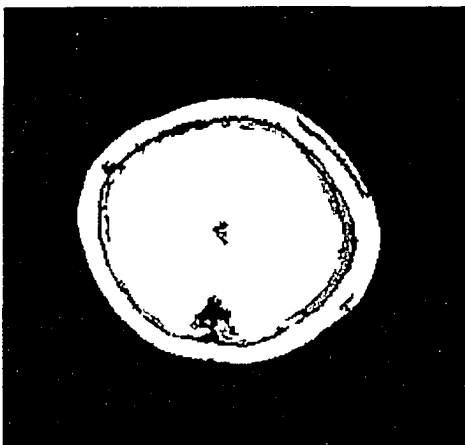


Figure 4.119 Spin-Echo Defects #20



(Figure 4.3 Spin-Echo Frame #20)



Figure 4.120 Z-axis Median Defects #25

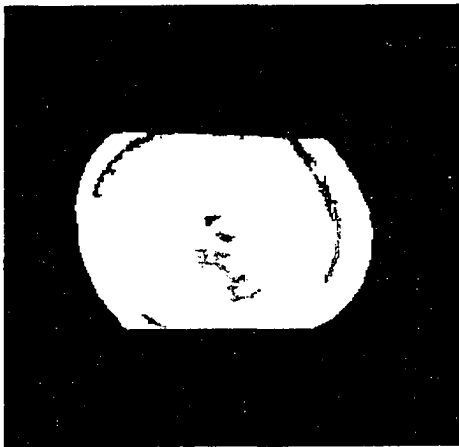


Figure 4.121 Gaussian Defects #25

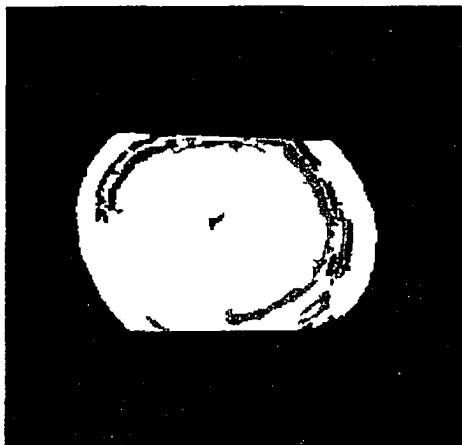


Figure 4.122 XYMedian Defects #25

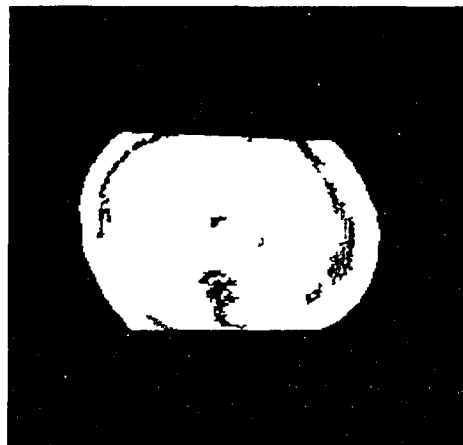


Figure 4.123 Unprocessed Defects #25

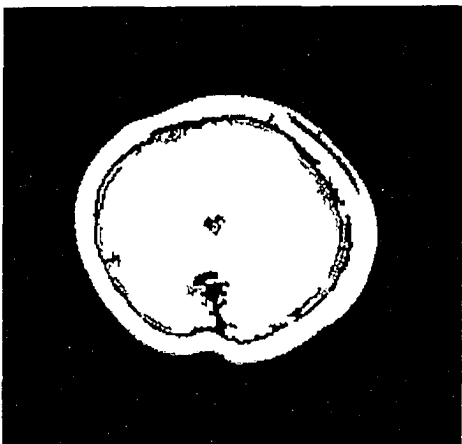


Figure 4.124 Spin-Echo Defects #25



(Figure 4.5 Spin-Echo Frame #25)

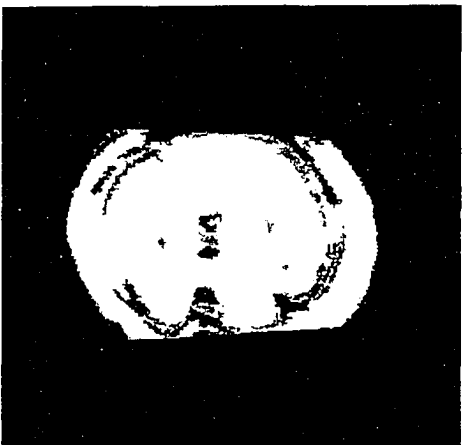


Figure 4.125 Z-axis Median Defects #30

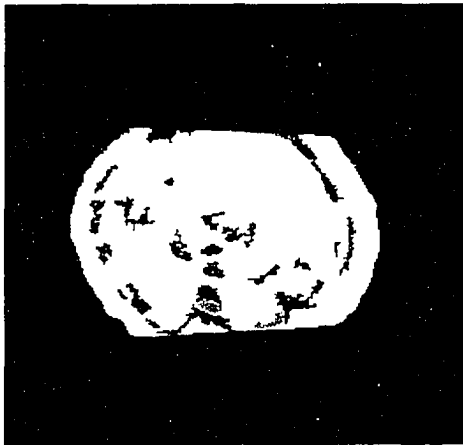


Figure 4.126 Gaussian Defects #30

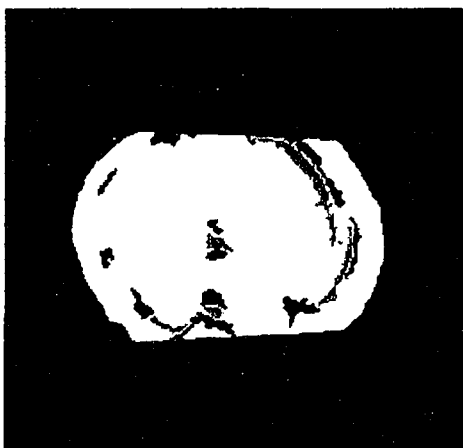


Figure 4. 127 XYMedian Defects #30

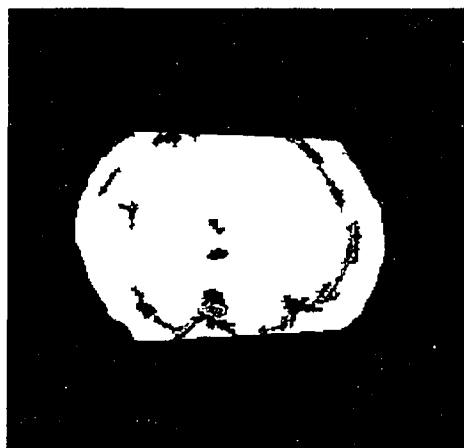


Figure 4. 128 Unprocessed Defects #30

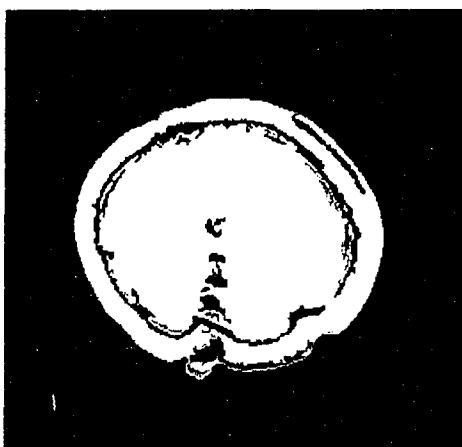


Figure 4. 129 Spin-Echo Defects #30



(Figure 4. 7 Spin-Echo Defects #30)

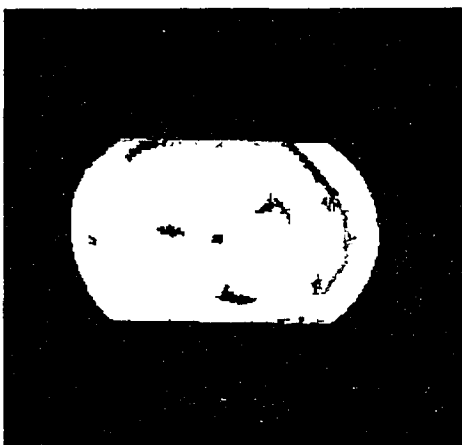


Figure 4. 130 Z-axis Median #50



Figure 4. 131 Gaussian Defects #50

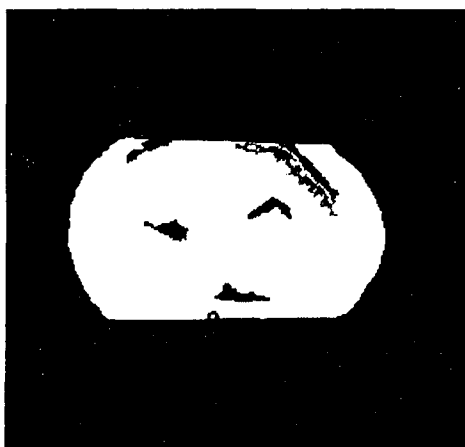


Figure 4.132 XYMedian #50



Figure 4.133 Unprocessed Defects #50

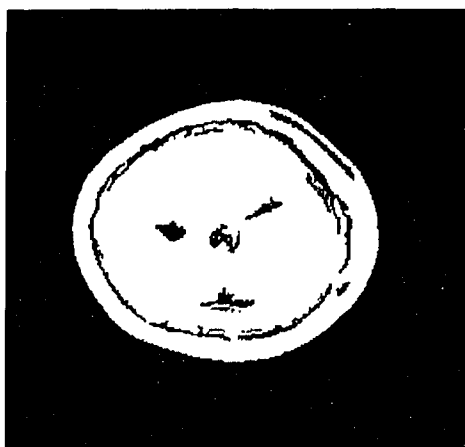
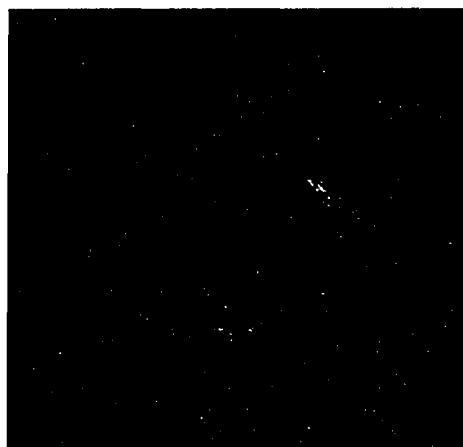


Figure 4.134 Spin-Echo Defects #50



(Figure 4.9 Spin-Echo Frame #50)

Visual comparison of the defect images confirms the results in Table 4.24 that indicate that about 1.23 defects per frame goes undetected. That missing defect, for most of the 55 frames, happened to be the scar tissue near the outer edge of the log at the 1 to 2 o'clock positions. Also, although the main ring defect located near the heartwood sapwood border is detected in places, it is generally not picked up in the 7 to 9 o'clock positions in most frames. Figure 4.117 demonstrates what happens when all the parameters are predicted badly at the same time.

Otherwise, in the remaining images, the major defects including the knots and the pith are usually detected. Since the z-axis median preprocessing step improves the appearance of the echo-planar images and has yields improvement in defect detection ability as far as fewer false alarms are concerned, it would be the recommended approach when substituting the quick echo-planar imaging for the slow spin-echo imaging method.

CHAPTER 5

CONCLUSIONS AND FUTURE RESEARCH

5.1 Conclusions

The data for the research consisted of 55 useful MRI scans of a black oak log taken across the longitudinal axis 10 mm apart using the spin-echo method and the echo-planar method. Black oak (*Quercus velutina*) is a species of the red oak family which is the most commercially important group of hardwoods as shown in Table 1.3 (Chang, 1992). Of the following defects used by the NHLA in grading lumber: bark pockets, checks (cracks), decay, grub holes, holes, knots, mineral streak, pith, shake, splits, stain, wane, and worm holes; this study has examined decay, knots, pith, and shake. Knots are by far the most numerous and the most important grading defect (Harding *et al.*, 1993). Many checks are produced as a result of a subsequent lumber drying operation after the sawing has been completed and wane is also a defect produced in the sawing stage. This leaves bark pockets, grub holes, holes, mineral streak, splits, and stain defects that have not been included in this study.

The major contributions of this research are given below:

- (1) Developed an algorithm that identifies internal defects from MRI spin-echo images with greater border accuracy than thresholding methods. The steps are outlined below:
 - Determine a threshold for the image using the moment-preserving threshold method. Using the threshold and 4-connectivity, begin in a corner of the image and grow the background region.

- Pass the spin-echo image through an average filter with various window sizes to determine the underlying gray levels of the foreground.
 - Choose the average filter window size that maximizes the skewness of the residuals between the original image and the averaged image.
 - Subtract the average-filtered image from the original image to obtain the pure variation.
 - Compute image statistics on the original image foreground.
 - Use the image statistics to predict the four algorithm parameters (upper normal variation limit, lower normal variation limit, upper region stopping limit, lower region stopping limit) based upon the prediction equations from the regression analysis.
 - Pass the difference image through a threshold operation. All pixels above the upper normal variation limit are labeled as “light” wet defect seeds. All pixels below the lower normal variation limit are labeled as “dark” dry defect seeds.
 - Using the difference image, grow the defect seeds using 4-connectivity as long as the wet-defect region gray levels are above the upper region-stopping limit and the dry-region gray levels are below the lower region-stopping limit.
- (2) Developed an algorithm that identifies virtually the same set of internal defects from MRI echo-planar images that are found from the spin-echo images. The steps are outlined below:
- Preprocess the echo planar images with “z-axis” median smoothing. Use a preceding frame and a subsequent frame for a window size of 3.

- Determine the threshold percentile for the image using the moment-preserving threshold method. Subtract 0.03 (or 3%) from the moment-preserving threshold percentile. Use this percentile to generate an adjusted threshold. Using the adjusted threshold and 4-connectivity, begin in a corner of the image and grow the background region.
- Using a binary image to represent the background and the foreground, use morphological smoothing (closing) and a ball element to smooth the border.
- Pass the echo-planar image through an average filter of various window sizes to determine the underlying gray levels of the foreground.
- Choose the window size that maximizes the skewness of the residuals between the original image and the averaged imaged.
- Subtract the average-filtered image from the original image to obtain the pure variation.
- Compute image statistics on the original image foreground.
- Use the image statistics to predict the six algorithm parameters (upper normal-variation limit, lower normal-variation limit, upper region-stopping limit, lower region-stopping limit, pith-area lower normal-variation limit, and pith-area lower region-stopping limit) based upon the prediction equations from the regression analysis.
- Pass the difference image through a threshold operation. All pixels above the upper normal-variation limit are labeled as wet defect seeds. All pixels below the lower normal-variation limit are labeled as dry defect seeds.

- Using the difference image, grow the defect seeds using 4-connectivity pixel aggregation as long as the wet-defect region gray-levels are above the upper region-stopping limit and the dry-region defect gray levels are below the lower region-stopping limit.
- Find the geometric center of the foreground.
- Search in the 20 x 20 pixel central area of the foreground for detected defect pixels. If pixels were already detected, then stop.
- Otherwise, pass the 20x 20 central area through a threshold operation. All pixels below the pith-area lower normal-variation limit are labeled as dry defect seeds.
- Using the difference image, grow the pith area defect seed(s) using 4-connectivity pixel aggregation as long as the dry-region defect gray levels are below the pith-area lower region-stopping limit.

(3) Evaluated the defect detection capability of the two MRI methods. These include the following comparisons:

- Visual comparison of the defect images confirms that about 1.23 defects per frame goes undetected. That missing defect, for most of the 55 frames, happened to be the scar tissue near the outer edge of the log at the 1 to 2 o'clock positions.
- Figure 4.117 demonstrates what happens when all the parameters are predicted badly at the same time.
- The main ring defect located near the heartwood sapwood border is detected in places, it is generally not picked up in the 7 to 9 o'clock positions in most frames.
- The remaining major defects including the knots and the pith are usually detected.

5.2 Other Conclusions

The proposed algorithms were based on these other conclusions:

For spin-echo images:

- (1) The moment-preserving thresholding method with a connectivity modification does an excellent job of separating the log foreground from the air background.
- (2) The significance of all the regression equations indicates that the algorithm parameters must be adjusted for each frame.
- (3) Image statistics such as the mean, standard deviation, RMS, skewness, kurtosis, entropy and contrast obtained from the foreground can be used to predict the defect detection parameters for each individual image.
- (4) The spin-echo defect detection algorithm can obtain defect regions with high accuracy.
- (5) Ordinary thresholding can not obtain the defect regions to the same degree without generating a large number of false alarm defect regions. This is because the gray levels of pixels in the defect regions often overlap with the gray levels of pixels in the clear wood areas.
- (6) Many current research efforts are being put into more sophisticated methods of determining the proper threshold to isolate defects in logs. Yet any threshold is a compromise of undetected defect pixels and false alarms. These results indicate that superior defect detection from logs can be obtained using the simultaneous dual consideration of adjacency relationships and gray level and not just gray level alone.

For echo-planar images:

- (1) As with the spin-echo images, the moment-preserving thresholding method with a connectivity modification does an excellent job of separating the log foreground from the air background. A small amount of morphological border smoothing is required for best results.
- (2) The significance of all the regression equations indicates that the algorithm parameters must be adjusted for each frame.
- (3) Image statistics such as the mean, standard deviation, RMS, skewness, kurtosis, entropy and contrast obtained from the foreground can be used to predict the defect detection parameters for each individual image although with less accuracy than achieved with spin-echo images.
- (4) ANOVA analysis of the signal-noise ratio indicates that the common general image processing methods do not enhance echo-planar images.
- (5) Visual inspection indicates further that attempting to enhance echo-planar images using general indiscriminant smoothing methods will attenuate speckle noise. However, the clarity and sharpness of the original image is sharply reduced because the lack of discrimination allows the process to smooth vital image structures as well as noise.
- (6) By taking advantage of the structure of a tree, image enhancement methods were designed to smooth between the same pixel locations in adjacent frames to accomplish the goal of smoothing along homogenous areas and not smoothing across image structures. Two methods were proposed: Gaussian smoothing between frames and median smoothing between frames (z-axis median smoothing).

- (7) Based upon the *SNR* measure, preprocessing by smoothing along the z-axis (between frames) did not significantly improve the images.
- (8) The sharpness of the z-axis median-filtered images are indistinguishable from the originals, yet, the speckle noise has been removed. Visual inspection indicates that the proposed image smoothing methods especially the z-axis median smoothing method enhance the echo-planar images visually.
- (9) The z-axis median-filtered echo-planar images yield significantly less false alarm regions than unprocessed echo-planar images.

5.3 Improving the Research

There are three main needs for further data and these should definitely be considered when further study is done on this topic. They include:

- (1) obtaining more sample logs to provide an external validation set for the algorithm to check the reliability of the internal cross validation estimate methods,
- (2) obtaining more sample logs to validate the algorithm across different species, and
- (3) obtaining more sample logs that include the remaining grading defects that have not yet been studied.

Also, when there is more money available for further research on this topic, the priority for spending should be in the following areas in order of importance:

- (1) the collection of MRI scans of more sample logs. The need for that was just explained.
- (2) the improvement of the regression method. Currently, the method of regression of the parameters is implicitly assuming that the parameters are independent. Also, the regression is seeking to minimize the squared difference of the error of the

parameter value itself. Although it would require a substantial amount of programming time, the regressions for all parameters should be run simultaneously and seek to minimize the loss functions (Section 3.3.3) associated with the parameter values rather than the deviations of the parameter values themselves. This is because the loss functions give a better indication of the quality of the defect detection than just the deviation of the parameter value

5.4 Further Research

Areas for further research could include:

- (1) three-dimensional region growing approach for defect segmentation,
- (2) finding image descriptors for prediction of the algorithm parameters that contain spatial information rather than just gray level statistical summaries,
- (3) automatic compensation methods to unwarp the images obtained by the echo-planar imaging method,
- (4) examining the relationship between the MRI acquisition time and the quality of the defect detection,
- (5) testing the basic spin-echo defect detection algorithm approach on images acquired by X-ray computer-aided tomography (or CAT scans),
- (6) the determination of the optimal cutting plan based upon the now-obtainable defect information and
- (7) economic analysis of using the echo-planar method for acquiring MRI images for hardwood log defect detection.

Since a log is a three-dimensional structure, a three-dimensional approach to the defect segmentation would be the next logical area to seek improvement. The individual frames

would be stacked to model the log. Consideration has to be given both during the 3-D model construction and the defect segmentation steps that the data along the log axis (Z-axis) would be intermittently recorded unlike the continuous recording along the x-y planes.

The algorithm parameters were predicted using image characteristics. These characteristics were basic gray level statistical summaries that did not convey spatial information about the image. There is still a need to capture the certain essence of an image that determines how a parameter should be set to best segment the defect regions. Low-dimensional measures that convey spatial information are required to allow that regression of those measures to the best algorithm parameters for each image.

The echo-planar images used in this research were unwarped manually to register with the spin-echo images. Automatic methods would be required to use the echo-planar imaging process in industry to achieve production speed. Since the warping is primarily in the Y-direction and optical scanners are now capable of providing the outside dimensions of the log. The unwarping process represents an achievable goal.

In the current study, we have two acquisition methods: spin-echo that is slow (about 20 seconds per slice) and echo-planar that is fast (about 1/2 second per slice). It would be useful to look at acquisition speeds between these times to determine where the defect detection quality drops off.

Currently, X-ray CAT scans represent the fastest acquisition method. However, as explained in Chapter 2, Section 2.3.1, the density of knots is very similar to the density of sapwood making the detection of knots in that area difficult. One difficulty could stem from the current methods of defect detection that use only gray level information as

the determinant of defects. The spin-echo defect detection algorithm uses spatial information as well as gray levels to determine defect regions. The improvement for CAT scans could be dramatic.

The determination of the optimal cutting plan is a logical follow-up to the results of this research. As noted in Chapter 2, Section 2.4.3, the grade method of sawing refers to the method of always sawing planks from the best looking side of a log cant. The term, tomographic grade sawing, then refers to the current grade method of sawing a log but using the internal defect information rather than relying solely on the external appearance of the log cant. The cutting plan could possibly be solved as a 3-dimensional bin-packing problem. However, these approaches all have the appearance of the combinatorial optimization problems that traditionally require enumeration techniques. Therefore, quick methods of producing optimal or near-optimal plans would be needed.

Because the prices of MRI units and lumber are volatile and this study has made more information available about the defect detection capability of echo-planar imaging, a new economic analysis is needed.

There are quite a number of research directions that could be followed from the current research that have a practical application in the hardwood industry.

BIBLIOGRAPHY

- Abutaleb, A.S., 1989, Automatic Thresholding Of Grey-Level Pictures Using Two-Dimensional Entropy, *Computer Vision, Graphics, And Image Processing*, **47**, pp. 22-32.
- Ahn, C.B., Kim, J.H., and Cho, Z.H., 1986, High-Speed Spiral-Scan Echo Planar NMR Imaging--I, *IEEE Transactions on Medical Imaging*, **5**(1), pp. 2-7.
- Andrews, H.C., and Hunt, B.R., 1977, *Digital Image Restoration* (Englewood Cliffs, New Jersey: Prentice-Hall, Inc.)
- Åstrand, E., and Rönnqvist, M., 1994, Crosscut Optimization Of Boards Given Complete Defect Information, *Forest Products Journal*, **44**(2), pp. 15-24.
- Ayache, N., 1995, Medical Computer Vision, Virtual Reality and Robotics, *Image and Vision Computing*, **13**(4), pp. 295-298*.
- Bendel, P., 1985, Echo Projection Imaging - A Method To Obtain NMR Images Undistorted By Magnetic Field Inhomogeneities, *IEEE Transactions on Medical Imaging*, **4**(2), pp. 114-119.
- Birkeland, R., and Han, W., 1991, Ultrasonic Scanning For Internal Log Defects, *4th International Conference On Scanning Technology In Sawmilling*, Oakland, CA.
- Birkeland, R., and Holoyen, S., 1987, Industrial Methods For Internal Scanning Of Log Defects: A Progress Report On An Ongoing Project In Norway, *2nd International Conference On Scanning Technology In Sawmilling*, Oakland, CA, No. X.
- Blackman, T., 1995a, French Company Operates Full Wood Processing Plant, *Wood Technology*, May, 1995, pp.30-31.
- Blackman, T., 1995b, Automatic Scanner-Optimizer Helps Mill Dissect Lumber, *Wood Technology*, August, 1995, pp.18-19.
- Breiman, L., Friedman, J.H., Olshen, R.A., and Stone, C.J., 1984, *Classification and Regression Trees* (Belmont, California: Wadsworth Inc.)
- Brunner, C.C., Butler, D.A., Maristany, A.G., and VanLeeuwen, D., 1990, Optimal Clear-Area Sawing Patterns For Furniture And Millwork Blanks, *Forest Products Journal*, **40**(3), pp. 51-56.

- Burman, P., 1989, A Comparative Study Of Ordinary Cross Validation, v-fold Cross Validation And The Repeated Learning-testing methods, *Biometrika*, **76**(3), pp. 503-514.
- Butler, D.A., Brunner, C.C., and Funck, J.W., 1989, A Dual-Threshold Image Sweep-And-Mark Algorithm For Defect Detection In Veneer, *Forest Products Journal*, **39**(5), pp. 25-28.
- Butler, D.A., Funck, J.W., and Brunner, C.C., 1993, An Adaptive Image Preprocessing Algorithm For Defect Detection In Douglas-Fir Veneer, *Forest Products Journal*, **43**(5), pp. 57-60.
- Carnieri, C., Mendoza, G.A., and Luppold, W.G., 1993, Optimal Cutting Of Dimension Parts From Lumber With A Defect: A Heuristic Solution Procedure, *Forest Products Journal*, **43**(9), pp. 66-72.
- Carnieri, C., Mendoza, G.A., and Gavinho, L., 1994, Optimal Cutting Of Lumber And Particle Boards Into Dimension Parts: Some Algorithms And Solution Procedures, *Wood and Fiber Science*, **26**(1), pp. 131-141.
- Chang, S.J., 1989, Economic Feasibility Analysis Of Fast NMR Imaging Scanner, *3rd International Conference On Scanning Technology In Sawmilling*, Oakland, CA, No. VII.
- Chang, S.J., 1992, *External and Internal Defect Detection To Optimize Cutting Of Hardwood Logs And Lumber*, National Agricultural Library, Transferring Technologies For Industry; No. 3, Beltsville, MD.
- Chang, S.J., Cohen, M., and Wang, P.C., 1991, Ultra-fast Scanning Of Hardwood Logs With An NMR Scanner, *4th International Conference On Scanning Technology In Sawmilling*, Oakland, CA.
- Chang, S.J., and Guddanti, S., 1993, Application Of High Speed Image Processing In Hardwood Sawing Research, *5th International Conference On Scanning Technology In Sawmilling*, Oakland, CA, No. V.
- Chang, S.J., Guddanti, S., and Cooper, C.S., 1997, Measuring the Benefits of Internal Log Defect Scanning: A Mill-Based Study, *7th International Conference On Scanning Technology & Process Optimization For the Wood Products Industry*, Charlotte, NC.
- Chang, S.J., Olson, J.R., and Wang, P.C., 1989, NMR Imaging Of Internal Features In Wood, *Forest Products Journal*, **39**(6), pp. 43-49.
- Chang, S.J., Wang, P.C., and Olson, J.R., 1987, Nuclear Magnetic Resonance Imaging Of Hardwood Logs, *2nd International Conference On Scanning Technology In Sawmilling*, Oakland, CA, No. IX.

Chen, H., Li, A., Kaufman, L., and Hale, J., 1994, A Fast Filtering Algorithm For Image Enhancement, *IEEE Transactions on Medical Imaging*, **13**(3), pp. 557-564.

Cheng, F., and Venetsanopoulos, A.N., 1992, An Adaptive Morphological Filter For Image Processing, *IEEE Transactions On Image Processing*, **1**(4), pp. 533-539.

Chiang, J.Y., and Sullivan, B.J., 1993, Coincident Bit Counting -- A New Criterion For Image Registration, *IEEE Transactions on Medical Imaging*, **12**(1), pp. 30-38.

Cocklin, L., Gourlay, A.R., Jackson, P.H., Kaye, G., Kerr, I.H., and Lams, P., 1983, Digital Processing Of Chest Radiographs, *Image And Vision Computing*, **1**(2), pp. 67-78.

Conners, R.W., Cho, T., and Araman, P.A., 1989, Automated Grading Of Rough Hardwood Lumber, *3rd International Conference On Scanning Technology In Sawmilling*, Oakland, CA, No. XVI.

Conners, R.W., Klinkhachorn, P., McMillin, C.W., Franklin, J.P., and Ng, C., 1987, A Computer Vision System For Grading Hardwood Lumber, *2nd International Conference On Scanning Technology In Sawmilling*, Oakland, CA, No. XV.

Crooks, L.E., Kaufman, L., Hoenninger III, J., Arakawa, M., Watts, J., and Cannon, C.R., 1984, Spatial Resolution In NMR Imaging, *IEEE Transactions on Medical Imaging*, **3**(2), pp. 51-53.

Dale-Jones, R., and Tjahjadi, T., 1992, Four Algorithms For Enhancing Images With Large Peaks In Their Histogram, *Image And Vision Computing*, **10**(7), pp.495-507.

Davies, E.R., 1988, On the Noise Suppression And Image Enhancement Characteristics Of the Median, Truncated Median And Mode Filters, *Pattern Recognition Letters*, **7**(2), pp.87-97.

Deklerck, R., Cornelis, J., and Bister, M., 1993, Segmentation Of Medical Images, *Image and Vision Computing*, **11**(8), pp. 486-503.

Deng, G., Cahill, L.W., and Tobin, G.R., 1995, The Study Of Logarithmic Image Processing Model And Its Application To Image Enhancement, *IEEE Transactions On Image Processing*, **4**(4), pp. 506-512.

Deruyver, A., Hodé, Y., and Soufflet, L., 1994, A Segmentation Technique For Cerebral NMR Images, *1994 1st IEEE International Conference on Image Processing*, Proceedings ICIP94, Volume 3 of 3, IEEE Computer Society Press, Los Alamitos, CA, pp. 716-720.

- Flibotte, S., Menon, R.S., MacKay, A.L., and Hailey, J.R.T., 1990, Proton Magnetic Resonance Of Western Red Cedar, *Wood and Fiber Science*, **22**(4), pp. 362-376.
- Forrer, J.B., Butler, D.A., Funck, J.W., and Brunner, C.C., 1988, Image Sweep-And-Mark Algorithms. Part 1. Basic Algorithms, *Forest Products Journal*, **38**(11/12), pp. 75-79.
- Forrer, J.B., Butler, D.A., Brunner, C.C., and Funck, J.W., 1989, Image Sweep-And-Mark Algorithms. Part 2. Performance Evaluations, *Forest Products Journal*, **39**(1), pp. 39-42.
- Fuller, J.J., Ross, R.J., and Damm, J.R., 1995, Nondestructive Evaluation Of Honeycomb And Surface Checks In Red Oak Lumber, *Forest Products Journal*, **45**(5), pp. 42-44.
- Funck, J.W., Butler, D.A., Brunner, C.C., Shaw, G.B., Forrer, J.B., Maristany, A.G., and Pierson, P.K., 1987, Evaluation Of Image Analysis Routines For Defect Scanning On Wood Surfaces, *2nd International Conference On Scanning Technology In Sawmilling*, Oakland, CA, No. XIII.
- Funt, B.V., and Bryant, E.C., 1985, A Computer Vision System That Analyses CT-Scans Of Sawlogs, *Proceedings of 1985 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, San Francisco, CA, pp. 175-177.
- Funt, B.V., and Bryant, E.C., 1987, Detection Of Internal Log Defects By Automatic Interpretation Of Computer Tomography Images, *Forest Products Journal*, **37**(1), pp. 56-62.
- Gonzalez, R.C. and Woods, R.E., 1992, *Digital Image Processing* (Reading, Massachusetts: Addison-Wesley Publishing Company)
- Grundberg, S., and Gronlund, A., 1991, Methods For Reducing Data When Scanning For Internal Log Defects, *4th International Conference On Scanning Technology In Sawmilling*, Oakland, CA.
- Guddanti, S., and Chang, S.J., 1998, Replicating Sawmill Sawing With TOPSAW Using CT Images of a Full-Length Hardwood Log, *Forest Products Journal*, **48**(1), pp. 72-75.
- Hagman, O., and Grundberg, S., 1993, Multivariate Image Analysis Methods To Classify Features On Scots Pine - Evaluation Of A Multisensor Approach, *5th International Conference On Scanning Technology In Sawmilling*, Oakland, CA, No. VII.

Hailey, J.R.T., and Swanson, J.S., 1987, Imaging Wood Using Magnetic Resonance, *2nd International Conference On Scanning Technology In Sawmilling*, Oakland, CA, No. VIII.

Hall, L.D., and Rajanayagam, R., 1986, Evaluation Of The Distribution Of Water In Wood By Use Of Three Dimensional Proton NMR Volume Imaging, *Wood Science and Technology*, **20**, pp. 329-333.

Hanke, R.F., Hassler, U., and Heil, K., 1994, Fast Automatic X-Ray Image Processing By Means Of A New Multistage Filter For Background Modeling, *1994 1st IEEE International Conference on Image Processing*, Proceedings ICIP94, Volume 1 of 3, IEEE Computer Society Press, Los Alamitos, CA, pp. 392-396.

Hannah, I., Patel, D., and Davies, R., 1995, The Use Of Variance and Entropic Thresholding Methods For Image Segmentation, *Pattern Recognition*, **28**(8), pp. 1135-1143.

Harding, O.V., Steele, P.H., and Nordin, K., 1993, Description Of Defects By Type For Six Grades Of Red Oak Lumber, *Forest Products Journal*, **43**(6), pp. 45-50.

Harless, T.E.G., Wagner, F.G., Steele, P.H., Taylor, F.W., Yadama, V., and McMillin, C.W., 1991, Methodology For Locating Defects Within Hardwood Logs And Determining Their Impact On Lumber-Value Yield, *Forest Products Journal*, **41**(4), pp. 25-30.

Haygreen, J.G., and Bowyer, J.L., 1989, *Forest Products And Wood Science An Introduction* (Ames, Iowa: Iowa State University Press)

Herbin, M., Venot, A., Devaux, J.Y., Walter, E., Lebruchec, J.F., Dubertret, L., and Roucayrol, J.C., 1989, Automatic Registration Of Dissimilar Images: Application To Medical Imagery, *Computer Vision, Graphics, And Image Processing*, **47**, pp. 77-88.

Hodges, D.G., Anderson, W.C., and McMillin, C.W., 1990, The Economic Potential Of CT Scanners For Hardwood Sawmills, *Forest Products Journal*, **40**(3), pp. 65-69.

Huber, H.A., McMillin, C.W., and McKinney, J.P., 1985, Lumber Defect Detection Abilities Of Furniture Rough Mill Employees, *Forest Products Journal*, **35**(11/12), pp. 79-82.

Huber, H.A., Ruddell, S., Mukherjee, K., and McMillin, C.W., 1989, Economics Of Cutting Hardwood Dimension Parts With An Automated System, *Forest Products Journal*, **39**(5), pp. 46-50.

Hunt, D.J., Nolte, L.W., Reibman, A.R., and Ruedger, W.H., 1990, Hough Transform And Signal Detection Theory Performance For Images With Additive Noise, *Computer Vision, Graphics, And Image Processing*, **52**, pp. 386-401.

Hussian, I., and Reed, T.R., 1994, Segmentation-Based Nonlinear Image Smoothing, *1994 1st IEEE International Conference on Image Processing*, Proceedings ICIP94, Volume 2 of 3, IEEE Computer Society Press, Los Alamitos, CA, pp. 507-511.

Hwang, H., and Haddad, R.A., 1995, Adaptive Median Filters: New Algorithms And Results, *IEEE Transactions On Image Processing*, **4**(4), pp. 499-502.

Itagaki, H., 1993, Improvements Of Nuclear Magnetic Resonance Image Quality Using Iterations Of Adaptive Nonlinear Filtering, *IEEE Transactions On Medical Imaging*, **12**(2), pp.322-327.

Iwaoka, H., Sugiyama, T., Matsuura, H., and Fujino, K., 1984, A New Pulse Sequence For "Fast Recovery" Fast-Scan NMR Imaging, *IEEE Transactions on Medical Imaging*, **3**(1), pp. 41-46.

Karaman, M., Kutay, M.A., and Bozdagi, G., 1995, An Adaptive Speckle Suppression Filter For Medical Ultrasonic Imaging, *IEEE Transactions On Medical Imaging*, **14**(2), pp. 283-292.

Kass, M., and Witkin, A., 1987, Analyzing Oriented Patterns, *Computer Vision, Graphics, And Image Processing*, **37**, pp. 362-385.

Kaufman, L., Crooks, L.E., and Margulis, A.R., 1981, *Nuclear Magnetic Resonance Imaging In Medicine* (Tokyo: Igaku-Shion Ltd.)

Kennedy, D.N., Filipek, P.A., and Caviness, Jr., V.S., 1989, Anatomic Segmentation And Volumetric Calculations In Nuclear Magnetic Resonance Imaging, *IEEE Transactions on Medical Imaging*, **8**(1), pp. 1-7.

Kimmel, J.D., and Janowiak, J.J., 1995, Red Maple And Yellow-Poplar LVL From Ultrasonically Rated Veneer, *Forest Products Journal*, **45**(7/8), pp. 54-58.

Kittler, J., Illingworth, J., and Föglein, J., 1985, Threshold Selection Based On A Simple Image Statistic, *Computer Vision, Graphics, And Image Processing*, **30**, pp. 125-147.

Klinkhachorn, P., Franklin, J.P., McMillin, C.W., Conners, R.W., and Huber, H.A., 1988, Automated Computer Grading Of Hardwood Lumber, *Forest Products Journal*, **38**(3), pp. 67-69.

Klinkhachorn, P., Franklin, J.P., McMillin, C.W., and Huber, H.A., 1989a, ALPS: Yield Optimization Cutting Program, *Forest Products Journal*, **39**(3), pp. 53-56.

- Klinkhachorn, P., Schwehm, C.J., McMillin, C.W., and Huber, H.A., 1989b, HaLT: A Computerized Training Program For Hardwood Lumber Graders, *Forest Products Journal*, **39**(2), pp. 38-40.
- Kline, D.E., Conners, R.W., Schmoldt, D.L., Araman, P.A., and Brisbin, R.L., 1993, A Multiple Sensor Machine Vision System For Automatic Hardwood Feature Detection, *5th International Conference On Scanning Technology In Sawmilling*, Oakland, CA, No. VI.
- Knoll, T.F., and Delp, E.J., 1986, Adaptive Gray Scale Mapping To Reduce Registration Noise In Difference Images, *Computer Vision, Graphics, And Image Processing*, **33**, pp. 129-137.
- Koivo, A.J., and Kim, C.W., 1989, Automatic Classification Of Surface Defects On Red Oak Boards, *Forest Products Journal*, **39**(9), pp. 22-30.
- Kundu, A., and Zhou, J., 1992, Combination Median Filter, *IEEE Transactions On Image Processing*, **1**(3), pp. 422-429.
- Lee, S.C., Qian, G., Chen, J., Sun, Y., and Hay, D.A., 1991, Determining A Maximum Value Yield Of A Log Using An Optical Log Scanner, *Proceedings of 1991 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Lahaina, Maui, Hawaii, pp. 747-748.
- Lee, S.U., Chung, S.Y., and Park, R.H., 1990, A Comparative Performance Study Of Several Global Thresholding Techniques For Segmentation, *Computer Vision, Graphics, And Image Processing*, **52**, pp. 171-190.
- Leszczynski, K.W., and Shalev, S., 1989, A Robust Algorithm For Contrast Enhancement By Local Histogram Modification, *Image And Vision Computing*, **7**(3), pp. 205-209.
- Li, J.S.J., and Ramsingh, A., 1995, The Relationship Of The Multi-Shell To Multistage And Standard Median Filters, *IEEE Transactions On Image Processing*, **4**(8), pp. 1165-1169.
- Lin, W., Kline, D.E., Araman, P.A., and Wiedenbeck, J.K., 1995, Design And Evaluation Of Log-To-Dimension Manufacturing Systems Using System Simulation, *Forest Products Journal*, **45**(3), pp. 37-44.
- Lindgren, L.O., 1991a, Medical CAT-scanning: X-ray Absorption Coefficients, CT-numbers And Their Relation To Wood Density, *Wood Science and Technology*, **25**, pp. 341-349.

- Lindgren, L.O., 1991b, The Accuracy Of Medical CAT-scan Images For Non-destructive Density Measurements In Small Volume Elements Within Solid Wood, *Wood Science and Technology*, **25**, pp. 425-432.
- Maness, T.C., and Donald, W.S., 1994, The Effect Of Log Rotation On Value Recovery In Chip And Saw Sawmills, *Wood and Fiber Science*, **26**(4), pp. 546-555.
- Manos, G., Cairns, A.Y., Ricketts, I.W., and Sinclair, D., 1993, Automatic Segmentation Of Hand-Wrist Radiographs, *Image and Vision Computing*, **11**(2), pp. 100-111.
- Mansfield, P., 1977, Multi-planar Image Formation Using NMR Spin Echoes, *Journal of Physics C*, **10**, pp. 155-158.
- Martin, P., Collet, R., Barthelemy, P., and Roussy, G., 1987, Evaluation Of Wood Characteristics: Internal Scanning Of The Material By Microwaves, *Wood Science and Technology*, **21**, pp. 361-371.
- Matsopoulos, G.K., Marshall, S., Brunt, J.N.H., 1994, Multiresolution Morphological Fusion Of MR And CT Images Of The Human Brain, *Vision, Image and Signal Processing*, **141**(1), pp. 137-142.
- McMillin, C.W., 1984, Evaluating Wood Failure In Plywood Shear By Optical Image Analysis, *Forest Products Journal*, **34**(7/8), pp. 67-69.
- McMillin, C.W., Conners, R.W., and Huber, H.A., 1984, ALPS - A Potential New Automated Lumber Processing System, *Forest Products Journal*, **34**(1), pp. 13-20.
- Montgomery, D.C., 1997, *Design and Analysis of Experiments*, 4th Edition (New York: JohnWiley and Sons)
- Moore, C.J., 1986, Medical Image Processing: The Characterization Of Display Changes Using Histogram Entropy, *Image And Vision Computing*, **4**(4), pp. 197-202.
- Myers, R.H., 1990, *Classical and Modern Regression with Applications*, 2nd Edition (Boston: PWS-KENT Publishing Company)
- Myler, H.R., and Weeks, A.R., 1993, *Computer Imaging Recipes In C* (Englewood Cliffs, New Jersey: Prentice-Hall, Inc.)
- Neter, J., Kutner, M., Nachtsheim, C., and Wasserman, W., 1996, *Applied Linear Statistical Models*, 4th Edition (Boston: Richard D. Irwin, Inc.)
- Occeña, L.G., and Tanchoco, J.M.A., 1988, Computer Graphics Simulation Of Hardwood Log Sawing, *Forest Products Journal*, **38**(10), pp. 72-76.

- Peters, R.A., 1995, A New Algorithm For Image Noise Reduction Using Mathematical Morphology, *IEEE Transactions On Medical Imaging*, 4(5), pp. 554-568.
- Pettersen, R.C., Ward, J.C., and Lawrence, A.H., 1993, Detection Of Northern Red Oak Wetwood By Fast Heating And Ion Mobility Spectrometric Analysis, *Holzforschung*, 47(6), pp. 513-522.
- Philip, K.P., Dove, E.L., McPherson, D.D., Gotteiner, N.L., Stanford, W., and Chandran, K.B., 1994, The Fuzzy Hough Transform -- Feature Extraction In Medical Images, *IEEE Transactions on Medical Imaging*, 13(2), pp. 235-240.
- Pizer, S.M., Amburn, E.P., Austin, J.D., Cromartie, R., Geselowitz, A., Greer, T., Romeny, B.T.H., Zimmerman, J.B., and Zuiderveld, K., 1987, Adaptive Histogram Equalization And Its Variations, *Computer Vision, Graphics, And Image Processing*, 39, pp. 355-368.
- Quick, J.J., Hailey, J.R.T., and MacKay, A.L., 1990, Radial Moisture Profiles Of Cedar Sapwood During Drying: A Proton Magnetic Resonance Study, *Wood and Fiber Science*, 22(4), pp. 404-412.
- Rank, K., and Unbehauen, R., 1992, An Adaptive Recursive 2-D Filter For Removal Of Gaussian Noise In Images, *IEEE Transactions On Image Processing*, 1(3), pp. 431-436.
- Raya, S.P., 1990, Low-Level Segmentation Of 3-D Magnetic Resonance Brain Images -- A Rule-Based System, *IEEE Transactions on Medical Imaging*, 9(3), pp. 327-337.
- Richards, D.B., 1980, Lumber Values From Computerized Simulation Of Hardwood Sawing, *Res. Pap. FPL-356*. USDA Forest Serv., Forest Prod. Lab., Madison, Wis.
- Robb, Richard A., 1985a, *Three-Dimensional Biomedical Imaging Volume I* (Boca Raton, Florida: CRC Press, Inc.)
- Robb, Richard A., 1985b, *Three-Dimensional Biomedical Imaging Volume II* (Boca Raton, Florida: CRC Press, Inc.)
- Roder, F.L., Scheinman, E., and Magnuson, P., 1989, High-Speed CT Scanning Of Logs, *3rd International Conference On Scanning Technology In Sawmilling*, Oakland, CA, No. VI.
- Ross, R.J., Ward, J.C., and TenWolde, A., 1994, Stress Wave Nondestructive Evaluation Of Wetwood, *Forest Products Journal*, 44(7/8), pp. 79-83.
- Russo, F., and Ramponi, G., 1995, A Fuzzy Operator For The Enhancement Of Blurred And Noisy Images, *IEEE Transactions On Image Processing*, 4(8), pp. 1169-1174.

- Sahoo, P.K., Soltani, S., Wong, A.K.C., and Chen, Y.C., 1988, A Survey Of Thresholding Techniques, *Computer Vision, Graphics, And Image Processing*, **41**, pp. 233-260.
- Schwehm, C.J., Klinkhachorn, P., McMillin, C.W., and Huber, H.A., 1990, HaRem: Hardwood Lumber Remanufacturing Program For Maximizing Value Based On Size, Grade, And Current Market Prices, *Forest Products Journal*, **40**(7/8), pp. 27-30.
- Scollar, I., Weidner, B., and Huang, T.S., 1984, Image Enhancement Using The Median And The Interquartile Distance, *Computer Vision, Graphics, And Image Processing*, **25**, pp. 236-251.
- Sim, H., Govett, R.L., and Morris, J.S., 1991a, Linear Programming Model For The Conversion Of Small Hardwood Logs Into Furniture Shorts. Part 1. Conventional Basic Models And Effects Of Demand Constraints, *Forest Products Journal*, **41**(9), pp. 19-24.
- Sim, H., Govett, R.L., and Morris, J.S., 1991b, Linear Programming Model For The Conversion Of Small Hardwood Logs Into Furniture Shorts. Part 2. Probabilistic Extension Of The Basic Linear Programming Models, *Forest Products Journal*, **41**(10), pp. 76-80.
- Smith, P.M., Haas, M.P., and Luppold, W.G., 1995, An Analysis Of Tropical Hardwood Product Importation And Consumption In The United States, *Forest Products Journal*, **45**(4), pp. 31-37.
- Sochurek, Howard, 1988, *Medicine's New Vision* (Easton, Pennsylvania: Mack Publishing Company)
- Soltanian-Zadeh, H., Windham, J.P., and Yagle, A.E., 1995, A Multidimensional Nonlinear Edge-Preserving Filter For Magnetic Resonance Image Restoration, *IEEE Transactions On Image Processing*, **4**(2), pp. 147-161.
- Steele, P.H., Harless, T.E.G., Wagner, F.G., Kumar, L., and Taylor, F.W., 1994, Increased Lumber Value Form Optimum Orientation Of Internal Defects With Respect To Sawing Pattern In Hardwood Sawlogs, *Forest Products Journal*, **44**(3), pp. 69-72.
- Steele, P.H., Neal, S.C., McDonald, K.A., and Cramer, S.M., 1991, The Slope-Of-Grain Indicator For Detect Detection In Unplaned Hardwood Lumber, *Forest Products Journal*, **41**(1), pp. 15-20.
- Szymani, R., 1987, Summary Remarks On The 2nd International Conference On Scanning Technology In Sawmilling, *2nd International Conference On Scanning Technology In Sawmilling*, Oakland, CA, Summary.

Taylor, F.W., Wagner, F.G., McMillin, C.W., Morgan, I.L., and Hopkins, F.F., 1984, Locating Knots By Industrial Tomography-A Feasibility Study, *Forest Products Journal*, **34**(5), pp. 42-46.

Thomas, J.G., Peters II, R.A., and Jeanty, P., 1991, Automatic Segmentation Of Ultrasound Images Using Morphological Operators, *IEEE Transactions on Medical Imaging*, **10**(2), pp. 180-186.

Toulson, D.L., and Boyce, J.F., 1992, Segmentation of MR Images Using Neural Nets, *Image And Vision Computing*, **10**(5), pp. 324-328.

Tsai, W., 1985, Moment-Preserving Thresholding: A New Approach, *Computer Vision, Graphics, and Image Processing*, **29**, pp. 377-393.

Van den Elsen, P., Maintz, J.B., Pol, E.D., and Viergever, M.A., 1995, Automatic Registration Of CT and MR Brain Images Using Correlation Of Geometrical Features, *IEEE Transactions On Medical Imaging*, **14**(2), pp. 384-396.

Vincent, L., 1993, Morphological Grayscale Reconstruction In Image Analysis: Applications and Efficient Algorithms, *IEEE Transactions On Image Processing*, **2**(2), pp. 176-201.

Wagner, F.G., Taylor, F.W., Ladd, D.S., McMillin, C.W., and Roder, F.L., 1989a, Ultrafast CT Scanning Of An Oak Log For Internal Defects, *Forest Products Journal*, **39**(11/12), pp. 62-64.

Wagner, F.G., Taylor, F.W., Steele, P.H., and Harless, T.E.G., 1989b, Benefit Of Internal Log Scanning (Preliminary Results), *3rd International Conference On Scanning Technology In Sawmilling*, Oakland, CA, No. V.

Wang, D.C.C., Vagnucci, A.H., and Li, C.C., 1983, Digital Image Enhancement: A Survey, *Computer Vision, Graphics and Image Processing*, **24**, pp. 363-381.

Wang, P.C., and Chang, S.J., 1986, Nuclear Magnetic Resonance Imaging Of Wood, *Wood and Fiber Science*, **18**(2), pp. 308-314.

Wang, Y. and Lei, T., 1994, Statistical Analysis Of MR Imaging And Its Applications In Image Modeling, *1994 1st IEEE International Conference on Image Processing*, Proceedings ICIP94, Volume 1 of 3, IEEE Computer Society Press, Los Alamitos, CA, pp. 866-870.

Zhang, P., 1993, Model Selection Via Multifold Cross Validation, *The Annals Of Statistics*, **21**(1), pp. 299-313.

Zhu, D., Conners, R., Lamb, F., and Araman, P., 1991, A Computer Vision System For Locating And Identifying Internal Log Defects Using CT Imagery, *4th International Conference On Scanning Technology In Sawmilling*, Oakland, CA.

APPENDIX A - PROGRAM SOURCE CODE LISTINGS

The programs are listed in alphabetical order by the name of the routine.

```

/*****
*   FUNCTION NAME : apply_thresh
*   PURPOSE : separate the image into background and foreground regions
*   FUNCTIONS CALLED : none
*   INPUT EXPECTED : threshold, pointers to input image structure and output
*                   image structure
*   OUTPUT EXPECTED : image structure containing connected background and
*                   foreground regions
*   ALGORITHM : one forward scan and one backward scan is used to find the
*               4-connected regions.
*   REMARKS : no nested loops are used to improve program efficiency
*
*****/
#include <stdio.h>
#include "structure.h"

#define L 256 /* L is the possible number of gray levels */

void apply_thresh( float *threshold, struct int_image *inp_img, struct
                  int_image *out_img )
{
    int Nx, Ny, i, total_pixels, *inp_offset, *out_offset;

    inp_offset = inp_img->array;
    out_offset = out_img->array;
    Nx = out_img->Nx = inp_img->Nx;
    Ny = out_img->Ny = inp_img->Ny;
    total_pixels = Nx*Ny;

    /* Forward sweep */
    *(out_offset) = 0; /* Set first pixel to background level */
    for(i=1; i<Nx; ++i) { /* Check pixels on first row */
        if((*inp_offset + i) < *threshold) && !(*out_offset + i -1))
            *(out_offset + i) = 0;
        else
            *(out_offset + i) = 255;
    }

    for(i=Nx; i<total_pixels; ++i) { /* Check remaining pixels. */
        if((*inp_offset + i) < *threshold) && !(*out_offset + i -1) &&
            !(*out_offset + i - Nx))
            *(out_offset + i) = 0;
        else
            *(out_offset + i) = 255;
    }

    /* Backward sweep */
    /* Set last pixel to background */
    *(out_offset + total_pixels -1) = 0;
    /* Check pixels on last row */
    for(i=total_pixels-2; i>total_pixels-1-Nx; --i)
        if(*out_offset + i && (*inp_offset + i) < *threshold) &&
            !(*out_offset + i +1))
            *(out_offset + i) = 0;

    /* Check all preceeding rows. */
    for(i=total_pixels-1-Nx; i>0; --i)
        if(*out_offset+i && (*inp_offset + i) < *threshold) &&
            !(*out_offset + i +1) && !(*out_offset + i + Nx))
            *(out_offset + i) = 0;
}

```

```

/*****
FUNCTION NAME : calc_loss
* PURPOSE : calculate loss function
* FUNCTIONS CALLED : none
* INPUT EXPECTED : pointers to the input and good defect images, and
*                  weight factor
* OUTPUT EXPECTED : loss function values for upper and lower clip
* ALGORITHM : Using a single pass every pixel of the input image is
*              compared to the good defect image. Correctly marked seeds,
*              false alarms, and total defect areas from the good image
*              are accumulated. The loss function is then calculated based
*              upon the weight.
* REMARKS :
*****/
#include <stdio.h>
#include "structure.h"
#include "constants.h"

void calc_loss ( struct int_image *out_img, struct int_image *good_img,
                 float *upperloss, float *lowerloss, float *weight)
{
    int i, Nx, Ny, total_pixels, *good_offset, *out_offset;
    int lightdefect, darkdefect, lightseed, darkseed, lightfalse, darkfalse;

    good_offset = good_img->array; /* Find pointers to images */
    out_offset = out_img->array;

    Nx = good_img->Nx; /* Set output image dimensions to */
    Ny = good_img->Ny; /* match the input image. */

    total_pixels = Nx*Ny; /* Compute total pixels in image. */

    lightdefect = darkdefect = lightseed = 0;
    darkseed = lightfalse = darkfalse = 0;

    for(i=0; i<total_pixels; ++i) {
        if ((*good_offset + i) == WHITE)&&(*out_offset + i) == LIGHT)
            ++lightfalse;
        if ((*good_offset + i) == WHITE)&&(*out_offset + i) == DARK)
            ++darkfalse;
        if (*good_offset + i) == LIGHT
            ++lightdefect;
        if ((*good_offset + i) == LIGHT)&&(*out_offset + i) == LIGHT)
            ++lightseed;
        if ((*good_offset + i) == LIGHT)&&(*out_offset + i) == DARK)
            ++darkfalse;
        if (*good_offset + i) == DARK
            ++darkdefect;
        if ((*good_offset + i) == DARK)&&(*out_offset + i) == LIGHT)
            ++lightfalse;
        if ((*good_offset + i) == DARK)&&(*out_offset + i) == DARK)
            ++darkseed;
    }
    *upperloss = (*weight)*(lightdefect-lightseed)+(1.0-(*weight))*lightfalse;
    *lowerloss = (*weight)*(darkdefect-darkseed)+(1.0-(*weight))*darkfalse;
}

```

```

/*****
*   FUNCTION NAME : calc_skew
*   PURPOSE : calculate skewness of pixels in difference image
*   FUNCTIONS CALLED : none
*   INPUT EXPECTED : pointers to input image structure and mask image
*                   structure
*   OUTPUT EXPECTED : skewness of pixels in image
*   ALGORITHM : one forward scan is used to find the moments and then the
*               skewness
*   REMARKS:
*       no nested loops are used to improve program efficiency
*
*****/
#include <stdio.h>
#include <math.h>
#include "structure.h"
#include "constants.h"

void calc_skew( struct int_image *diff_img, struct int_image *msk_img,
               float *skewness )
{
    int Nx, Ny, i, total_pixels, *diff_offset, *msk_offset, vsum, vsum2, vsum3,
        vttotal, diff;
    float vmean, stddev;

    diff_offset = diff_img->array;
    msk_offset = msk_img->array;

    Nx = diff_img->Nx;
    Ny = diff_img->Ny;
    total_pixels = Nx*Ny;

    vsum = 0;
    vsum2 = 0;
    vsum3 = 0;
    vttotal = 0;

    /* Forward sweep */
    for(i=0; i<total_pixels; ++i) {
        diff = *(diff_offset + i);
        if (*(msk_offset + i) == WHITE) {
            vsum += diff;
            vsum2 += diff * diff;
            vsum3 += diff * diff * diff;
            ++vttotal;
        }
    }
    vmean = (float) vsum / (float) vttotal;
    stddev = ( (float) vsum2 / (float) vttotal ) - (vmean * vmean);
    stddev = sqrt( stddev );

    *skewness = (((float) vsum3 - 3*vmean*(float) vsum2) / ((float) vttotal) +
        2*vmean*vmean*vmean) / (stddev*stddev*stddev);
}

```

```
/******  
This is a symbolic constants header file constants.h.  
*****/  
#define SEEK_SET 0  
#define SEEK_CUR 1  
#define MAXIMUM 255  
#define MINIMUM 0  
#define RANGE 256  
  
#define ONE 1  
#define ZERO 0  
#define BYTE 256  
  
#define LEVELS 256  
#define BLACK 0  
#define WHITE 255  
#define DARK 150  
#define LIGHT 200  
#define YES 1  
#define NO 0
```

```

/*****
*   NAME : Eyler Coates
*   PROGRAM NAME : echo_defects.c
*   PURPOSE : This program generates a defect image from MRI echo-planar
*             images
*   FUNCTIONS USED : viffread.c, find_hist.c, echo_thresh.c, smooth_image.c,
*                   subtract_image.c, region_grow.c, find_pith.c,
*                   viffwrite.c
*   INCLUDE FILES : stdio.h, time.h, constants.h, structure.h
*   INPUT EXPECTED : Name of an input image file in VIFF format, name of a
*                   the respective thresholded image in VIFF format, the 6
*                   parameters of the echo-planar algorithm
*   OUTPUT EXPECTED : A Viff format image containing defect regions
*   REMARKS:
*   The input image file should be read from a integer image data, but if
*   another file is to be read, corresponding corrections are to be made.
*
*****/
#include <stdio.h>
#include "structure.h"
#include <time.h>
#include "constants.h"

/*****
*   PROGRAM NAME : main()
*   PURPOSE : Input file names, call other functions, and report progress
*   FUNCTIONS USED : viffread.c, find_hist.c, echo_thresh.c, smooth_image.c,
*                   subtract_image.c, region_grow.c, find_pith.c,
*                   viffwrite.c
*   INPUT EXPECTED : Name of an input image file in VIFF format, name of a
*                   the respective thresholded image in VIFF format, the 6
*                   parameters of the echo-planar algorithm
*   OUTPUT EXPECTED : A Viff format image containing defect regions
*   REMARKS:
*****/
main( )
{
    char infile[40], smthfile[40], outfile[40], mskfile[40], command[80],
        answer[2];
    float hist[LEVELS], m1, m2, m3, threshold, variance;
    int uclip, lclip, window, connected, ustop, lstop, upclip, lpclip, upstop,
        lpstop;
    double user_time, real_time;
    clock_t begin_clock;
    time_t begin_time;
    struct int_image inp_img, smth_img, diff_img, out_img, msk_img;

    printf("\nEnter the name of the input image file : ");
    scanf("%s", infile );

    printf("\nEnter the name of the thresholded input image file : ");
    scanf("%s", mskfile );

    printf("\nEnter the name of the output image file : ");
    scanf("%s", outfile );

    do {

        printf("\nEnter the upper outlier clipping limit (integer) : ");
        scanf("%d", &uclip );

        printf("\nEnter the lower outlier clipping limit (integer) : ");
        scanf("%d", &lclip );

        printf("\nEnter the averaging window size (odd integer) : ");
        scanf("%d", &window );

        printf("\nEnter the type of connectedness (4/8) : ");
        scanf("%d", &connected );

        printf("\nEnter the light region growing stop limit (integer) : ");
        scanf("%d", &ustop );

        printf("\nEnter the dark region growing stop limit (integer) : ");
        scanf("%d", &lstop );
    }

```

```

printf("\nEnter the upper pith outlier clipping limit (integer) : ");
scanf("%d", &supclip );

printf("\nEnter the lower pith outlier clipping limit (integer) : ");
scanf("%d", &lpclip );

printf("\nEnter the light pith region growing stop limit (integer) : ");
scanf("%d", &upstop );

printf("\nEnter the dark pith region growing stop limit (integer) : ");
scanf("%d", &lpstop );

printf("\nReading in a VIFF byte image file.\n");
viffread( &inp_img, infile );

printf("\nReading in a VIFF byte image file.\n");
viffread( &msk_img, mskfile );

begin_clock = clock();
begin_time = time(NULL);

printf("\nFinding the histogram and first 3 moments of input image.\n");
find_hist( &inp_img, hist, &m1, &m2, &m3 );

printf("\nFinding the moment-preserving threshold of the input image.\n");
echo_thresh( &m1, &m2, &m3, &threshold, hist );

printf("\n The threshold for the image is %f\n", threshold);

printf("\nCreating a smoothed image.\n");
smooth_image( &inp_img, &msk_img, &smth_img, &window );

printf("\nCreating difference image\n");
subtract_image( &inp_img, &smth_img, &diff_img, &msk_img, &variance);

printf("\nThe variance of the foreground from the smoothed surface is %f\n",
variance);

printf("\nGrowing defect regions\n");
region_grow( &diff_img, &out_img, &msk_img, &uclip, &lclip, &connected, &ustop,
&lstop );

printf("\nGrowing defect region in pith area\n");
find_pith( &diff_img, &out_img, &msk_img, &supclip, &lpclip, &connected, &upstop,
&lpstop );

user_time = (double) (clock() - begin_clock)/1000000.0;
real_time = difftime(time(NULL), begin_time);
printf("\n%s%7.3f%s\n%s%7.3f%s\n",
"CPU time: ", user_time, " seconds",
"Real time: ", real_time, " seconds");

printf("\nWriting a defect image in VIFF byte format.\n");
viffwrite( &out_img, outfile );

printf("\nDisplay images? (y/n) : ");
scanf("%s", answer);

if (answer[0] != 'n') {
printf("\nAbout to display input and output images ...\n");
sprintf(command, "editimage -i %s &", infile);
system(command);
sprintf(command, "editimage -i %s &", outfile);
system(command);
}

printf("\nRerun with same images? (y/n) : ");
scanf("%s", answer);

} while (answer[0] != 'n');

printf("\nDone.\n\n");
}

```



```

/*****
*   FUNCTION NAME : echo_thresh
*   PURPOSE : calculate moment-preserving threshold
*   FUNCTIONS CALLED : sqrt
*   INPUT EXPECTED : pointer to the histogram array, the first 3 moments of
*                   an image
*   OUTPUT EXPECTED : moment-preserving threshold value
*   ALGORITHM : The parameters are calculated according to the paper by
*               Tsai. The recommended pixel percentile, p0, is adjusted.
*               Then, p0 is used along with the histogram to determine
*               the threshold that separates the image into two parts
*   REMARKS:
*
*****/

#include <stdio.h>
#include <math.h>

#define L 256 /* L is the possible number of gray levels. */

void echo_thresh( float *m1, float *m2, float *m3, float *threshold,
                  float *hist )
{
    int i;
    float c0, c1, square_root, p0, sum;

    /* Compute parameters for threshold. */
    c0 = ((*m1)*(*m3)-(*m2)*(*m2))/((*m2)-(*m1)*(*m1));
    c1 = ((*m1)*(*m2)-(*m3))/((*m2)-(*m1)*(*m1));
    square_root = (float) sqrt (c1*c1-4*c0);
    p0 = (0.5 * (square_root - c1) - (*m1))/square_root;
    p0 = p0 - 0.03; /* Adjustment made by me to lower
                    /* threshold a bit

    sum = 0; /* Initialize accumulator.
    for(i=0; i<L; ++i){ /* Accumulate histogram probability
        sum += *(hist + i); /* to find threshold that matches
        if (sum > p0) break; /* the calculated percentile, p0.
    }
    *threshold = (float) i - 0.5;
}

```

```

/*****
*   NAME : Eyler Coates
*   PROGRAM NAME : equalize.c
*   PURPOSE : This program performs a histogram equalization on the fore-
*             ground of an image.
*   FUNCTIONS USED : viffread.c, fore_hist_eq.c, viffwrite.c
*   INCLUDE FILES : stdio.h, constants.h, Structure.h
*   INPUT EXPECTED : Name of an input image file in VIFF matrix format, name
*                   of an thresholded image in VIFF format
*   OUTPUT EXPECTED : A Viff format image whose foreground has been
*                     equalized.
*   REMARKS:
*   The input image file should be read from a integer image data, but if
*   another file is to be read, corresponding corrections are to be made.
*****/
#include <stdio.h>
#include "structure.h"
#include "constants.h"

/*****
*   PROGRAM NAME : main()
*   PURPOSE : Input file names, call other functions, and report progress
*   FUNCTIONS USED : viffread.c, fore_hist_eq.c, viffwrite.c
*   INPUT EXPECTED : Name of an input image file in VIFF matrix format, name
*                   of an thresholded image in VIFF format
*   OUTPUT EXPECTED : A Viff format image whose foreground has been
*                     equalized.
*   REMARKS:
*****/
main( )
{
    char infile[40], mskfile[40], outfile[40], command[80], answer[2];
    struct int_image inp_img, out_img, msk_img;

    printf("\nEnter the name of the input image file : ");
    scanf("%s", infile );

    printf("\nEnter the name of the input mask image file : ");
    scanf("%s", mskfile );

    printf("\nEnter the name of the output equalized image file : ");
    scanf("%s", outfile );

    printf("\nReading in a VIFF byte image file.\n");
    viffread( &inp_img, infile );

    printf("\nReading in a VIFF byte image file.\n");
    viffread( &msk_img, mskfile );

    printf("\nEqualizing the foreground of an image.\n");
    fore_hist_eq( &inp_img, &out_img, &msk_img );

    printf("\nWriting an equalized image in VIFF byte format.\n");
    viffwrite( &out_img, outfile );

    printf("\nDisplay images? (y/n) : ");
    scanf("%s", answer);

    if (answer[0] != 'n') {
        printf("\nAbout to display input and equalized images ...\n");
        sprintf(command, "editimage -i %s &", infile);
        system(command);
        sprintf(command, "editimage -i %s &", outfile);
        system(command);
    }

    printf("\nDone.\n\n");
}

```

```

/*****
*   FUNCTION NAME : find_hist
*   PURPOSE : find histogram of gray levels in an image
*   FUNCTIONS CALLED : none
*   INPUT EXPECTED : pointer to input image structure and pointer to
*                   histogram array,
*   OUTPUT EXPECTED : histogram of gray levels stored in an array, also
*                   values for the first three moments of the image
*   ALGORITHM : using one for loop, the pixels are scanned and the count
*               each gray level is incremented if found in a pixel. After
*               the scan, the relative frequency is determined by dividing
*               the counts by the total number of pixels in the image
*   REMARKS : no nested loops are used to improve program efficiency
*
*****/
#include <stdio.h>
#include "structure.h"

#define L 256 /* L is the possible number of gray levels */

void find_hist( struct int_image *inp_img, float *hist, float *m1, float *m2,
               float *m3 )
{
    int Nx, Ny, i, pixels, *offset, gray_level;

    Nx = inp_img->Nx; /* Find dimension of input image */
    Ny = inp_img->Ny;

    pixels = Nx*Ny; /* Compute range of image array */
    offset = inp_img->array;

    *m1 = *m2 = *m3 = 0;
    for(i=0; i<L; ++i) /* Initialize all counts to zero. */
        hist[i] = 0;

    for(i=0; i<pixels; ++i){ /* Scan each pixel and increment */
        gray_level = inp_img->array[i]; /* histogram count. Also, update */
        if (gray_level > 256) /* moments. */
            gray_level = 0;
        ++(hist + gray_level);
        *m1 += gray_level;
        *m2 += gray_level * gray_level;
        *m3 += gray_level * gray_level * gray_level;
    }

    for(i=0; i<L; ++i) /* Divide totals to get relative */
        *(hist + i) /= pixels; /* frequency */
    *m1 /= pixels;
    *m2 /= pixels;
    *m3 /= pixels;
}

```

```

/*****
*   FUNCTION NAME : find_pith
*   PURPOSE : locate center of foreground area and search for defects in
*             a window around the center
*   FUNCTIONS CALLED : none
*   INPUT EXPECTED : pointers to the difference, mask and output images
*                   upper threshold, lower threshold, upper region stop and
*                   lower region stop limits
*   OUTPUT EXPECTED : output image that has the center region checked for
*                   defects
*   ALGORITHM : Using a for loop, the centroid of the foreground is cal-
*               lated. The output image is checked to determine whether any
*               defect labelled pixels are in a window that is centered on
*               the centroid. If there are already defect pixels near the
*               foreground center, then the program assumes that the pith
*               defect has already been found and program ends. If no
*               defects are found initially, the program looks for pixel
*               gray levels in the center window that are greater than the
*               upper threshold or lower than the lower threshold. Then the
*               same regions growing routine employed earlier is used to
*               grow the newly discovered defect regions.
*   REMARKS :
*****/
#include <stdio.h>
#include "structure.h"
#include "constants.h"

#define NULL 0
#define PAREA 10

struct linked_list {
    int          d1;
    int          d2;
    struct linked_list *next;
};

typedef struct linked_list ELEMENT;
typedef ELEMENT *LINK;

struct queue {
    LINK front, rear;
};

typedef struct queue QUEUE;

void find_pith ( struct int_image *diff_img, struct int_image *out_img,
                 struct int_image *msk_img, int *uclip, int *lclip,
                 int *connected, int *ustop, int *lstop)
{
    int i, Nx, Ny, total_pixels, *diff_offset, *out_offset, *msk_offset,
        approach, x, y, foreground_pixels, Previous, Found;
    float xCoord, yCoord;

    QUEUE edgies = {NULL, NULL};

    diff_offset = diff_img->array; /* Find pointers to images */
    out_offset = out_img->array;
    msk_offset = msk_img->array;

    Nx = out_img->Nx = diff_img->Nx; /* Set output image dimensions to */
    Ny = out_img->Ny = diff_img->Ny; /* match the input image. */

    total_pixels = Nx*Ny; /* Compute total pixels in image. */

    printf ("\n Locating center of foreground\n");
    i=0;
    xCoord = 0;
    yCoord = 0;
    foreground_pixels = 0;
    for (y=0; y<Ny; ++y)
        for (x=0; x<Nx; ++x) {
            if (*(msk_offset + i)) {
                xCoord += x;
                yCoord += y;
                ++foreground_pixels;
            }
        }
}

```

```

        ++i;
    }
    xCoord /= foreground_pixels;
    yCoord /= foreground_pixels;
    printf ("\n The centroid coordinates are %f, %f \n", xCoord, yCoord);

    printf ("\n Looking for previous defects in centroid area ...\n");

    Previous = 0;
    for (x=(int)xCoord-PAREA; x<(int)xCoord+PAREA; ++x) {
        for(y=(int)yCoord-PAREA; y<(int)yCoord+PAREA; ++y) {
            i = Nx*y + x;
            if ((*out_offset + i) == LIGHT || (*out_offset + i) == DARK) {
                Previous = 1;
                break;
            }
        }
        if (Previous) break;
    }

    /* Return if defects are already present */
    /* in the center window */
    if (Previous) return;

    printf ("\n No previous defects were found near the centroid.");
    printf ("\n Looking for seeds with less severe restrictions\n");

    Found = 0;
    for (x=(int)xCoord-PAREA; x<(int)xCoord+PAREA; ++x)
        for(y=(int)yCoord-PAREA; y<(int)yCoord+PAREA; ++y) {
            i = Nx*y + x;
            if ((*diff_offset + i) > *uclip) {
                *out_offset + i = LIGHT;
                Found = 1;
            }
            if ((*diff_offset + i) < -(*lclip)) {
                *out_offset + i = DARK;
                Found = 1;
            }
        }

    if (Found) {
        printf("\n Defect region(s) found near centroid.\n");
    }

    printf("\n Initializing queue of pith defect pixels on edge of seeds.\n");

    for (x=(int)xCoord-PAREA; x<(int)xCoord+PAREA; ++x)
        for(y=(int)yCoord-PAREA; y<(int)yCoord+PAREA; ++y) {
            i = Nx*y + x;
            if ((*out_offset + i) == LIGHT) {

                if ((*out_offset + i-1) == WHITE && *diff_offset + i-1 > *ustop) {
                    enqueue (&edgies, i, LIGHT);
                    continue;
                }
                if ((*out_offset + i+1) == WHITE && *diff_offset + i+1 > *ustop) {
                    enqueue (&edgies, i, LIGHT);
                    continue;
                }
                if ((*out_offset + i-Nx) == WHITE && *diff_offset + i-Nx > *ustop) {
                    enqueue (&edgies, i, LIGHT);
                    continue;
                }
                if ((*out_offset + i+Nx) == WHITE && *diff_offset + i+Nx > *ustop) {
                    enqueue (&edgies, i, LIGHT);
                    continue;
                }
                if (*connected != 8)
                    continue;
                if ((*out_offset + i-Nx-1) == WHITE && *diff_offset + i-Nx-1 > *ustop) {
                    enqueue (&edgies, i, LIGHT);
                    continue;
                }
                if ((*out_offset + i-Nx+1) == WHITE && *diff_offset + i-Nx+1 > *ustop) {
                    enqueue (&edgies, i, LIGHT);
                    continue;
                }
            }
        }

```

```

    }
    if (*(out_offset + i+Nx-1) == WHITE && *(diff_offset + i+Nx-1) > *ustop) {
        enqueue (&edgies, i, LIGHT);
        continue;
    }
    if (*(out_offset + i+Nx+1) == WHITE && *(diff_offset + i+Nx+1) > *ustop) {
        enqueue (&edgies, i, LIGHT);
        continue;
    }
}
if (*(out_offset + i) == DARK) {
    if (*(out_offset + i-1) == WHITE && *(diff_offset + i-1) < -(*lstop)) {
        enqueue (&edgies, i, DARK);
        continue;
    }
    if (*(out_offset + i+1) == WHITE && *(diff_offset + i+1) < -(*lstop)) {
        enqueue (&edgies, i, DARK);
        continue;
    }
    if (*(out_offset + i-Nx) == WHITE && *(diff_offset + i-Nx) < -(*lstop)) {
        enqueue (&edgies, i, DARK);
        continue;
    }
    if (*(out_offset + i+Nx) == WHITE && *(diff_offset + i+Nx) < -(*lstop)) {
        enqueue (&edgies, i, DARK);
        continue;
    }
    if (*connected != 8)
        continue;
    if (*(out_offset + i-Nx-1) == WHITE && *(diff_offset + i-Nx-1) < -(*lstop)) {
        enqueue (&edgies, i, DARK);
        continue;
    }
    if (*(out_offset + i-Nx+1) == WHITE && *(diff_offset + i-Nx+1) < -(*lstop)) {
        enqueue (&edgies, i, DARK);
        continue;
    }
    if (*(out_offset + i+Nx-1) == WHITE && *(diff_offset + i+Nx-1) < -(*lstop)) {
        enqueue (&edgies, i, DARK);
        continue;
    }
    if (*(out_offset + i+Nx+1) == WHITE && *(diff_offset + i+Nx+1) < -(*lstop)) {
        enqueue (&edgies, i, DARK);
        continue;
    }
}
}

printf ("\n Growing defect regions from pith defect seeds.\n");

while(!isempty(edgies)) {
    printf(".");
    fflush(stdout);
    dequeue (&edgies, &i, &approach);

    if (approach == LIGHT) {
        if (*(out_offset + i-1) == WHITE && *(diff_offset + i-1) > *ustop) {
            *(out_offset + i-1) = LIGHT;
            enqueue (&edgies, i-1, LIGHT);
        }
        if (*(out_offset + i+1) == WHITE && *(diff_offset + i+1) > *ustop) {
            *(out_offset + i+1) = LIGHT;
            enqueue (&edgies, i+1, LIGHT);
        }
        if (*(out_offset + i-Nx) == WHITE && *(diff_offset + i-Nx) > *ustop) {
            *(out_offset + i-Nx) = LIGHT;
            enqueue (&edgies, i-Nx, LIGHT);
        }
        if (*(out_offset + i+Nx) == WHITE && *(diff_offset + i+Nx) > *ustop) {
            *(out_offset + i+Nx) = LIGHT;
            enqueue (&edgies, i+Nx, LIGHT);
        }
        if (*connected != 8)
            continue;
    }
}

```

```

    if (*(out_offset + i-Nx-1) == WHITE && *(diff_offset + i-Nx-1) > *ustop) {
        *(out_offset + i-Nx-1) = LIGHT;
        enqueue (&edgies, i-Nx-1, LIGHT);
    }
    if (*(out_offset + i-Nx+1) == WHITE && *(diff_offset + i-Nx+1) > *ustop) {
        *(out_offset + i-Nx+1) = LIGHT;
        enqueue (&edgies, i-Nx+1, LIGHT);
    }
    if (*(out_offset + i+Nx-1) == WHITE && *(diff_offset + i+Nx-1) > *ustop) {
        *(out_offset + i+Nx-1) = LIGHT;
        enqueue (&edgies, i+Nx-1, LIGHT);
    }
    if (*(out_offset + i+Nx+1) == WHITE && *(diff_offset + i+Nx+1) > *ustop) {
        *(out_offset + i+Nx+1) = LIGHT;
        enqueue (&edgies, i+Nx+1, LIGHT);
    }
}

if (approach == DARK) {
    if (*(out_offset + i-1) == WHITE && *(diff_offset + i-1) < -(*lstop)) {
        *(out_offset + i-1) = DARK;
        enqueue (&edgies, i-1, DARK);
    }
    if (*(out_offset + i+1) == WHITE && *(diff_offset + i+1) < -(*lstop)) {
        *(out_offset + i+1) = DARK;
        enqueue (&edgies, i+1, DARK);
    }
    if (*(out_offset + i-Nx) == WHITE && *(diff_offset + i-Nx) < -(*lstop)) {
        *(out_offset + i-Nx) = DARK;
        enqueue (&edgies, i-Nx, DARK);
    }
    if (*(out_offset + i+Nx) == WHITE && *(diff_offset + i+Nx) < -(*lstop)) {
        *(out_offset + i+Nx) = DARK;
        enqueue (&edgies, i+Nx, DARK);
    }
    if (*connected != 8)
        continue;
    if (*(out_offset + i-Nx-1) == WHITE && *(diff_offset + i-Nx-1) < -(*lstop)) {
        *(out_offset + i-Nx-1) = DARK;
        enqueue (&edgies, i-Nx-1, DARK);
    }
    if (*(out_offset + i-Nx+1) == WHITE && *(diff_offset + i-Nx+1) < -(*lstop)) {
        *(out_offset + i-Nx+1) = DARK;
        enqueue (&edgies, i-Nx+1, DARK);
    }
    if (*(out_offset + i+Nx-1) == WHITE && *(diff_offset + i+Nx-1) < -(*lstop)) {
        *(out_offset + i+Nx-1) = DARK;
        enqueue (&edgies, i+Nx-1, DARK);
    }
    if (*(out_offset + i+Nx+1) == WHITE && *(diff_offset + i+Nx+1) < -(*lstop)) {
        *(out_offset + i+Nx+1) = DARK;
        enqueue (&edgies, i+Nx+1, DARK);
    }
}

}

printf("\n");
}

```

```

/*****
*   FUNCTION NAME : find_thresh
*   PURPOSE : calculate moment-preserving threshold
*   FUNCTIONS CALLED : sqrt
*   INPUT EXPECTED : pointer to the histogram array, the first 3 moments of
*                   an image
*   OUTPUT EXPECTED : moment-preserving threshold value
*   ALGORITHM : The parameters are calculated according to the paper by
*               Tsai. The recommended pixel percentile, p0, from the
*               calculations is used along with the histogram to determine
*               the threshold that separates the image into two parts
*   REMARKS:
*
*****/

#include <stdio.h>
#include <math.h>

#define      L      256          /* L is the possible number of gray levels. */

void find_thresh( float *m1, float *m2, float *m3, float *threshold,
                  float *hist )
{
    int i;
    float c0, c1, square_root, p0, sum;

    /* Compute parameters for threshold.*/
    c0 = ((*m1)*(*m3)-(*m2)*(*m2))/((*m2)-(*m1)*(*m1));
    c1 = ((*m1)*(*m2)-(*m3))/((*m2)-(*m1)*(*m1));
    square_root = (float) sqrt (c1*c1-4*c0);
    p0 = (0.5 * (square_root - c1) - (*m1))/square_root;

    sum = 0;
    for(i=0; i<L; ++i){
        sum += *(hist + i);
        if (sum > p0) break;
    }
    *threshold = (float) i - 0.5;
}

```



```

/*****
*   FUNCTION NAME : fore_hist_eq
*   PURPOSE : equalize foreground of an input image & put the resultant gray*
*             level values in an output image
*   FUNCTIONS CALLED : none
*   INPUT EXPECTED : pointers to the input, mask and output images
*   OUTPUT EXPECTED : output image that has an equalized histogram in
*                     foreground
*   ALGORITHM : Using a for loop, for each possible gray level, the entire
*                input image is scanned. During each of these scans, the
*                input pixels in the foreground that match the gray level at
*                hand in the loop are sent to the output image. Each output
*                gray level is used as the level for the output image until
*                the "quota" for that gray level is filled, then the next
*                output gray level is used for output. The change in output
*                gray levels occurs independently of the input gray level
*                status. Since the process is continued for each input image
*                gray level, all input pixels are assigned to the output.
*   REMARKS :
*****/
#include <stdio.h>
#include "structure.h"
#include "constants.h"

void fore_hist_eq ( struct int_image *inp_img, struct int_image *out_img,
                    struct int_image *msk_img )
{
    int i, out_gray, in_gray, bin_count, Nx, Ny, total_pixels, foregrd_pixels,
        *inp_offset, *out_offset, *msk_offset, pixels_per_gray_level,
        meter_count, divisor;

    inp_offset = inp_img->array;          /* Find pointers to images */
    out_offset = out_img->array;
    msk_offset = msk_img->array;

    Nx = out_img->Nx = inp_img->Nx;        /* Set output image dimensions to */
    Ny = out_img->Ny = inp_img->Ny;        /* match the input image. */

    total_pixels = Nx*Ny;                 /* Compute total pixels in image. */

    printf("\n Total pixels in image = %d\n", total_pixels);

    foregrd_pixels = 0;                   /* Compute total foreground pixels. */
    for(i=0; i<total_pixels; ++i)
        if(*msk_offset + i)
            ++foregrd_pixels;

    printf("\n Total pixels in foreground = %d\n", foregrd_pixels);
    fflush(stdout);

    /* Compute pixels for each gray level */
    /* on the output image. */
    pixels_per_gray_level = (foregrd_pixels/LEVELS) + 1;

    /* Set up a completion meter. */

    printf("\n          Percent Completion          1");
    printf("\n          1    2    3    4    5    6    7    8    9    0");
    printf("\n0....0....0....0....0....0....0....0....0....0");
    printf("\n");

    meter_count = 0;
    divisor = LEVELS/50;

    out_gray = 0;                         /* Initialize output gray level to 0 */
    bin_count = 0;                         /* Initialize the count of the pixels */
    /* at the output gray level to 0. */
    for (in_gray = 0; in_gray < LEVELS; ++in_gray) { /* For each gray level */
        /* of the input image, ... */
        ++meter_count;
        if(meter_count%divisor == 0) {
            printf("*");
            fflush(stdout);
        }
        i = 0;
        /* Start scan at beginning of the */
        /* input image. */
        while (i < total_pixels) {
            /* Until all pixels in input are */

```

```

/* scanned, ... */
/* If the input pixel is in the */
/* foreground and matches the current*/
/* gray level of concern, ... */
if(*(msk_offset + i) && (*(inp_offset + i) == in_gray)) {
/* If the output gray level quota has*/
/* been filled, ... */
if(bin_count == pixels_per_gray_level) {
/* Reset the count of the pixels in*/
bin_count = 0; /* the output gray level to 0. */
/* Go to a new output gray level. */
++out_gray;
}
++bin_count; /* Increment count of the pixels in */
/* the output gray level. */
*(out_offset + i) = out_gray; /* Set the output pixel to */
/* current unfilled gray level. */
}
++i; /* Look at the next input pixel. */
}
}
printf("\n");
}

```

```

[]
/*****
[]
*      NAME : Eyler Coates
*
*      PROGRAM NAME : loss_seeds.c
*      PURPOSE : This program generates a loss function profile for the seed
*                generation stage
*      FUNCTIONS USED : viffwrite.c, find_hist.c, find_thresh.c,
*                      apply_thresh.c, smooth_image.c, subtract_image.c, seeds.c
*                      calc_loss.c
*      INCLUDE FILES : stdio.h, time.h, constants.h, structure.h
*      INPUT EXPECTED : Name of an input image file in VIFF format, name of an
*                      output text file, weight factor, avg filter window size
*      OUTPUT EXPECTED : An array of upper limits versus loss function and an
*                      array of lower limits versus a loss function
*      REMARKS:
*****/
#include <stdio.h>
#include "structure.h"
#include <time.h>
#include "constants.h"

/*****
*      PROGRAM NAME : main()
*      PURPOSE : This program generates a loss function profile for the seed
*                generation stage
*      FUNCTIONS USED : viffwrite.c, find_hist.c, find_thresh.c,
*                      apply_thresh.c, smooth_image.c, subtract_image.c, seeds.c
*                      calc_loss.c
*      INCLUDE FILES : stdio.h, time.h, constants.h, structure.h
*      INPUT EXPECTED : Name of an input image file in VIFF format, name of an
*                      output text file, weight factor, avg filter window size
*      OUTPUT EXPECTED : An array of upper limits versus loss function and an
*                      array of lower limits versus a loss function
*      REMARKS:
*****/
main( )
{
    char infile[40], smthfile[40], outfile[40], mskfile[40];
    float hist[LEVELS], m1, m2, m3, threshold, variance;
    int uclip, lclip, window, uprange, lowrange;
    float weight, upperloss, lowerloss;
    double user_time, real_time;
    clock_t begin_clock;
    time_t begin_time;
    struct int_image inp_img, smth_img, diff_img, out_img, msk_img, good_img;
    FILE *opt;

    printf("\nEnter the name of the input image file : ");
    scanf("%s", infile );

    printf("\nEnter the name of the good defect image file : ");
    scanf("%s", goodfile );

    printf("\nEnter the name of the output text file : ");
    scanf("%s", outfile );

    printf("\nEnter the averaging window size (odd integer) : ");
    scanf("%d", &window );

    printf("\nEnter the upper range of the seed limits : ");
    scanf("%d", &uprange );

    printf("\nEnter the lower range of the seed limits : ");
    scanf("%d", &lowrange );

    printf("\nEnter the loss function weight factor (0 < w < 1) : ");
    scanf("%f", &weight );

    printf("\nReading in the input image file.\n");
    viffread( &inp_img, infile );

    printf("\nReading in the good defect image file.\n");

```

```

viffread( &good_img, goodfile );

/* Open image file for writing */
opt = fopen( outfile,"w");
if( opt == NULL )
{
    printf("\nUnable to open the output file for writing.\n");
    printf("\nBad Path.\n");
    exit(1);
}

begin_clock = clock();
begin_time = time(NULL);

printf("\nFinding the histogram and first 3 moments of input image.\n");
find_hist( &inp_img, hist, &m1, &m2, &m3 );

printf("\nFinding the moment-preserving threshold of the input image.\n");
find_thresh( &m1, &m2, &m3, &threshold, hist );

printf("\n The threshold for the image is %f\n", threshold);

printf("\nApplying threshold to the input image to produce output.\n");
apply_thresh( &threshold, &inp_img, &msh_img );

printf("\nCreating a smoothed image.\n");
smooth_image( &inp_img, &msh_img, &smth_img, &window );

printf("\nCreating difference image\n");
subtract_image( &inp_img, &smth_img, &diff_img, &msh_img, &variance);

printf("\nThe variance of the foreground from the smoothed surface is %f\n",
variance);

printf("\nUpper/Lower Clip   Upper Loss   Lower Loss\n");

for (uclip=uprange; uclip >= lowrange; --uclip) {
    lclip = uclip;
    seeds( &diff_img, &out_img, &msh_img, &uclip, &lclip);
    calc_loss ( &out_img, &good_img, &upperloss, &lowerloss, &weight);
    printf("%8d      %f      %f\n", uclip, upperloss, lowerloss);
    fprintf("%8d      %f      %f\n", uclip, upperloss, lowerloss);
}

fclose(opt);

user_time = (double) (clock() - begin_clock)/1000000.0;
real_time = difftime(time(NULL), begin_time);
printf("\n%s%7.3f%s\n%s%7.3f%s\n",
"CPU time: ", user_time, " seconds",
"Real time: ", real_time, " seconds");

printf("\nDone.\n\n");
}

```

```

□
/*****
*   NAME : Eyler Coates
*   PROGRAM NAME : loss_stops.c
*   PURPOSE : This program generates a loss function profile for the pixel
*             aggregation stage
*   FUNCTIONS USED : viffwrite.c, find_hist.c, find_thresh.c,
*                   apply_thresh.c, smooth_image.c, subtract_image.c, stops.c
*                   calc_loss.c
*   INCLUDE FILES : Stdio.h, time.h, constants.h, structure.h
*   INPUT EXPECTED : Name of an input image file in VIFF format, name of an
*                   output text file, weight factor, avg filter window size
*   OUTPUT EXPECTED : An array of upper stops versus loss function and an
*                   array of lower stops versus a loss function
*   REMARKS:
*****/
#include <stdio.h>
#include "structure.h"
#include <time.h>
#include "constants.h"

/*****
*   PROGRAM NAME : loss_stops.c
*   PURPOSE : This program generates a loss function profile for the pixel
*             aggregation stage
*   FUNCTIONS USED : viffwrite.c, find_hist.c, find_thresh.c,
*                   apply_thresh.c, smooth_image.c, subtract_image.c, stops.c
*                   calc_loss.c
*   INCLUDE FILES : Stdio.h, time.h, constants.h, structure.h
*   INPUT EXPECTED : Name of an input image file in VIFF format, name of an
*                   output text file, weight factor, avg filter window size
*   OUTPUT EXPECTED : An array of upper stops versus loss function and an
*                   array of lower stops versus a loss function
*   REMARKS:
*****/
main( )
{
    char infile[40], smthfile[40], outfile[40], mskfile[40];
    float hist[LEVELS], m1, m2, m3, threshold, variance;
    int uclip, lclip, window, connectedness, uprange, lowrange;
    float weight, upperloss, lowerloss;
    double user_time, real_time;
    clock_t begin_clock;
    time_t begin_time;
    struct int_image inp_img, smth_img, diff_img, out_img, msk_img, good_img;
    FILE *opt;

    printf("\nEnter the name of the input image file : ");
    scanf("%s", infile );

    printf("\nEnter the name of the good defect image file : ");
    scanf("%s", goodfile );

    printf("\nEnter the name of the output text file : ");
    scanf("%s", outfile );

    printf("\nEnter the averaging window size (odd integer) : ");
    scanf("%d", &window );

    printf("\nEnter the connectivity (4 or 8) : ");
    scanf("%d", &connectedness );

    printf("\nEnter the upper seed limit : ");
    scanf("%d", &uclip );

    printf("\nEnter the lower seed limit : ");
    scanf("%d", &lclip );

    printf("\nEnter the upper range of the stop limits : ");
    scanf("%d", &uprange );

    printf("\nEnter the lower range of the stop limits : ");
    scanf("%d", &lowrange );

```

```

printf("\nEnter the loss function weight factor (0 < w < 1) : ");
scanf("%f", &weight );

printf("\nReading in the input image file.\n");
viffread( &inp_img, infile );

printf("\nReading in the good defect image file.\n");
viffread( &good_img, goodfile );

/* Open image file for writing */
opt = fopen( outfile, "w");
if( opt == NULL )
{
    printf("\nUnable to open the output file for writing.\n");
    printf("\nBad Path.\n");
    exit(1);
}

begin_clock = clock();
begin_time = time(NULL);

printf("\nFinding the histogram and first 3 moments of input image.\n");
find_hist( &inp_img, hist, &m1, &m2, &m3 );

printf("\nFinding the moment-preserving threshold of the input image.\n");
find_thresh( &m1, &m2, &m3, &threshold, hist );

printf("\n The threshold for the image is %f\n", threshold);

printf("\nApplying threshold to the input image to produce output.\n");
apply_thresh( &threshold, &inp_img, &msk_img );

printf("\nCreating a smoothed image.\n");
smooth_image( &inp_img, &msk_img, &smth_img, &window );

printf("\nCreating difference image\n");
subtract_image( &inp_img, &smth_img, &diff_img, &msk_img, &variance);

printf("\nThe variance of the foreground from the smoothed surface is %f\n",
variance);

printf("\nUpper/Lower Stop   Upper Loss   Lower Loss\n");

for (ustop=uprange; ustop >= lowrange; --ustop) {
    lstop = ustop;
    stops( &diff_img, &out_img, &msk_img, &uclip, &lclip, &connectedness, &ustop,
&lstop);
    calc_loss ( &out_img, &good_img, &upperloss, &lowerloss, &weight);
    printf("%8d      %f      %f\n", ustop, upperloss, lowerloss);
    fprintf("%8d      %f      %f\n", ustop, upperloss, lowerloss);
}

fclose(opt);

user_time = (double) (clock() - begin_clock)/1000000.0;
real_time = difftime(time(NULL), begin_time);
printf("\n%s%7.3f%s\n%s%7.3f%s\n",
"CPU time: ", user_time, " seconds",
"Real time: ", real_time, " seconds");

printf("\nDone.\n\n");
}

```

```

# This is the makefile for all the programs used in the dissertation
#
MODULES = threshold defects spin_defects equalize echo_defects zmed3

all: ${MODULES}

clean:
    rm -f *.o ${MODULES}

co:
    rm -f *.o

viffread.o: viffread.c
    acc -c viffread.o viffread.c

viffwrite.o:    viffwrite.c
    acc -c viffwrite.o viffwrite.c

threshold: threshold.o viffread.o find_hist.o find_thresh.o apply_thresh.o viffwrite.o
    acc -o threshold threshold.o viffread.o find_hist.o find_thresh.o apply_thresh.o
    viffwrite.o

threshold.o:    threshold.c
    acc -c threshold.o threshold.c

find_hist.o:    find_hist.c
    acc -c find_hist.o find_hist.c

find_thresh.o:  find_thresh.c
    acc -c find_thresh.o find_thresh.c

apply_thresh.o: apply_thresh.c
    acc -c apply_thresh.o apply_thresh.c

fore_hist_eq.o: fore_hist_eq.c
    acc -c fore_hist_eq.o fore_hist_eq.c

region_grow.o:  region_grow.c
    acc -c region_grow.o region_grow.c

subtract_image.o: subtract_image.c
    acc -c subtract_image.o subtract_image.c

spin_defects:  spin_defects.o viffread.o find_hist.o find_thresh.o apply_thresh.o
smooth_image.o subtract_image.o region_grow.o viffwrite.o
    acc -o spin_defects spin_defects.o viffread.o find_hist.o find_thresh.o
    apply_thresh.o smooth_image.o subtract_image.o region_grow.o viffwrite.o

smooth_image.o: smooth_image.c
    acc -c smooth_image.o smooth_image.c

equalize:  equalize.o viffread.o fore_hist_eq.o viffwrite.o
    acc -o equalize equalize.o viffread.o fore_hist_eq.o viffwrite.o

echo_thresh.o: echo_thresh.c
    acc -c echo_thresh.o echo_thresh.c

find_pith.o:    find_pith.c
    acc -c find_pith.o find_pith.c

echo_defects:  echo_defects.o viffread.o find_hist.o echo_thresh.o smooth_image.o
subtract_image.o region_grow.o find_pith.o viffwrite.o
    acc -o echo_defects echo_defects.o viffread.o find_hist.o echo_thresh.o
    smooth_image.o subtract_image.o region_grow.o find_pith.o viffwrite.o

zmed3:  zmed3.o viffread.o viffwrite.o
    acc -o zmed3 zmed3.o viffread.o viffwrite.o

```

```

[]
/*****
[]
*      NAME : Eyler Coates
*
*      PROGRAM NAME : peak_skew.c
*      PURPOSE : This program applies an averaging filter over a range of window*
*                sizes and calculates the skewness of the residuals for each.
*      FUNCTIONS USED : viffread.c, skew_smooth_image.c, subtract_image.c,
*                      calc_skew.c
*      INCLUDE FILES : stdio.h, constants.h, structure.h, time.h
*      INPUT EXPECTED : Name of an input image file in VIFF format, name of a
*                      mask image file in VIFF format
*      OUTPUT EXPECTED : An array of numbers with the first column for the
*                      window size and the second column for the skewness
*      REMARKS:
*****/
#include <stdio.h>
#include "structure.h"
#include <time.h>
#include "constants.h"

/*****
*      PROGRAM NAME : main()
*      PURPOSE : This program applies an averaging filter over a range of window*
*                sizes and calculates the skewness of the residuals for each.
*      FUNCTIONS CALLED : viffread.c, skew_smooth_image.c, subtract_image.c,
*                      calc_skew.c
*      INPUT EXPECTED : Name of an input image file in VIFF format, name of a
*                      mask image file in VIFF format
*      OUTPUT EXPECTED : An array of numbers with the first column for the
*                      window size and the second column for the skewness
*      REMARKS:
*****/
main( )
{
    char infile[40], outfile[40], mskfile[40];
    float skewness, variance;
    int window;
    double user_time, real_time;
    clock_t begin_clock;
    time_t begin_time;
    struct int_image inp_img, smth_img, diff_img, out_img, msk_img;
    FILE *opt;

    printf("\nEnter the name of the input image file : ");
    scanf("%s", infile );

    printf("\nEnter the name of the mask image file : ");
    scanf("%s", mskfile );

    printf("\nEnter the name of the output text file : ");
    scanf("%s", outfile );

    printf("\nReading in input image file.\n");
    viffread( &inp_img, infile );

    printf("\nReading in msk image file.\n");
    viffread( &msk_img, mskfile );

    /* Open image file for writing */
    opt = fopen( outfile,"w");
    if( opt == NULL )
    {
        printf("\nUnable to open the output file for writing.\n");
        printf("\nBad Path.\n");
        exit(1);
    }

    begin_clock = clock();
    begin_time = time(NULL);

    printf("\nWindow Size    Skewness \n");

```



```

for(window=5; window<=41; window = window + 2) {
    skew_smooth_image( &inp_img, &msk_img, &smth_img, &window );
    subtract_image( &inp_img, &smth_img, &diff_img, &msk_img, &variance);
    calc_skew( &diff_img, &msk_img, &skewness);
    printf("%6d      %f\n", window, skewness);
    fprintf( opt, "%6d      %f\n", window, skewness);
}

fclose(opt);

user_time = (double) (clock() - begin_clock)/1000000.0;
real_time = difftime(time(NULL), begin_time);
printf("\n%s%7.3f%s\n%s%7.3f%s\n",
    "CPU time: ", user_time, " seconds",
    "Real time: ", real_time, " seconds");

printf("\nDone.\n\n");
}

```

```

/*****

This is a header file queue.h containing declarations and
code for queue functions

*****/

#define      NULL 0

struct linked_list {
    int      d1;
    int      d2;
    struct linked_list *next;
};

typedef      struct linked_list      ELEMENT;
typedef      ELEMENT                *LINK;

struct queue {
    LINK front, rear;
};

typedef      struct queue            QUEUE;

int isempty(QUEUE q)
{
    return (q.front == NULL);
}

int vfront(QUEUE q)
{
    return (q.front -> d1);
}

void dequeue (QUEUE *q, int *x, int *y)
{
    LINK temp = q->front;

    if (!isempty(*q)) {
        *x = temp -> d1;
        *y = temp -> d2;
        q -> front = temp -> next;
        free(temp);
    }
    else
        printf("Empty queue.\n");
}

void enqueue (QUEUE *q, int x, int y)
{
    LINK temp;

    temp = (LINK) malloc(sizeof(ELEMENT));
    temp -> d1 = x;
    temp -> d2 = y;
    temp -> next = NULL;
    if(isempty(*q))
        q -> front = q -> rear = temp;
    else {
        q -> rear -> next = temp;
        q -> rear = temp;
    }
}

```

```

/*****
*   FUNCTION NAME : region_grow
*   PURPOSE : locate defect seeds on difference image foreground and grow
*             defect regions. Label defect pixels in output image
*   FUNCTIONS CALLED : none
*   INPUT EXPECTED : pointers to the difference, mask and output images
*                   upper and lower thresholds, upper and lower region stop
*   OUTPUT EXPECTED : output image with defect region pixels labelled
*   ALGORITHM : Using a for loop, foreground area of the difference image
*               is checked for pixel gray levels that are greater than the
*               upper threshold or lower than the lower threshold. Then
*               using a queue structure, adjacent pixels to the defect seeds
*               are added to the region if the gray level does not violate
*               the region stop limits. When the queue is exhausted, there
*               are no more pixels to be added to the defect regions.
*   REMARKS :
*****/
#include <stdio.h>
#include "structure.h"
#include "queue.h"
#include "constants.h"

void region_grow ( struct int_image *diff_img, struct int_image *out_img,
                  struct int_image *msk_img, int *uclip, int *lclip,
                  int *connected, int *ustop, int *lstop)
{
    int i, Nx, Ny, total_pixels, *diff_offset, *out_offset, *msk_offset,
        approach;

    QUEUE edgies = {NULL, NULL};

    diff_offset = diff_img->array; /* Find pointers to images */
    out_offset = out_img->array;
    msk_offset = msk_img->array;

    Nx = out_img->Nx = diff_img->Nx; /* Set output image dimensions to */
    Ny = out_img->Ny = diff_img->Ny; /* match the input image. */

    total_pixels = Nx*Ny; /* Compute total pixels in image. */

    printf ("\n Marking output image with defect seeds\n");

    for(i=0; i<total_pixels; ++i) {
        *(out_offset + i) = *(msk_offset + i);
        if (*(diff_offset + i) > *uclip)
            *(out_offset + i) = LIGHT;
        if (*(diff_offset + i) < -*lclip)
            *(out_offset + i) = DARK;
    }

    printf("\n Initializing queue of defect pixels on edge of seeds.\n");

    for(i=0; i<total_pixels; ++i) {
        if (*(out_offset + i) == LIGHT) {
            if (*(out_offset + i-1) == WHITE && *(diff_offset + i-1) > *ustop) {
                enqueue (&edgies, i, LIGHT);
                continue;
            }
            if (*(out_offset + i+1) == WHITE && *(diff_offset + i+1) > *ustop) {
                enqueue (&edgies, i, LIGHT);
                continue;
            }
            if (*(out_offset + i-Nx) == WHITE && *(diff_offset + i-Nx) > *ustop) {
                enqueue (&edgies, i, LIGHT);
                continue;
            }
            if (*(out_offset + i+Nx) == WHITE && *(diff_offset + i+Nx) > *ustop) {
                enqueue (&edgies, i, LIGHT);
                continue;
            }
            if (*connected != 8)
                continue;
            if (*(out_offset + i-Nx-1) == WHITE && *(diff_offset + i-Nx-1) > *ustop) {
                enqueue (&edgies, i, LIGHT);
                continue;
            }

```

```

    }
    if (*(out_offset + i-Nx+1) == WHITE && *(diff_offset + i-Nx+1) > *ustop) {
        enqueue (&edgies, i, LIGHT);
        continue;
    }
    if (*(out_offset + i+Nx-1) == WHITE && *(diff_offset + i+Nx-1) > *ustop) {
        enqueue (&edgies, i, LIGHT);
        continue;
    }
    if (*(out_offset + i+Nx+1) == WHITE && *(diff_offset + i+Nx+1) > *ustop) {
        enqueue (&edgies, i, LIGHT);
        continue;
    }
}
if (*(out_offset + i) == DARK) {
    if (*(out_offset + i-1) == WHITE && *(diff_offset + i-1) < -(*lstop)) {
        enqueue (&edgies, i, DARK);
        continue;
    }
    if (*(out_offset + i+1) == WHITE && *(diff_offset + i+1) < -(*lstop)) {
        enqueue (&edgies, i, DARK);
        continue;
    }
    if (*(out_offset + i-Nx) == WHITE && *(diff_offset + i-Nx) < -(*lstop)) {
        enqueue (&edgies, i, DARK);
        continue;
    }
    if (*(out_offset + i+Nx) == WHITE && *(diff_offset + i+Nx) < -(*lstop)) {
        enqueue (&edgies, i, DARK);
        continue;
    }
    if (*connected != 8)
        continue;
    if (*(out_offset + i-Nx-1) == WHITE && *(diff_offset + i-Nx-1) < -(*lstop)) {
        enqueue (&edgies, i, DARK);
        continue;
    }
    if (*(out_offset + i-Nx+1) == WHITE && *(diff_offset + i-Nx+1) < -(*lstop)) {
        enqueue (&edgies, i, DARK);
        continue;
    }
    if (*(out_offset + i+Nx-1) == WHITE && *(diff_offset + i+Nx-1) < -(*lstop)) {
        enqueue (&edgies, i, DARK);
        continue;
    }
    if (*(out_offset + i+Nx+1) == WHITE && *(diff_offset + i+Nx+1) < -(*lstop)) {
        enqueue (&edgies, i, DARK);
        continue;
    }
}
}

printf ("\n Growing defect regions from defect seeds.\n");

while(!isempty(edgies)) {
    printf(".");
    fflush(stdout);
    dequeue (&edgies, &i, &approach);

    if (approach == LIGHT) {
        if (*(out_offset + i-1) == WHITE && *(diff_offset + i-1) > *ustop) {
            *(out_offset + i-1) = LIGHT;
            enqueue (&edgies, i-1, LIGHT);
        }
        if (*(out_offset + i+1) == WHITE && *(diff_offset + i+1) > *ustop) {
            *(out_offset + i+1) = LIGHT;
            enqueue (&edgies, i+1, LIGHT);
        }
        if (*(out_offset + i-Nx) == WHITE && *(diff_offset + i-Nx) > *ustop) {
            *(out_offset + i-Nx) = LIGHT;
            enqueue (&edgies, i-Nx, LIGHT);
        }
        if (*(out_offset + i+Nx) == WHITE && *(diff_offset + i+Nx) > *ustop) {
            *(out_offset + i+Nx) = LIGHT;
        }
    }
}

```

```

        enqueue (&edgies, i+Nx, LIGHT);
    }
    if (*connected != 8)
        continue;
    if (*(out_offset + i-Nx-1) == WHITE && *(diff_offset + i-Nx-1) > *ustop) {
        *(out_offset + i-Nx-1) = LIGHT;
        enqueue (&edgies, i-Nx-1, LIGHT);
    }
    if (*(out_offset + i-Nx+1) == WHITE && *(diff_offset + i-Nx+1) > *ustop) {
        *(out_offset + i-Nx+1) = LIGHT;
        enqueue (&edgies, i-Nx+1, LIGHT);
    }
    if (*(out_offset + i+Nx-1) == WHITE && *(diff_offset + i+Nx-1) > *ustop) {
        *(out_offset + i+Nx-1) = LIGHT;
        enqueue (&edgies, i+Nx-1, LIGHT);
    }
    if (*(out_offset + i+Nx+1) == WHITE && *(diff_offset + i+Nx+1) > *ustop) {
        *(out_offset + i+Nx+1) = LIGHT;
        enqueue (&edgies, i+Nx+1, LIGHT);
    }
}

if (approach == DARK) {
    if (*(out_offset + i-1) == WHITE && *(diff_offset + i-1) < -(*lstop)) {
        *(out_offset + i-1) = DARK;
        enqueue (&edgies, i-1, DARK);
    }
    if (*(out_offset + i+1) == WHITE && *(diff_offset + i+1) < -(*lstop)) {
        *(out_offset + i+1) = DARK;
        enqueue (&edgies, i+1, DARK);
    }
    if (*(out_offset + i-Nx) == WHITE && *(diff_offset + i-Nx) < -(*lstop)) {
        *(out_offset + i-Nx) = DARK;
        enqueue (&edgies, i-Nx, DARK);
    }
    if (*(out_offset + i+Nx) == WHITE && *(diff_offset + i+Nx) < -(*lstop)) {
        *(out_offset + i+Nx) = DARK;
        enqueue (&edgies, i+Nx, DARK);
    }
    if (*connected != 8)
        continue;
    if (*(out_offset + i-Nx-1) == WHITE && *(diff_offset + i-Nx-1) < -(*lstop)) {
        *(out_offset + i-Nx-1) = DARK;
        enqueue (&edgies, i-Nx-1, DARK);
    }
    if (*(out_offset + i-Nx+1) == WHITE && *(diff_offset + i-Nx+1) < -(*lstop)) {
        *(out_offset + i-Nx+1) = DARK;
        enqueue (&edgies, i-Nx+1, DARK);
    }
    if (*(out_offset + i+Nx-1) == WHITE && *(diff_offset + i+Nx-1) < -(*lstop)) {
        *(out_offset + i+Nx-1) = DARK;
        enqueue (&edgies, i+Nx-1, DARK);
    }
    if (*(out_offset + i+Nx+1) == WHITE && *(diff_offset + i+Nx+1) < -(*lstop)) {
        *(out_offset + i+Nx+1) = DARK;
        enqueue (&edgies, i+Nx+1, DARK);
    }
}

}

printf("\n");
}

```

```

/*****
FUNCTION NAME : seeds
[ ]
*   PURPOSE : generate only seeds in output image
*   FUNCTIONS CALLED : none
*   INPUT EXPECTED : pointers to the input, mask and output images, upper
*                   and lower clip limits
*   OUTPUT EXPECTED : output image that has seeds only
*   ALGORITHM : Using a single pass the entire output is made identical to
*               the mask image so that the background area is all set to 0.
*               At each pixel, the difference image is checked at the same
*               location. If the difference exceeds the upper clip or the
*               lower clip, the color of the output image is set according
*               to which limit is exceeded.
*   REMARKS :
*****/
#include <stdio.h>
#include "structure.h"
#include "constants.h"

void seeds ( struct int_image *diff_img, struct int_image *out_img,
             struct int_image *msk_img, int *uclip, int *lclip)
{
    int i, Nx, Ny, total_pixels, *diff_offset, *out_offset, *msk_offset;

    diff_offset = diff_img->array;      /* Find pointers to images */
    out_offset = out_img->array;
    msk_offset = msk_img->array;

    Nx = out_img->Nx = diff_img->Nx;      /* Set output image dimensions to */
    Ny = out_img->Ny = diff_img->Ny;      /* match the input image. */

    total_pixels = Nx*Ny;                /* Compute total pixels in image. */

    for(i=0; i<total_pixels; ++i) {
        *(out_offset + i) = *(msk_offset + i);
        if (*(diff_offset + i) > *uclip)
            *(out_offset + i) = LIGHT;
        if (*(diff_offset + i) < -*lclip)
            *(out_offset + i) = DARK;
    }
}

```

```

/*****
[]
*   FUNCTION NAME : skew_smooth_image.c                               *
[]
*   PURPOSE : apply averaging filter to image to generate a new image *
*   FUNCTIONS CALLED : none                                           *
*   INPUT EXPECTED : pointers to input image structure, mask image structure*
*                   and output image structure                         *
*   OUTPUT EXPECTED : average filtered image over region defined by mask *
*                   image                                              *
*   ALGORITHM : one forward scan is used to average filter the region *
*               defined by the mask image, a square window is used except at*
*               the edges of the region where only pixels that are in both *
*               the window and the mask region are used to generate the avg.*
*   REMARKS:                                                         */
*****/
#include <stdio.h>
#include "structure.h"
#include "constants.h"

void skew_smooth_image( struct int_image *inp_img, struct int_image *msk_img,
                        struct int_image *smth_img, int *window )
{
    int Nx, Ny, i, j, k, total_pixels, *inp_offset, *msk_offset, *smth_offset,
        start_over, pixel_count, sum, j1, j2, divisor;

    inp_offset = inp_img->array;
    smth_offset = smth_img->array;
    msk_offset = msk_img->array;
    Nx = smth_img->Nx = inp_img->Nx;
    Ny = smth_img->Ny = inp_img->Ny;
    total_pixels = Nx*Ny;
                                /* Forward sweep                        */

    start_over = YES;
    pixel_count = 0;
    sum = 0;

    for(i=0; i<(Nx * (*window/2)); ++i) {

        if (*(msk_offset + i) == BLACK) {
            start_over = YES;
            *(smth_offset + i) = BLACK;
            continue;
        }
        if (start_over == NO) {
            j1 = -(*window/2);
            j2 = (*window/2);
            for (k=-(i/Nx); k<=(*window/2); ++k) {
                if (*(msk_offset + i + j1 + k*Nx) == WHITE) {
                    --pixel_count;
                    sum -= *(inp_offset + i + j1 + k*Nx);
                }
                if (*(msk_offset + i + j2 + k*Nx) == WHITE) {
                    ++pixel_count;
                    sum += *(inp_offset + i + j2 + k*Nx);
                }
            }
            *(smth_offset + i) = sum/pixel_count;
            continue;
        }
        else {
            start_over = NO;
            pixel_count = 0;
            sum = 0;
            for (k=-(i/Nx); k<=(*window/2); ++k)
                for (j=-(*window/2); j<=(*window/2); ++j)
                    if (*(msk_offset + i + j + k*Nx) == WHITE) {
                        ++pixel_count;
                        sum += *(inp_offset + i + j + k*Nx);
                    }
            *(smth_offset + i) = sum/pixel_count;
        }
    }
}

```

```

for(i=(Nx * (*window/2)); i<total_pixels - (Nx * (*window/2)); ++i) {

    if (*(msk_offset + i) == BLACK) {
        start_over = YES;
        *(smth_offset + i) = BLACK;
        continue;
    }
    if (start_over == NO) {
        j1 = -(*window/2);
        j2 = (*window/2);
        for (k=-(*window/2); k<=(*window/2); ++k) {
            if (*(msk_offset + i + j1 + k*Nx) == WHITE) {
                --pixel_count;
                sum -= *(inp_offset + i + j1 + k*Nx);
            }
            if (*(msk_offset + i + j2 + k*Nx) == WHITE) {
                ++pixel_count;
                sum += *(inp_offset + i + j2 + k*Nx);
            }
        }
        *(smth_offset + i) = sum/pixel_count;
        continue;
    }
    else {
        start_over = NO;
        pixel_count = 0;
        sum = 0;
        for (k=-(*window/2); k<=(*window/2); ++k)
            for (j=-(*window/2); j<=(*window/2); ++j)
                if (*(msk_offset + i + j + k*Nx) == WHITE) {
                    ++pixel_count;
                    sum += *(inp_offset + i + j + k*Nx);
                }
        *(smth_offset + i) = sum/pixel_count;
    }
}

for(i=total_pixels - (Nx * (*window/2)); i<total_pixels; ++i) {

    if (*(msk_offset + i) == BLACK) {
        start_over = YES;
        *(smth_offset + i) = BLACK;
        continue;
    }
    if (start_over == NO) {
        j1 = -(*window/2);
        j2 = (*window/2);
        for (k=-(*window/2); k<=(total_pixels-i)/Nx; ++k) {
            if (*(msk_offset + i + j1 + k*Nx) == WHITE) {
                --pixel_count;
                sum -= *(inp_offset + i + j1 + k*Nx);
            }
            if (*(msk_offset + i + j2 + k*Nx) == WHITE) {
                ++pixel_count;
                sum += *(inp_offset + i + j2 + k*Nx);
            }
        }
        *(smth_offset + i) = sum/pixel_count;
        continue;
    }
    else {
        start_over = NO;
        pixel_count = 0;
        sum = 0;
        for (k=-(*window/2); k<=(total_pixels-i)/Nx; ++k)
            for (j=-(*window/2); j<=(*window/2); ++j)
                if (*(msk_offset + i + j + k*Nx) == WHITE) {
                    ++pixel_count;
                    sum += *(inp_offset + i + j + k*Nx);
                }
        *(smth_offset + i) = sum/pixel_count;
    }
}
printf("\n");
}

```



```

/*****
*   NAME : Eyler Coates
*   PROGRAM NAME : spin_defects.c
*   PURPOSE : This program generates an image that identifies the defect
*             regions from a spin-echo MRI image of a hardwood log
*   FUNCTIONS USED : viffread.c, viffwrite.c, find_hist.c, find_thresh.c,
*                   apply_thresh.c, smooth_image.c, subtract_image.c,
*                   region_grow.c
*   INCLUDE FILES : stdio.h, time.h, constants.h, structure.h
*   INPUT EXPECTED : Name of an input image file in VIFF format and 4
*                   parameters
*   OUTPUT EXPECTED : A Viff format image
*   REMARKS:
*
*****/
#include <stdio.h>
#include "structure.h"
#include <time.h>
#include "constants.h"

/*****
*   PROGRAM NAME : main()
*   PURPOSE : Input file names, call other functions, and report progress
*   FUNCTIONS CALLED : viffread.c, viffwrite.c, find_hist.c, find_thresh.c,
*                   apply_thresh.c, smooth_image.c, subtract_image.c,
*                   region_grow.c
*   INPUT EXPECTED : Name of an input image file in VIFF format and 4
*                   parameters
*   OUTPUT EXPECTED : A Viff format image
*   REMARKS:
*
*****/
main( )
{
    char infile[40], smthfile[40], outfile[40], mskfile[40], command[80],
        answer[2];
    float hist[LEVELS], m1, m2, m3, threshold;
    int clip, window, connected, stop;
    double user_time, real_time;
    clock_t begin_clock;
    time_t begin_time;
    struct int_image inp_img, smth_img, diff_img, out_img, msk_img;

    printf("\nEnter the name of the input image file : ");
    scanf("%s", infile );

    printf("\nEnter the name of the output image file : ");
    scanf("%s", outfile );

    printf("\nEnter the outlier clipping limit (integer) : ");
    scanf("%d", &clip );

    printf("\nEnter the averaging window size (odd integer) : ");
    scanf("%d", &window );

    printf("\nEnter the type of connectedness (4/8) : ");
    scanf("%d", &connected );

    printf("\nEnter the region growing stop limit (integer) : ");
    scanf("%d", &stop );

    printf("\nReading in a VIFF byte image file.\n");
    viffread( &inp_img, infile );

    begin_clock = clock();
    begin_time = time(NULL);

    printf("\nFinding the histogram and first 3 moments of input image.\n");
    find_hist( &inp_img, hist, &m1, &m2, &m3 );

    printf("\nFinding the moment-preserving threshold of the input image.\n");
    find_thresh( &m1, &m2, &m3, &threshold, hist );

    printf("\n The threshold for the image is %f\n", threshold);

    printf("\nApplying threshold to the input image to produce output.\n");

```

```

    apply_thresh( &threshold, &inp_img, &msk_img );

    printf("\nCreating a smoothed image.\n");
    smooth_image( &inp_img, &msk_img, &smth_img, &window );

    printf("\nCreating difference image\n");
    subtract_image( &inp_img, &smth_img, &diff_img);

    printf("\nGrowing defect regions\n");
    region_grow( &diff_img, &out_img, &msk_img, &clip, &connected, &stop);

    user_time = (double) (clock() - begin_clock)/1000000.0;
    real_time = difftime(time(NULL), begin_time);
    printf("\n%s%.7f%s\n",
           "CPU time: ", user_time, " seconds",
           "Real time: ", real_time, " seconds");

    printf("\nWriting a defect image in VIFF byte format.\n");
    viffwrite( &out_img, outfile );

    printf("\nDisplay images? (y/n) : ");
    scanf("%s", answer);

    if (answer[0] != 'n') {
        printf("\nAbout to display input and output images ...\n");
        sprintf(command, "editimage -i %s &", infile);
        system(command);
        sprintf(command, "editimage -i %s &", outfile);
        system(command);
    }

    printf("\nDone.\n\n");
}

```

```

/*****
*   FUNCTION NAME : smooth_image.c
*   PURPOSE : apply averaging filter to image to generate a new image
*   FUNCTIONS CALLED : none
*   INPUT EXPECTED : pointers to input image structure, mask image structure
*                   and output image structure
*   OUTPUT EXPECTED : average filtered image over region defined by the mask
*                   image
*   ALGORITHM : one forward scan is used to average filter the region
*               defined by the mask image, a square window is used except at
*               the edges of the region where only pixels that are in both
*               the window and the mask region are used to generate the avg.
*   REMARKS:
*
*****/
#include <stdio.h>
#include "structure.h"
#include "constants.h"

void smooth_image( struct int_image *inp_img, struct int_image *msk_img,
                  struct int_image *smth_img, int *window)
{
    int Nx, Ny, i, j, k, total_pixels, *inp_offset, *msk_offset, *smth_offset,
        start_over, pixel_count, sum, j1, j2, divisor;

    inp_offset = inp_img->array;
    smth_offset = smth_img->array;
    msk_offset = msk_img->array;
    Nx = smth_img->Nx = inp_img->Nx;
    Ny = smth_img->Ny = inp_img->Ny;
    total_pixels = Nx*Ny;

    start_over = YES;
    pixel_count = 0;
    sum = 0;

    /* Set up completion meter */
    printf("\n          Percent Completion          1");
    printf("\n      1      2      3      4      5      6      7      8      9      0");
    printf("\n0.....0.....0.....0.....0.....0.....0.....0.....0");
    printf("\n");

    divisor = total_pixels/50;

    /* Forward sweep */
    /* For first few rows, .. */
    for(i=0; i<(Nx * (*window/2)); ++i) {

        if(i%divisor == 0) {
            printf("***");
            fflush(stdout);
        }

        /* If window center pixel is not in */
        /* foreground, set output pixel to */
        /* black. Indicate that the next */
        /* average is to be calculated */
        /* from scratch and move on to the */
        /* next pixel. */
        if (*(msk_offset + i) == BLACK) {
            start_over = YES;
            *(smth_offset + i) = BLACK;
            continue;
        }

        /* Check if window average needs to */
        /* calculated from scratch. */
        if (start_over == YES) {
            start_over = NO;
            pixel_count = 0;
            sum = 0;
            for (k=-(i/Nx); k<=(*window/2); ++k)
                for (j=-(i/Nx); j<=(*window/2); ++j)
                    if (*(msk_offset + i + j + k*Nx) == WHITE) {
                        ++pixel_count;
                        sum += *(inp_offset + i + j + k*Nx);
                    }
            *(smth_offset + i) = sum/pixel_count;
        }
    }
}

```

```

        continue;
    }

    /* If window does not need to be
    /* calculated from scratch. Then
    /* remove the first window row and
    /* add a new row. Recalculate average*/

    if (start_over == NO) {
        j1 = -(*window/2);
        j2 = (*window/2);
        for (k=-(*window/2); k<=(*window/2); ++k) {
            if (*(msk_offset + i + j1 + k*Nx) == WHITE) {
                --pixel_count;
                sum -= *(inp_offset + i + j1 + k*Nx);
            }
            if (*(msk_offset + i + j2 + k*Nx) == WHITE) {
                ++pixel_count;
                sum += *(inp_offset + i + j2 + k*Nx);
            }
        }
        *(smth_offset + i) = sum/pixel_count;
    }

    /* Continue forward sweep.
    /* These pixels are in the center of
    /* of the image.
    for(i=(Nx * (*window/2)); i<total_pixels - (Nx * (*window/2)); ++i) {

        if(i%divisor == 0) {
            printf("\n");
            fflush(stdout);
        }

        /* If window center pixel is not in
        /* foreground, set output pixel to
        /* black. Indicate that the next
        /* average is to be calculated
        /* from scratch and move on to the
        /* next pixel.

        if (*(msk_offset + i) == BLACK) {
            start_over = YES;
            *(smth_offset + i) = BLACK;
            continue;
        }

        /* Check if window average needs to
        /* calculated from scratch.

        if (start_over == YES) {
            start_over = NO;
            pixel_count = 0;
            sum = 0;
            for (k=-(*window/2); k<=(*window/2); ++k)
                for (j=-(*window/2); j<=(*window/2); ++j)
                    if (*(msk_offset + i + j + k*Nx) == WHITE) {
                        ++pixel_count;
                        sum += *(inp_offset + i + j + k*Nx);
                    }
            *(smth_offset + i) = sum/pixel_count;
            continue;
        }

        /* If window does not need to be
        /* calculated from scratch. Then
        /* remove the first window row and
        /* add a new row. Recalculate average*/

        if (start_over == NO) {
            j1 = -(*window/2);
            j2 = (*window/2);
            for (k=-(*window/2); k<=(*window/2); ++k) {
                if (*(msk_offset + i + j1 + k*Nx) == WHITE) {
                    --pixel_count;
                    sum -= *(inp_offset + i + j1 + k*Nx);
                }
                if (*(msk_offset + i + j2 + k*Nx) == WHITE) {
                    ++pixel_count;
                    sum += *(inp_offset + i + j2 + k*Nx);
                }
            }
            *(smth_offset + i) = sum/pixel_count;
        }
    }

```

```

    }
}

/* Continue forward sweep. */
/* These pixels are in the last few */
/* rows of the image. */

for(i=total_pixels - (Nx * (*window/2)); i<total_pixels; ++i) {

    if(i%divisor == 0) {
        printf(" ");
        fflush(stdout);
    }

    /* If window center pixel is not in */
    /* foreground, set output pixel to */
    /* black. Indicate that the next */
    /* average is to be calculated */
    /* from scratch and move on to the */
    /* next pixel. */

    if (*(msk_offset + i) == BLACK) {
        start_over = YES;
        *(smth_offset + i) = BLACK;
        continue;
    }

    /* Check if window average needs to */
    /* be calculated from scratch. */

    if (start_over == YES) {
        start_over = NO;
        pixel_count = 0;
        sum = 0;
        for (k=-(*window/2); k<=(total_pixels-i)/Nx; ++k)
            for (j=-(*window/2); j<=(*window/2); ++j)
                if (*(msk_offset + i + j + k*Nx) == WHITE) {
                    ++pixel_count;
                    sum += *(inp_offset + i + j + k*Nx);
                }
        *(smth_offset + i) = sum/pixel_count;
        continue;
    }

    /* If window does not need to be */
    /* calculated from scratch. Then */
    /* remove the first window row and */
    /* add a new row. Recalculate average*/

    if (start_over == NO) {
        j1 = -(*window/2);
        j2 = (*window/2);
        for (k=-(*window/2); k<=(total_pixels-i)/Nx; ++k) {
            if (*(msk_offset + i + j1 + k*Nx) == WHITE) {
                --pixel_count;
                sum -= *(inp_offset + i + j1 + k*Nx);
            }
            if (*(msk_offset + i + j2 + k*Nx) == WHITE) {
                ++pixel_count;
                sum += *(inp_offset + i + j2 + k*Nx);
            }
        }
        *(smth_offset + i) = sum/pixel_count;
    }
}

printf("\n");
}

```

```

/*****
FUNCTION NAME : stops
[]
*   PURPOSE : generate defect areas in output image
*   FUNCTIONS CALLED : none
*   INPUT EXPECTED : pointers to the input, mask and output images, upper
*                   and lower clip limits, and upper and lower stops
*   OUTPUT EXPECTED : output image that has defect areas
*   ALGORITHM : Using a single pass the entire output is made identical to
*               the mask image so that the background area is all set to 0.
*               At each pixel, the difference image is checked at the same
*               location. If the difference exceeds the upper clip or the
*               lower clip, the color of the output image is set according
*               to which limit is exceeded. After this seed generation stage
*               the neighboring pixels are aggregated via queues until the
*               pixel values fall below the upper and lower stops.
*   REMARKS :
*****/
#include <stdio.h>
#include "structure.h"
#include "queue.h"
#include "constants.h"

void stops ( struct int_image *diff_img, struct int_image *out_img,
             struct int_image *msk_img, int *uclip, int *lclip,
             int *connected, int *ustop, int *lstop)
{
    int i, Nx, Ny, total_pixels, *diff_offset, *out_offset, *msk_offset,
        approach;

    QUEUE edgies = {NULL, NULL};

    diff_offset = diff_img->array; /* Find pointers to images */
    out_offset = out_img->array;
    msk_offset = msk_img->array;

    Nx = out_img->Nx = diff_img->Nx; /* Set output image dimensions to */
    Ny = out_img->Ny = diff_img->Ny; /* match the input image. */

    total_pixels = Nx*Ny; /* Compute total pixels in image. */

    for(i=0; i<total_pixels; ++i) {
        *(out_offset + i) = *(msk_offset + i);
        if (*(diff_offset + i) > *uclip)
            *(out_offset + i) = LIGHT;
        if (*(diff_offset + i) < -(*lclip))
            *(out_offset + i) = DARK;
    }

    for(i=0; i<total_pixels; ++i) {
        if (*(out_offset + i) == LIGHT) {
            if (*(out_offset + i-1) == WHITE && *(diff_offset + i-1) > *ustop) {
                enqueue (&edgies, i, LIGHT);
                continue;
            }
            if (*(out_offset + i+1) == WHITE && *(diff_offset + i+1) > *ustop) {
                enqueue (&edgies, i, LIGHT);
                continue;
            }
            if (*(out_offset + i-Nx) == WHITE && *(diff_offset + i-Nx) > *ustop) {
                enqueue (&edgies, i, LIGHT);
                continue;
            }
            if (*(out_offset + i+Nx) == WHITE && *(diff_offset + i+Nx) > *ustop) {
                enqueue (&edgies, i, LIGHT);
                continue;
            }
            if (*connected != 8)
                continue;
            if (*(out_offset + i-Nx-1) == WHITE && *(diff_offset + i-Nx-1) > *ustop) {
                enqueue (&edgies, i, LIGHT);
                continue;
            }
            if (*(out_offset + i-Nx+1) == WHITE && *(diff_offset + i-Nx+1) > *ustop) {
                enqueue (&edgies, i, LIGHT);
            }
        }
    }
}

```

```

        continue;
    }
    if (*(out_offset + i+Nx-1) == WHITE && *(diff_offset + i+Nx-1) > *ustop) {
        enqueue (&edgies, i, LIGHT);
        continue;
    }
    if (*(out_offset + i+Nx+1) == WHITE && *(diff_offset + i+Nx+1) > *ustop) {
        enqueue (&edgies, i, LIGHT);
        continue;
    }
}
if (*(out_offset + i) == DARK) {
    if (*(out_offset + i-1) == WHITE && *(diff_offset + i-1) < -(*lstop)) {
        enqueue (&edgies, i, DARK);
        continue;
    }
    if (*(out_offset + i+1) == WHITE && *(diff_offset + i+1) < -(*lstop)) {
        enqueue (&edgies, i, DARK);
        continue;
    }
    if (*(out_offset + i-Nx) == WHITE && *(diff_offset + i-Nx) < -(*lstop)) {
        enqueue (&edgies, i, DARK);
        continue;
    }
    if (*(out_offset + i+Nx) == WHITE && *(diff_offset + i+Nx) < -(*lstop)) {
        enqueue (&edgies, i, DARK);
        continue;
    }
    if (*connected != 8)
        continue;
    if (*(out_offset + i-Nx-1) == WHITE && *(diff_offset + i-Nx-1) < -(*lstop)) {
        enqueue (&edgies, i, DARK);
        continue;
    }
    if (*(out_offset + i-Nx+1) == WHITE && *(diff_offset + i-Nx+1) < -(*lstop)) {
        enqueue (&edgies, i, DARK);
        continue;
    }
    if (*(out_offset + i+Nx-1) == WHITE && *(diff_offset + i+Nx-1) < -(*lstop)) {
        enqueue (&edgies, i, DARK);
        continue;
    }
    if (*(out_offset + i+Nx+1) == WHITE && *(diff_offset + i+Nx+1) < -(*lstop)) {
        enqueue (&edgies, i, DARK);
        continue;
    }
}
}

while(!isempty(edgies)) {
    dequeue (&edgies, &i, &approach);
    if (approach == LIGHT) {
        if (*(out_offset + i-1) == WHITE && *(diff_offset + i-1) > *ustop) {
            *(out_offset + i-1) = LIGHT;
            enqueue (&edgies, i-1, LIGHT);
        }
        if (*(out_offset + i+1) == WHITE && *(diff_offset + i+1) > *ustop) {
            *(out_offset + i+1) = LIGHT;
            enqueue (&edgies, i+1, LIGHT);
        }
        if (*(out_offset + i-Nx) == WHITE && *(diff_offset + i-Nx) > *ustop) {
            *(out_offset + i-Nx) = LIGHT;
            enqueue (&edgies, i-Nx, LIGHT);
        }
        if (*(out_offset + i+Nx) == WHITE && *(diff_offset + i+Nx) > *ustop) {
            *(out_offset + i+Nx) = LIGHT;
            enqueue (&edgies, i+Nx, LIGHT);
        }
        if (*connected != 8)
            continue;
        if (*(out_offset + i-Nx-1) == WHITE && *(diff_offset + i-Nx-1) > *ustop) {

```

```

        *(out_offset + i-Nx-1) = LIGHT;
        enqueue (&edgies, i-Nx-1, LIGHT);
    }
    if (*(out_offset + i-Nx+1) == WHITE && *(diff_offset + i-Nx+1) > *ustop) {
        *(out_offset + i-Nx+1) = LIGHT;
        enqueue (&edgies, i-Nx+1, LIGHT);
    }
    if (*(out_offset + i+Nx-1) == WHITE && *(diff_offset + i+Nx-1) > *ustop) {
        *(out_offset + i+Nx-1) = LIGHT;
        enqueue (&edgies, i+Nx-1, LIGHT);
    }
    if (*(out_offset + i+Nx+1) == WHITE && *(diff_offset + i+Nx+1) > *ustop) {
        *(out_offset + i+Nx+1) = LIGHT;
        enqueue (&edgies, i+Nx+1, LIGHT);
    }
}

if (approach == DARK) {
    if (*(out_offset + i-1) == WHITE && *(diff_offset + i-1) < -(*lstop)) {
        *(out_offset + i-1) = DARK;
        enqueue (&edgies, i-1, DARK);
    }
    if (*(out_offset + i+1) == WHITE && *(diff_offset + i+1) < -(*lstop)) {
        *(out_offset + i+1) = DARK;
        enqueue (&edgies, i+1, DARK);
    }
    if (*(out_offset + i-Nx) == WHITE && *(diff_offset + i-Nx) < -(*lstop)) {
        *(out_offset + i-Nx) = DARK;
        enqueue (&edgies, i-Nx, DARK);
    }
    if (*(out_offset + i+Nx) == WHITE && *(diff_offset + i+Nx) < -(*lstop)) {
        *(out_offset + i+Nx) = DARK;
        enqueue (&edgies, i+Nx, DARK);
    }
    if (*connected != 8)
        continue;
    if (*(out_offset + i-Nx-1) == WHITE && *(diff_offset + i-Nx-1) < -(*lstop)) {
        *(out_offset + i-Nx-1) = DARK;
        enqueue (&edgies, i-Nx-1, DARK);
    }
    if (*(out_offset + i-Nx+1) == WHITE && *(diff_offset + i-Nx+1) < -(*lstop)) {
        *(out_offset + i-Nx+1) = DARK;
        enqueue (&edgies, i-Nx+1, DARK);
    }
    if (*(out_offset + i+Nx-1) == WHITE && *(diff_offset + i+Nx-1) < -(*lstop)) {
        *(out_offset + i+Nx-1) = DARK;
        enqueue (&edgies, i+Nx-1, DARK);
    }
    if (*(out_offset + i+Nx+1) == WHITE && *(diff_offset + i+Nx+1) < -(*lstop)) {
        *(out_offset + i+Nx+1) = DARK;
        enqueue (&edgies, i+Nx+1, DARK);
    }
}
}
}

```



```

/*****
*   FUNCTION NAME : subtract_image
*   PURPOSE : to generate new image which is the difference between the
*             original image and the smoothed image
*   FUNCTIONS CALLED : none
*   INPUT EXPECTED : pointers to original image structure, smoothed image
*                   structure and output image structure
*   OUTPUT EXPECTED : image structure containing pixel by pixel difference
*   ALGORITHM : one forward scan is used and subtraction takes place pixel
*               by pixel
*   REMARKS: no nested loops are used to improve program efficiency
*
*****/
#include <stdio.h>
#include "structure.h"

void subtract_image( struct int_image *inp_img, struct int_image *smth_img,
                    struct int_image *diff_img )
{
    int Nx, Ny, i, total_pixels, *inp_offset, *smth_offset, *diff_offset;

    inp_offset = inp_img->array;
    smth_offset = smth_img->array;
    diff_offset = diff_img->array;

    Nx = diff_img->Nx = inp_img->Nx;
    Ny = diff_img->Ny = inp_img->Ny;
    total_pixels = Nx*Ny;

    /* Forward sweep */
    for(i=0; i<total_pixels; ++i)
        *(diff_offset + i) = *(inp_offset + i) - *(smth_offset + i);
}

```

```

/*****
*   NAME : Eyler Coates
*   PROGRAM NAME : threshold.c
*   PURPOSE : This program generates a moment-preserving threshold.
*             The details of the method are found in a paper:
*             Tsai, W., 1985, Moment-Preserving Threshold: A new Approach,
*             Computer Vision, Graphics, and Image Processing, 29, pp377-393
*   FUNCTIONS USED : input.c, viffwrite.c, find_hist.c, find_thresh.c,
*                   apply_thresh.c
*   INCLUDE FILES : stdio.h, constants.h, structure.h
*   INPUT EXPECTED : input image file in VIFF format, name
*                   of an output (mask) image file in VIFF format
*   OUTPUT EXPECTED : A Viff format image thresholded by a moment-preserving
*                   threshold.
*   REMARKS:
*
*****/
#include <stdio.h>
#include "structure.h"
#include "constants.h"
/*****
*   PROGRAM NAME : main()
*   PURPOSE : Input file names, call other functions, and report progress
*   FUNCTIONS CALLED : input.c, viffwrite.c, find_hist.c, find_thresh.c
*                   apply_thresh.c
*   INPUT EXPECTED : Name of an input image file in raw matrix format, name
*                   of an output image file in VIFF format
*   OUTPUT EXPECTED : A Viff format image thresholded by a moment-preserving
*                   threshold.
*   REMARKS:
*
*****/
main( )
{
    char infile[40], mskfile[40], command[80], answer[2];
    float hist[LEVELS], m1, m2, m3, threshold;
    struct int_image inp_img, out_img, msk_img;

    printf("\nEnter the name of the input image file : ");
    scanf("%s", infile);

    printf("\nEnter the name of the output thresholded image file : ");
    scanf("%s", mskfile);

    printf("\nReading in a VIFF byte image file.\n");
    viffread( &inp_img, infile );

    printf("\nFinding the histogram and first 3 moments of input image.\n");
    find_hist( &inp_img, hist, &m1, &m2, &m3 );

    printf("\nFinding the moment-preserving threshold of the input image.\n");
    find_thresh( &m1, &m2, &m3, &threshold, hist );

    printf("\n The threshold for the image is %f\n", threshold);

    printf("\nApplying threshold to the input image to produce output.\n");
    apply_thresh( &threshold, &inp_img, &msk_img );

    printf("\nWriting a thresholded image in VIFF byte format.\n");
    viffwrite( &msk_img, mskfile );

    printf("\nDisplay images? (y/n) : ");
    scanf("%s", answer);

    if (answer[0] != 'n') {
        printf("\nAbout to display input and thresholded images ...\n");
        printf(command, "editimage -i %s &", infile);
        system(command);
        printf(command, "editimage -i %s &", mskfile);
        system(command);
    }
    printf("\nDone.\n\n");
}

```

```

/*****

This is a header file timer.h containing declarations and
code for timing C-code

*****/

#include <time.h>

#define      MAXSTRING    100

struct timekeeper {
    clock_t    begin_clock, save_clock;
    time_t     begin_time, save_time;
};

typedef      struct timekeeper TIMEKEEPER;

static      TIMEKEEPER tk;

void start_time(void)
{
    tk.begin_clock = tk.save_clock = clock();
    tk.begin_time = tk.save_time = time(NULL);
}

double prn_time(void)
{
    char s1[MAXSTRING], s2[MAXSTRING];
    int  field_width, n1, n2;
    double user_time, real_time;

    user_time = (clock() - tk.save_clock);
    real_time = difftime(time(NULL), tk.save_time);
    tk.save_clock = clock();
    tk.save_time = time(NULL);

    n1 = sprintf(s1, "%.1f", user_time);
    n2 = sprintf(s2, "%.1f", real_time);
    field_width = (n1 > n2) ? n1 : n2;
    printf("%s%.1f%s\n%s%.1f%s\n\n",
        "User time: ", field_width, user_time, " seconds",
        "Real time: ", field_width, real_time, " seconds");
    return user_time;
}

```

```

/*****
*   NAME : Eyler Coates
*   PROGRAM NAME : zmed3.c
*   PURPOSE : This program compares the gray levels of the same pixel from
*             three images and computes the median of the three. The process
*             is repeated for every pixel location. So the program can be
*             used to compute the median image between 3 other images.
*   FUNCTIONS USED : viffread.c, viffwrite.c
*   INCLUDE FILES : stdio.h, time.h, constants.h, structure.h
*   INPUT EXPECTED : Names of 3 input image files in VIFF format, name of an
*                   output image file in VIFF format
*   OUTPUT EXPECTED : A Viff format image that is the median of three other
*                   images.
*   REMARKS:
*
*****/
#include <stdio.h>
#include "structure.h"
#include <time.h>
#include "constants.h"

/*****
*   PROGRAM NAME : main()
*   PURPOSE : Input file names, call other functions, and report progress
*   FUNCTIONS CALLED : viffread.c, viffwrite.c
*   INPUT EXPECTED : Names of 3 input image files in VIFF format, name of an
*                   output image file in VIFF format
*   OUTPUT EXPECTED : A Viff format image that is the median of three other
*                   images.
*   REMARKS:
*
*****/
main ( )
{
    char infile1[40], infile2[40], infile3[40],
        outfile[40], command[80], answer[2];
    int Nx, Ny, i, j, k, temp, grey[4], total_pixels, *inp1_offset,
        *inp2_offset, *inp3_offset, *out_offset, pixel_count, divisor;
    double user_time, real_time;
    clock_t begin_clock;
    time_t begin_time;
    struct int_image inp1_img, inp2_img, inp3_img, out_img;

    printf("\nEnter the name of the first input image file : ");
    scanf("%s", infile1 );

    printf("\nEnter the name of the second input image file : ");
    scanf("%s", infile2 );

    printf("\nEnter the name of the third input image file : ");
    scanf("%s", infile3 );

    printf("\nEnter the path and name of the output image file : ");
    scanf("%s", outfile );

    printf("\nReading in a VIFF byte image file.\n");
    viffread( &inp1_img, infile1 );

    printf("\nReading in a VIFF byte image file.\n");
    viffread( &inp2_img, infile2 );

    printf("\nReading in a VIFF byte image file.\n");
    viffread( &inp3_img, infile3 );

    begin_clock = clock();
    begin_time = time(NULL);

    printf("\nFinding the median of the three images.\n");

    inp1_offset = inp1_img.array;
    inp2_offset = inp2_img.array;
    inp3_offset = inp3_img.array;
    out_offset = out_img.array;

    Nx = out_img.Nx = inp1_img.Nx;
    Ny = out_img.Ny = inp1_img.Ny;

```


APPENDIX B - IMAGE STATISTICS

Table B. 1 Parameters for Defect Detection Algorithm for Spin-Echo Images

<i>Frame</i>	<i>Upper Normal Limit of Variation</i>	<i>Lower Normal Limit of Variation</i>	<i>Upper Region Growing Stopping Limit</i>	<i>Lower Region Growing Stopping Limit</i>
5	38	39	11	7
6	41	39	12	6
7	51	39	11	6
8	43	36	11	5
9	42	36	12	7
10	43	35	9	5
11	46	37	11	6
12	44	43	11	6
13	53	46	9	8
14	47	39	8	8
15	41	35	10	5
16	45	34	7	6
17	46	34	7	5
18	53	34	10	8
19	46	34	20	7
20	47	34	21	7
21	51	35	26	10
22	58	48	32	11
23	60	43	27	8
24	41	36	22	7
25	41	38	18	5
26	42	39	19	7
27	45	42	27	8
28	48	45	28	8
29	45	40	24	7
30	42	41	25	5
31	49	42	27	8
32	46	44	24	9
33	43	47	28	9
34	49	43	23	16
35	45	42	24	5
36	54	41	24	8
37	50	38	26	8
38	46	38	20	11
39	50	42	25	10

table continued on next page

<i>Frame</i>	<i>Upper Normal Limit of Variation</i>	<i>Lower Normal Limit of Variation</i>	<i>Upper Region Growing Stopping Limit</i>	<i>Lower Region Growing Stopping Limit</i>
40	50	40	24	7
41	50	45	23	10
42	49	43	22	12
43	51	40	24	11
44	52	46	17	12
45	48	39	18	6
46	51	43	21	11
47	53	47	22	11
48	50	50	22	14
49	52	49	25	12
50	54	54	30	9
51	51	41	28	13
52	54	55	38	9
53	61	44	42	9
54	55	45	41	9
55	56	52	34	11
56	61	48	38	9
57	56	49	39	6
58	45	40	27	7
59	49	36	20	7
The connectivity for region growing for all frames was 4-connectivity				

Table B. 2 Image Statistics of Spin-Echo Images

<i>Frame</i>	<i>Thres- hold</i>	<i>Variance</i>	<i>Mean</i>	<i>Std.Dev.</i>	<i>RMS</i>	<i>Skewness</i>	<i>Kurtosis</i>	<i>Entropy</i>	<i>Contrast</i>
5	64.5	142.4706	85.7573	15.7888	87.1986	.316029	-.495353	5.96664	7603.59
6	67.5	151.3524	89.4193	15.8085	90.8059	.265269	-.388572	5.98661	8245.71
7	70.5	161.289	92.7555	15.8945	94.1074	.25573	-.191533	6.00746	8856.2
8	70.5	169.3017	93.1378	16.0177	94.505	.251054	-.092177	6.02435	8931.2
9	71.5	178.181	94.9599	16.1595	96.325	.213531	-.0397235	6.03877	9278.5
10	72.5	188.8125	96.2415	16.5051	97.6464	.229599	.154903	6.06838	9534.83
11	74.5	190.5412	97.9643	16.5706	99.3558	.265941	.168392	6.07597	9871.58
12	75.5	203.8824	99.0131	16.9718	100.457	.319127	.270493	6.10765	10091.6
13	76.5	204.7678	99.9419	17.0513	101.386	.356854	.535063	6.11096	10279.1
14	76.5	203.9415	100.361	17.0207	101.794	.363587	.56797	6.10897	10361.9
15	75.5	200.589	99.007	16.9214	100.443	.334519	.465787	6.10091	10088.7
16	75.5	199.4306	99.508	16.9359	100.939	.357529	.51353	6.1008	10188.7
17	76.5	207.3448	100.805	17.221	102.265	.385814	.611043	6.12302	10458.1
18	77.5	214.5742	101.448	17.561	102.957	.378658	.623298	6.14942	10600.1
19	75.5	219.5715	99.6579	18.0685	101.283	.31353	.638016	6.18877	10258.2
20	74.5	233.9009	98.2818	20.4685	100.39	-.322002	2.00024	6.32457	10078.1
21	74.5	240.2256	98.5148	21.7514	100.887	-.772768	3.47842	6.347	10178.3
22	75.5	239.2168	99.7965	21.0742	101.997	-.721083	3.64347	6.30886	10403.4
23	75.5	239.2078	99.4704	20.4154	101.544	-.486099	2.64294	6.31077	10311.1
24	75.5	236.3634	99.7865	19.6067	101.694	-.129671	1.03884	6.29823	10341.7
25	75.5	229.6513	99.9992	18.9452	101.778	.013676	.978451	6.25841	10358.7
26	75.5	231.9225	99.6793	18.706	101.419	-.173348	1.82636	6.22132	10285.9
27	74.5	238.3953	98.0598	18.8193	99.8493	-.402404	2.52705	6.20984	9969.88
28	73.5	243.5051	97.6213	19.0568	99.4638	-.541005	2.99429	6.20864	9893.06
29	74.5	230.2408	97.843	18.4986	99.5763	-.391355	2.5217	6.18653	9915.44
30	72.5	215.1833	96.5822	18.6802	98.372	-.507031	2.36609	6.19051	9677.05
31	70.5	204.7784	93.6953	18.3931	95.4835	-.576139	2.27528	6.17114	9117.09
32	85.5	289.9327	115.054	21.5802	117.06	-.493847	2.11527	6.41207	13703.1
33	86.5	289.5156	116.227	20.8502	118.082	-.26536	1.69856	6.38038	13943.3
34	85.5	288.0488	115.602	20.4139	117.391	-.092612	1.26411	6.36244	13780.6
35	88.5	289.155	117.395	20.0797	119.1	-.00386417	1.07246	6.34421	14184.9
36	89.5	293.2635	118.216	20.277	119.943	.0766541	.860019	6.36399	14386.2
37	90.5	289.4144	119.299	20.0847	120.977	.0970716	.86.124	6.35033	14635.5
38	89.5	276.504	118.532	19.6074	120.143	.0760638	.859137	6.31551	14434.2
39	89.5	284.4292	118.47	19.6727	120.093	.0552137	1.10994	6.31692	14422.2
40	89.5	285.5954	118.157	19.78	119.801	.125901	1.24164	6.32297	14352.3
41	89.5	289.2903	118.338	19.7185	119.97	.0747469	1.30283	6.31888	14392.8
42	88.5	300.7373	117.719	19.9728	119.401	.00627552	1.5964	6.33007	14256.6
43	89.5	295.7591	118.046	19.7767	119.691	.107947	1.76987	6.3125	14325.8
44	89.5	293.1606	118.175	19.8413	119.829	.120655	1.68879	6.319	14359.1
45	88.5	292.2869	117.546	19.7846	119.199	.0636504	1.70644	6.31516	14208.4
46	88.5	291.5114	116.606	19.72	118.262	.0236869	1.79074	6.30538	13985.9
47	88.5	295.6955	116.802	19.8385	118.474	.0494781	2.1173	6.30745	14036.2
48	88.5	302.4521	117.567	19.9367	119.245	.0147954	1.98954	6.31836	14219.5
49	87.5	309.4559	116.862	19.8898	118.542	-.0808089	2.14339	6.30961	14052.2
50	86.5	327.9633	114.675	20.7021	116.529	-.398934	2.99426	6.33709	13579
51	84.5	328.3883	113.531	20.2093	115.315	-.294133	2.59663	6.31661	13297.6
52	84.5	336.9727	112.992	21.412	115.003	-.594428	2.89081	6.3733	13225.6
53	83.5	337.9923	111.835	22.3666	114.049	-.940822	3.98967	6.38815	13007.3
54	81.5	320.5953	109.983	21.619	112.088	-.833995	3.7109	6.35526	12563.7
55	81.5	306.5762	108.086	21.1371	110.133	-.888689	4.53283	6.30158	12129.3
56	80.5	303.1737	106.938	21.2348	109.026	-.946559	4.78129	6.29689	11886.6
57	79.5	289.1402	104.457	21.0313	106.553	-.840395	4.13935	6.30658	11353.6

table continued on next page

<i>Frame</i>	<i>Thres- hold</i>	<i>Variance</i>	<i>Mean</i>	<i>Std.Dev.</i>	<i>RMS</i>	<i>Skewness</i>	<i>Kurtosis</i>	<i>Entropy</i>	<i>Contrast</i>
58	76.5	257.1385	100.774	20.0748	102.754	-.483841	3.0065	6.28288	10558.4
59	70.5	213.7431	94.2466	19.0755	96.1575	-.134199	1.99781	6.23913	9246.27

Table B. 3 Prediction of Parameters for Spin-Echo Images

<i>Frame</i>	<i>Upper Normal Limit of Variation</i>		<i>Lower Normal Limit of Variation</i>		<i>Upper Region Growing Stopping Limit</i>		<i>Lower Region Growing Stopping Limit</i>	
	Est't.	Error	Est't.	Error	Est't.	Error	Est't.	Error
5	59	21	38	-1	14	3	3	-4
6	52	11	37	-2	13	1	3	-3
7	48	-3	36	-3	12	1	3	-3
8	47	4	36	0	13	2	3	-2
9	45	3	35	-1	12	0	5	-2
10	45	2	35	0	13	4	5	0
11	47	1	36	-1	16	5	6	0
12	45	1	38	-5	16	5	7	1
13	45	-8	40	-6	12	3	9	1
14	47	0	38	-1	10	2	8	0
15	47	6	38	3	10	0	7	2
16	46	1	37	3	9	2	8	2
17	47	1	38	4	9	2	9	4
18	47	-6	38	4	10	0	7	-1
19	43	-3	37	3	11	-9	7	0
20	48	1	34	0	21	0	3	-4
21	57	6	34	-1	24	-2	7	-3
22	61	3	44	-4	32	0	9	-2
23	52	-8	41	-2	28	1	5	-3
24	43	2	34	-2	19	-3	3	-4
25	42	1	35	-3	16	-2	4	-1
26	42	0	37	-2	21	2	5	-2
27	46	1	42	0	25	-2	5	-3
28	48	0	43	-2	27	-1	7	-1
29	47	2	44	4	24	0	6	-1
30	43	1	37	-4	22	-3	7	2
31	45	-4	43	1	26	-1	7	-1
32	44	-2	40	-4	31	7	10	1
33	46	3	42	-5	29	1	10	1
34	46	-3	39	-4	25	2	11	-5
35	48	3	44	2	23	-1	10	5
36	50	-4	46	5	23	-1	10	2
37	49	-1	47	9	20	-6	10	2
38	49	3	44	6	19	-1	10	-1
39	51	1	47	5	20	-5	11	1
40	50	0	49	9	19	-5	10	3

table continued on next page

<i>Frame</i>	<i>Upper Normal Limit of Variation</i>		<i>Lower Normal Limit of Variation</i>		<i>Upper Region Growing Stopping Limit</i>		<i>Lower Region Growing Stopping Limit</i>	
	Est't.	Error	Est't.	Error	Est't.	Error	Est't.	Error
41	50	0	50	5	19	-4	10	-1
42	47	-2	46	3	21	-1	10	-2
43	53	2	45	5	20	-4	11	-1
44	52	0	42	-4	21	4	11	-1
45	52	4	43	4	22	4	11	5
46	49	-2	41	-2	22	1	11	-1
47	50	-3	41	-6	22	0	12	0
48	52	2	44	-6	24	2	11	-4
49	51	-1	45	-4	23	-2	11	-1
50	52	-2	49	-5	32	2	11	3
51	50	-1	48	7	25	-3	10	-2
52	54	0	51	-4	31	-7	10	0
53	59	-2	54	10	32	-10	11	2
54	54	-1	51	6	29	-12	10	0
55	57	1	53	1	30	-4	10	0
56	57	-4	53	5	30	-8	11	0
57	58	2	51	2	32	-7	9	-1
58	52	7	44	4	27	0	8	0
59	45	-4	38	2	21	1	7	-1

Table B. 4 Comparison of Defects Results of Spin-Echo Defect Detection Algorithm Versus Ordinary Thresholding

<i>Frame</i>	<i>Light Pixels</i>	<i>Dark Pixels</i>	<i>High Threshold</i>	<i>Low Threshold</i>	<i>Undetected Defect Pixels</i>	<i>False Alarm Pixels</i>
5	220	1233	120	65	1006	1037
6	172	1177	124	68	892	968
7	75	1253	135	71	873	812
8	217	1376	130	72	1021	1019
9	190	1351	132	73	942	898
10	347	1430	131	75	1120	1184
11	216	1333	137	76	974	1011
12	305	1393	138	77	1030	1120
13	260	1275	141	77	947	915
14	241	1328	142	78	985	1025
15	398	1595	135	78	1263	1306
16	376	1590	137	78	1277	1223
17	385	1623	139	79	1283	1241
18	321	1460	142	79	1085	1092
19	269	1645	142	78	1092	1175
20	313	1907	141	76	1126	1087
21	181	1636	148	75	869	930
22	249	1548	145	76	877	885
23	205	1675	146	77	918	1096
24	337	1738	142	77	1021	1064
25	434	1949	139	78	1224	1143
26	366	1727	139	78	1080	1138
27	342	1967	137	78	1167	1261
28	419	1847	135	77	1106	1141
29	397	1979	135	78	1121	1188
30	310	2120	136	78	1151	1154
31	192	2225	135	75	1083	1036
32	438	1996	156	92	1066	986
33	367	2026	158	93	1043	1032
34	326	1482	159	89	759	713
35	318	1879	161	94	1096	1077
36	187	1696	168	93	970	958
37	253	1501	165	93	877	811
38	272	1414	163	92	811	741
39	200	1470	166	93	804	819
40	327	1707	162	94	1089	1069
41	303	1423	162	92	880	804
42	292	1484	162	92	799	785

table continued on next page

<i>Frame</i>	<i>Light Pixels</i>	<i>Dark Pixels</i>	<i>High Threshold</i>	<i>Low Threshold</i>	<i>Undetected Defect Pixels</i>	<i>False Alarm Pixels</i>
43	246	1392	164	92	764	753
44	268	1415	162	92	786	784
45	299	1761	160	94	994	1039
46	252	1441	161	91	763	790
47	233	1469	162	91	796	755
48	271	1225	162	90	663	649
49	183	1438	166	91	729	745
50	203	1878	162	91	943	905
51	239	1649	159	89	725	791
52	185	1806	162	88	781	762
53	151	1938	164	88	857	835
54	214	1805	157	86	821	861
55	211	1816	153	85	834	822
56	142	1616	157	83	705	711
57	311	1652	145	81	896	910
58	339	1766	141	78	1141	1075
59	163	1950	140	73	1179	1084
<i>Averages</i>	269	1623	148	82	966	968

Table B. 5 Parameters for Defect Detection Algorithm for Echo-Planar Images

<i>Frame</i>	<i>Upper Normal Limit of Variation</i>	<i>Lower Normal Limit of Variation</i>	<i>Upper Region Growing Stopping Limit</i>	<i>Lower Region Growing Stopping Limit</i>	<i>Pith Area Lower Normal Limit of Variation</i>	<i>Pith Area Lower Region Growing Stopping Limit</i>
5	95	55	15	15	49	15
6	90	61	15	14	49	17
7	84	63	15	14	63	14
8	85	69	15	10	64	13
9	92	65	16	15	55	15
10	95	59	15	15	59	15
11	85	60	16	15	60	15
12	90	65	15	13	45	16
13	86	66	15	15	51	15
14	87	72	12	13	57	16
15	95	74	15	12	58	12
16	86	83	15	12	49	16
17	92	70	13	13	61	17
18	93	71	14	13	63	16
19	91	62	15	13	62	13
20	85	56	16	14	56	14
21	82	64	17	11	64	11
22	89	71	17	13	71	13
23	89	63	15	13	63	13
24	86	62	14	14	54	16
25	88	64	6	15	64	15
26	85	65	12	14	65	14
27	82	79	12	12	66	15
28	85	73	15	7	61	15
29	80	67	11	13	67	13
30	84	67	12	15	67	15
31	82	73	13	14	73	14
32	86	65	17	17	65	17
33	90	61	21	21	61	21
34	93	63	14	12	46	18
35	97	68	12	8	48	14
36	89	62	13	12	54	15
37	88	66	13	11	55	17
38	94	63	12	9	51	18
39	93	57	12	10	47	21

table continued on next page

<i>Frame</i>	<i>Pith Area</i>					
	<i>Upper Normal Limit of Variation</i>	<i>Lower Normal Limit of Variation</i>	<i>Upper Region Growing Stopping Limit</i>	<i>Lower Region Growing Stopping Limit</i>	<i>Pith Area Lower Normal Limit of Variation</i>	<i>Lower Region Growing Stopping Limit</i>
40	91	59	12	10	59	10
41	87	61	13	11	43	18
42	91	61	16	12	41	21
43	86	65	12	10	41	14
44	86	58	12	10	36	18
45	91	54	12	12	48	18
46	90	62	11	12	48	21
47	94	58	11	15	40	20
48	88	58	12	20	51	14
49	92	64	24	15	50	18
50	95	56	16	16	56	16
51	92	63	25	15	40	20
52	92	56	12	14	48	18
53	90	55	12	13	41	15
54	94	51	15	18	51	18
55	85	56	6	8	56	8
56	84	56	15	15	56	15
57	83	58	18	18	58	18
58	80	54	15	15	54	15
59	70	51	12	13	51	13

Table B. 6 Image Statistics for Echo-Planar Images

<i>Frame</i>	<i>Thres- hold</i>	<i>Variance</i>	<i>Mean</i>	<i>Std. Dev.</i>	<i>RMS</i>	<i>Skewness</i>	<i>Kurtosis</i>	<i>Entropy</i>	<i>Contrast</i>
5	75.5	451.5141	130.433	30.7281	134.004	0.458649	0.134059	6.94566	17957
6	79.5	476.3557	135.523	31.5395	139.144	0.469723	0.108775	6.98478	19361.2
7	74.5	463.6523	128.505	29.8395	131.924	0.45701	0.279061	6.90657	17403.9
8	75.5	487.4789	128.752	30.1959	132.245	0.447075	0.215212	6.92377	17488.7
9	78.5	511.0821	136.582	31.9809	140.276	0.465672	0.09175	7.0006	19677.4
10	77.5	527.3595	136.877	31.9764	140.562	0.543717	0.173787	7.00064	19757.7
11	75.5	496.508	129.44	30.1182	132.897	0.512133	0.410539	6.90668	17661.7
12	68.5	490.4464	130.147	30.0157	133.563	0.388557	0.573868	6.91009	17839.2
13	65.5	524.5284	136.379	31.4965	139.969	0.404103	0.334403	6.98839	19591.2
14	69.5	516.6325	135.125	31.4544	138.737	0.49539	0.445125	6.98652	19248
15	67.5	485.5269	128.161	29.5486	131.523	0.488281	0.698699	6.89141	17298.3
16	65.5	496.8986	128.874	29.6017	132.23	0.488281	0.749502	6.89138	17484.8
17	65.5	519.0647	134.066	30.9221	137.586	0.434474	0.657291	6.9572	18929.8
18	64.5	517.7714	136.209	30.8535	139.659	0.41814	0.796906	6.94967	19504.7
19	66.5	489.1444	131.457	29.853	134.804	0.509176	0.939877	6.89687	18172
20	65.5	497.8709	128.851	32.9493	132.997	-0.06349	1.51314	7.02124	17688.1
21	67.5	524.6932	130.816	35.3059	135.496	-0.27993	1.71958	7.09084	18359.1
22	67.5	520.7023	131.752	34.2732	136.137	-0.12244	1.40624	7.07747	18533.2
23	67.5	203.1809	131.937	33.4626	136.114	-0.11315	1.28367	7.05415	18527.1
24	64.5	530.4955	135.38	33.9425	139.57	-0.0669	0.804162	7.09487	19479.8
25	68.5	529.5161	141.461	34.1656	145.528	-0.01459	0.402537	7.11842	21178.4
26	71.5	523.2209	144.121	33.6428	147.995	0.020161	0.278771	7.10023	21902.6
27	66.5	533.726	142.75	32.8318	146.475	0.012892	0.241889	7.06585	21454.9
28	66.5	522.2352	144.209	32.9876	147.933	0.008666	0.105132	7.07483	21884.3
29	72.5	568.5515	148.112	34.7142	152.126	0.033988	-0.01948	7.14684	23142.2
30	74.5	589.1495	149.369	36.4592	153.754	-0.07856	-0.03892	7.21864	23640.2
31	73.5	610.4246	149.668	38.627	154.572	-0.31396	0.305274	7.29021	23892.4
32	74.5	565.7193	142.802	40.5888	148.458	-0.29396	0.069463	7.35881	22039.9
33	75.5	572.4235	144.037	39.9161	149.465	-0.21718	-0.12626	7.34126	22339.8
34	70.5	475.793	135.104	33.2916	139.145	-0.03331	0.068956	7.08225	19361.2
35	68.5	480.7831	137.336	32.599	141.152	0.032741	-0.0392	7.05695	19923.9
36	68.5	468.5071	134.377	32.1967	138.18	0.051895	-0.10916	7.03987	19093.8
37	70.5	464.8221	136.534	32.4607	140.34	0.071944	-0.08414	7.05259	19695.2
38	59.5	412.3227	123.882	28.3804	127.091	0.180301	0.053077	6.85692	16152.1
39	61.5	416.2172	124.862	28.7686	128.133	0.171134	0.154594	6.87579	16418
40	65.5	411.2655	129.231	29.9051	132.646	0.177126	0.118328	6.9284	17594.9
41	64.5	423.6854	129.381	30.5145	132.931	0.142539	-0.01265	6.96013	17670.6
42	58.5	386.8726	117.287	27.8176	120.54	0.178904	0.303476	6.82829	14530
43	57.5	381.6148	117.034	27.4481	120.21	0.256427	0.244937	6.80486	14450.4
44	63.5	391.354	123.798	28.8938	127.125	0.216832	0.115351	6.88127	16160.8
45	59.5	398.9648	123.759	29.011	127.114	0.157424	0.094222	6.88837	16157.9
46	56.5	372.9756	114.494	26.9856	117.631	0.233775	0.289985	6.78509	13837.1
47	59.5	387.2335	114.566	27.2476	117.761	0.259077	0.221192	6.79725	13867.8
48	61.5	394.6928	119.513	28.7329	122.919	0.191681	0.108737	6.87311	15109
49	61.5	388.1727	118.873	28.7363	122.297	0.153318	0.048065	6.8753	14956.6

table continued on next page

<i>Frame</i>	<i>Thres- hold</i>	<i>Variance</i>	<i>Mean</i>	<i>Std. Dev.</i>	<i>RMS</i>	<i>Skewness</i>	<i>Kurtosis</i>	<i>Entropy</i>	<i>Contrast</i>
50	58.5	394.1252	115.181	28.6786	118.697	0.0263	0.320764	6.87383	14089
51	58.5	385.5923	114.74	28.6878	118.272	0.070426	0.158654	6.87479	13988.3
52	60.5	404.6361	113.791	30.6857	117.855	-0.05603	0.396514	6.96479	13889.9
53	59.5	395.0419	112.724	31.4307	117.023	-0.19681	0.612152	6.98721	13694.4
54	55.5	390.313	111.245	30.2544	115.286	-0.21279	0.581166	6.93761	13290.8
55	55.5	377.8	110.294	29.5757	114.19	-0.23864	0.582343	6.90597	13039.4
56	59.5	369.7896	115.547	30.4381	119.489	-0.22709	0.495934	6.94855	14277.5
57	57.5	360.5996	112.941	30.134	116.892	-0.22628	0.52281	6.93386	13663.7
58	57.5	317.7261	111.299	29.2127	115.069	-0.1812	0.390016	6.8922	13240.8
59	56.5	301.7111	104.064	30.8816	108.55	0.062564	-0.04717	6.98053	11783

Table B. 7 Prediction of Parameters for Echo-Planar Images

Frame	<i>Upper Normal Limit of Variation</i>		<i>Lower Normal Limit of Variation</i>		<i>Upper Region Growing Stopping Limit</i>		<i>Lower Region Growing Stopping Limit</i>		<i>Pith Area Lower Normal Limit of Variation</i>		<i>Pith Area Lower Region Growing Stopping Limit</i>	
	<i>Est.</i>	<i>Err.</i>	<i>Est.</i>	<i>Err.</i>	<i>Est.</i>	<i>Err.</i>	<i>Est.</i>	<i>Err.</i>	<i>Est.</i>	<i>Err.</i>	<i>Est.</i>	<i>Err.</i>
5	92	-3	65	10	16	1	14	-1	49	0	14	-1
6	89	-1	64	3	15	0	14	0	51	2	13	-4
7	96	12	59	-4	15	0	13	-1	50	-13	13	-1
8	101	16	59	-10	15	0	13	3	49	-15	14	1
9	96	4	61	-4	15	-1	15	0	53	-2	15	0
10	96	1	63	4	15	0	15	0	47	-12	16	1
11	98	13	63	3	15	-1	13	-2	47	-13	14	-1
12	95	5	65	0	14	-1	12	-1	52	7	15	-1
13	94	8	68	2	13	-2	13	-2	48	-3	18	3
14	95	8	65	-7	12	0	15	2	40	-17	18	2
15	100	5	62	-12	11	-4	14	2	46	-12	17	5
16	100	14	65	-18	12	-3	14	2	41	-8	21	5
17	94	2	69	-1	12	-1	15	2	47	-14	21	4
18	90	-3	69	-2	11	-3	11	-2	50	-13	19	3
19	89	-2	66	4	11	-4	12	-1	50	-12	17	4
20	93	8	63	7	11	-5	17	3	60	4	15	1
21	88	6	63	-1	15	-2	16	5	69	5	15	4
22	93	4	63	-8	13	-4	19	6	54	-17	17	4
23	89	0	62	-1	13	-2	16	3	58	-5	13	0
24	93	7	66	4	12	-2	15	1	53	-1	15	-1
25	91	3	66	2	10	4	11	-4	58	-6	13	-2
26	83	-2	66	1	9	-3	7	-7	66	1	12	-2
27	87	5	70	-9	10	-2	6	-6	63	-3	14	-1
28	89	4	66	-7	11	-4	8	1	66	5	13	-2
29	90	10	67	0	10	-1	12	-1	71	4	13	0
30	94	10	63	-4	9	-3	16	1	74	7	14	-1
31	94	12	63	-10	10	-3	17	3	75	2	14	0
32	83	-3	57	-8	16	-1	13	-4	62	-3	12	-5
33	89	-1	59	-2	16	-5	15	-6	61	0	15	-6
34	88	-5	59	-4	16	2	14	2	51	5	16	-2
35	89	-8	61	-7	13	1	14	6	57	9	16	2
36	89	0	62	0	16	3	15	3	52	-2	16	1
37	86	-2	62	-4	15	2	16	5	56	1	16	-1

table continued on next page

Frame	Upper Normal Limit of Variation		Lower Normal Limit of Variation		Upper Region Growing Stopping Limit		Lower Region Growing Stopping Limit		Pith Area Lower Normal Limit of Variation		Pith Area Lower Region Growing Stopping Limit	
	Est. Err.		Est. Err.		Est. Err.		Est. Err.		Est. Err.		Est. Err.	
	Est.	Err.	Est.	Err.	Est.	Err.	Est.	Err.	Est.	Err.	Est.	Err.
38	89	-5	66	3	17	5	13	4	50	-1	17	-1
39	91	-2	66	9	17	5	12	2	49	2	18	-3
40	88	-3	62	3	15	3	13	3	50	-9	17	7
41	89	2	62	1	15	2	15	4	46	3	19	1
42	89	-2	61	0	14	-2	13	1	50	9	16	-5
43	88	2	62	-3	14	2	13	3	49	8	16	2
44	90	4	60	2	14	2	13	3	49	13	17	-1
45	92	1	63	9	14	2	12	0	45	-3	17	-1
46	81	-9	63	1	15	4	13	1	51	3	16	-5
47	82	-12	63	5	15	4	13	-2	50	10	16	-4
48	87	-1	61	3	14	2	12	-8	46	-5	17	3
49	87	-5	60	-4	13	-11	10	-5	44	-6	17	-1
50	87	-8	59	3	10	-6	12	-4	54	-2	19	3
51	86	-6	58	-5	10	-15	12	-3	51	11	20	0
52	88	-4	57	1	14	2	14	0	46	-2	23	5
53	86	-4	56	1	13	1	13	0	50	9	20	5
54	84	-10	59	8	12	-3	12	-6	44	-7	20	2
55	83	-2	57	1	11	5	11	3	45	-11	20	12
56	88	4	54	-2	11	-4	11	-4	47	-9	20	5
57	88	5	54	-4	11	-7	13	-5	46	-12	21	3
58	87	7	50	-4	10	-5	14	-1	46	-8	20	5
59	87	17	46	-5	12	0	21	8	50	-1	28	15

Table B. 8 SNR of Echo-Planar Images (Unprocessed and General Processing Methods)

<i>Frame</i>	<i>Unprocessed</i>	<i>Despeckled</i>	<i>XYMedian</i>	<i>Average</i>	<i>Count</i>	<i>Y_j</i>
3	0.320478	0.30637	0.308494	0.281859	4	1.217201
4	0.448294	0.432052	0.436056	0.438421	4	1.754823
5	0.419348	0.405332	0.409026	0.385605	4	1.619312
6	0.455324	0.437327	0.440917	0.429574	4	1.763142
7	0.504522	0.498313	0.50216	0.46803	4	1.973025
8	0.476294	0.470071	0.472836	0.439318	4	1.85852
9	0.490019	0.477493	0.482503	0.437812	4	1.887826
10	0.482039	0.474805	0.480002	0.495104	4	1.93195
11	0.549042	0.544453	0.477846	0.52064	4	2.091983
12	0.55452	0.550438	0.551528	0.531673	4	2.188159
13	0.501126	0.494297	0.500221	0.451005	4	1.946649
14	0.565233	0.556519	0.558617	0.527527	4	2.207897
15	0.529422	0.523398	0.529779	0.539356	4	2.121955
16	0.575581	0.568488	0.574037	0.531647	4	2.249753
17	0.56394	0.554527	0.556191	0.482423	4	2.157081
18	0.575207	0.563433	0.56646	0.537034	4	2.242134
19	0.582443	0.573383	0.578363	0.557187	4	2.291376
20	0.563076	0.560734	0.572499	0.599703	4	2.296012
21	0.586283	0.576111	0.581956	0.577159	4	2.321508
22	0.580029	0.565842	0.570916	0.541548	4	2.258335
23	0.594632	0.583063	0.590347	0.552717	4	2.32076
24	0.453717	0.455918	0.461684	0.443816	4	1.815135
25	0.447262	0.446512	0.449162	0.490201	4	1.833138
26	0.466796	0.458453	0.466544	0.486703	4	1.878496
27	0.42608	0.420479	0.425316	0.423191	4	1.695067
28	0.422272	0.414915	0.416725	0.421592	4	1.675504
29	0.400538	0.395409	0.399344	0.410873	4	1.606165
30	0.463104	0.449974	0.454069	0.442708	4	1.809855
31	0.475536	0.460198	0.463457	0.465103	4	1.864294
32	0.447608	0.436089	0.438473	0.464884	4	1.787055
33	0.468819	0.453113	0.457818	0.472723	4	1.852473
34	0.475338	0.470187	0.474694	0.459694	4	1.879913
35	0.445165	0.435151	0.438806	0.442516	4	1.761639
36	0.456837	0.455164	0.457739	0.462567	4	1.832307
37	0.518897	0.507625	0.511265	0.476921	4	2.014707
38	0.453544	0.458413	0.463257	0.481529	4	1.856742
39	0.44377	0.454482	0.461021	0.472599	4	1.831872
40	0.552102	0.536913	0.538784	0.499682	4	2.12748

table continued on next page

<i>Frame</i>	<i>Unprocessed</i>	<i>Despeckled</i>	<i>XYMedian</i>	<i>Average</i>	<i>Count</i>	<i>Y_j</i>
41	0.4735	0.466324	0.470392	0.464743	4	1.874959
42	0.587616	0.577521	0.582693	0.542529	4	2.29036
43	0.438782	0.445775	0.451029	0.4311	4	1.766687
44	0.479089	0.478464	0.483219	0.487009	4	1.927781
45	0.437582	0.435595	0.440397	0.450907	4	1.764481
46	0.558469	0.553332	0.557263	0.524056	4	2.19312
47	0.499244	0.495734	0.497785	0.475228	4	1.967992
48	0.558068	0.544198	0.545738	0.514284	4	2.162288
49	0.543405	0.530739	0.531227	0.472366	4	2.077737
50	0.507807	0.503173	0.507921	0.493816	4	2.012717
51	0.482995	0.48336	0.486268	0.43618	4	1.888803
52	0.572389	0.562514	0.564339	0.486906	4	2.186147
53	0.532379	0.526015	0.526157	0.473567	4	2.058117
54	0.47007	0.470325	0.474057	0.452498	4	1.86695
55	0.473866	0.468733	0.470974	0.417544	4	1.831117
56	0.491667	0.472282	0.475101	0.429525	4	1.868575
57	0.451266	0.441189	0.443393	0.392756	4	1.728605
58	0.416594	0.400507	0.40365	0.371672	4	1.592423
59	0.504661	0.490209	0.493794	0.498062	4	1.986726
60						
<i>Average</i>	0.494977	0.487218	0.4899	0.472937		
<i>Count</i>	57	57	57	57		
<i>Y_i</i>	28.21369	27.77144	27.92431	26.95739	<i>Y_{..}</i>	110.8668
$\sum Y_{ij}^2$	54.65999					
$\sum Y_{i.}^2$	3073.733					
$\sum Y_{.j}^2$	218.4314					

Table B. 9 SNR of Echo-Planar Images (Unprocessed and Proposed Processing Methods)

<i>Frame</i>	<i>Unprocessed</i>	$\sigma = 5mm$	$\sigma = 10mm$	$\sigma = 15mm$	<i>ZMedian</i>	<i>Count</i>	Y_j
3	0.320478	0.349964	0.307513	0.303278	0.358027	5	1.63926
4	0.448294	0.444498	0.447677	0.444291	0.454240	5	2.239001
5	0.419348	0.419123	0.422376	0.420761	0.427596	5	2.109205
6	0.455324	0.45136	0.453012	0.446011	0.457289	5	2.262996
7	0.504522	0.503613	0.484402	0.473826	0.497305	5	2.463668
8	0.476294	0.484326	0.493621	0.491826	0.503465	5	2.449532
9	0.490019	0.496803	0.506885	0.509	0.504033	5	2.506741
10	0.482039	0.499794	0.520472	0.514144	0.525711	5	2.54216
11	0.549042	0.540112	0.504022	0.483859	0.538060	5	2.615095
12	0.55452	0.558927	0.54618	0.53988	0.555572	5	2.755078
13	0.501126	0.521847	0.56461	0.57189	0.569906	5	2.729378
14	0.565233	0.5685	0.560263	0.56172	0.559135	5	2.814851
15	0.529422	0.537967	0.547016	0.543176	0.552051	5	2.709631
16	0.575581	0.58331	0.585966	0.585885	0.583915	5	2.914656
17	0.56394	0.580713	0.600328	0.599099	0.594378	5	2.938459
18	0.575207	0.585114	0.597963	0.59089	0.593355	5	2.94253
19	0.582443	0.583382	0.582541	0.578672	0.573978	5	2.901017
20	0.563076	0.573728	0.599826	0.601705	0.597498	5	2.935833
21	0.586283	0.588613	0.592784	0.59143	0.588726	5	2.947836
22	0.580029	0.587362	0.597574	0.587487	0.602507	5	2.954958
23	0.594632	0.583683	0.563716	0.597902	0.568989	5	2.908923
24	0.453717	0.470031	0.489074	0.490152	0.513654	5	2.416628
25	0.447262	0.450487	0.4645	0.468459	0.460896	5	2.291604
26	0.466796	0.46154	0.457505	0.453357	0.447826	5	2.287024
27	0.42608	0.423204	0.423051	0.418955	0.429027	5	2.120317
28	0.422272	0.416765	0.415226	0.416864	0.411199	5	2.082326
29	0.400538	0.405779	0.41869	0.417209	0.420699	5	2.062916
30	0.463104	0.457256	0.444381	0.431231	0.456357	5	2.252329
31	0.475536	0.459942	0.437052	0.422425	0.447707	5	2.242662
32	0.447608	0.444469	0.456462	0.455846	0.458615	5	2.263
33	0.468819	0.458926	0.460647	0.456098	0.458559	5	2.303048
34	0.475338	0.467938	0.450558	0.444316	0.453617	5	2.291768
35	0.445165	0.44646	0.46064	0.457313	0.460476	5	2.270054
36	0.456837	0.460365	0.465215	0.459152	0.472483	5	2.314051
37	0.518897	0.508321	0.49407	0.487077	0.494926	5	2.503292
38	0.453544	0.465818	0.484231	0.48605	0.467284	5	2.356926
39	0.44377	0.455247	0.465031	0.466112	0.462286	5	2.292446
40	0.552102	0.534945	0.509842	0.504208	0.496898	5	2.597995

table continued on next page

Frame	Unprocessed	$\sigma = 5mm$	$\sigma = 10mm$	$\sigma = 15mm$	ZMedian	Count	Y_j
41	0.4735	0.488131	0.513911	0.510739	0.533674	5	2.519956
42	0.587616	0.57345	0.540499	0.525373	0.522533	5	2.749471
43	0.438782	0.453387	0.467532	0.466234	0.496345	5	2.32228
44	0.479089	0.473273	0.481008	0.49608	0.465583	5	2.395033
45	0.437582	0.451074	0.489966	0.5009	0.484559	5	2.36408
46	0.558469	0.54275	0.503479	0.498791	0.493740	5	2.597229
47	0.499244	0.515573	0.537497	0.525117	0.548915	5	2.626346
48	0.558068	0.550117	0.528423	0.511823	0.529824	5	2.678254
49	0.543405	0.544192	0.536339	0.529253	0.551676	5	2.704865
50	0.507807	0.513182	0.527656	0.52686	0.534253	5	2.609757
51	0.482995	0.491375	0.491649	0.486183	0.506670	5	2.458872
52	0.572389	0.570397	0.548035	0.532023	0.550571	5	2.773414
53	0.532379	0.537634	0.533513	0.523836	0.538913	5	2.666274
54	0.47007	0.478896	0.484446	0.481379	0.493057	5	2.407847
55	0.473866	0.477732	0.47567	0.472758	0.483636	5	2.383662
56	0.491667	0.486133	0.479444	0.469852	0.479545	5	2.406641
57	0.451266	0.454305	0.449188	0.440044	0.456387	5	2.251189
58	0.416594	0.399152	0.3864	0.386585	0.396308	5	1.985039
59	0.504661	0.518911	0.494412	0.531185	0.512587	5	2.561755
60							
Average	0.494977	0.497367	0.497263	0.494501	0.501702		
Count	57	57	57	57	57		
Y_i	28.21369	28.34989	28.34399	28.18657	28.59702	$Y_{..}$	141.6912
$\sum Y_{ij}^2$	71.39355						
$\sum Y_i^2$	4015.383						
$\sum Y_j^2$	356.7215						

Table B. 10 SNR of Defect Images from Unprocessed and Processed Echo-Planar Images

<i>Frame</i>	<i>Unprocessed</i>	$\sigma = 5mm$	<i>ZMedian</i>	<i>XYMedian</i>	<i>Count</i>	Y_j
5	1.597537	1.467438	1.572687	1.610128	4	6.24779
6	1.545978	1.510713	1.783584	1.756776	4	6.597051
7	1.629551	1.655056	1.617876	1.712049	4	6.614532
8	1.537931	1.681798	1.680509	1.301559	4	6.201796
9	1.684173	1.68319	1.64652	1.705329	4	6.719212
10	1.62379	1.692639	1.614829	1.565652	4	6.496909
11	1.635993	1.670854	1.324132	1.651075	4	6.282055
12	1.64618	1.646503	1.665279	1.727039	4	6.685
13	1.694771	1.457858	1.460224	1.733064	4	6.345918
14	1.637231	1.632839	1.574223	1.777036	4	6.62133
15	1.58292	1.516732	1.477765	1.519136	4	6.096552
16	1.670746	1.596973	1.597264	1.63008	4	6.495062
17	1.713284	1.64722	1.455192	1.636423	4	6.452119
18	1.591326	1.643767	1.700839	1.355653	4	6.291585
19	1.693987	1.581159	1.558589	1.2135	4	6.047235
20	1.632218	1.57219	1.531743	0.887735	4	5.623885
21	1.623736	1.47679	1.584214	0.799313	4	5.484053
22	1.628507	1.462649	1.571167	0.997762	4	5.660084
23	1.637666	1.632393	1.5843	1.067982	4	5.922342
24	1.556731	1.544024	1.540035	1.376533	4	6.017323
25	1.56829	1.486012	1.454588	1.321145	4	5.830035
26	1.326485	1.623391	1.467405	1.585236	4	6.002518
27	1.480546	1.491798	1.378633	1.512504	4	5.863481
28	1.397678	1.468319	1.390663	1.484867	4	5.741526
29	1.546086	1.475725	1.314745	1.517889	4	5.854446
30	1.529354	1.484033	1.36877	1.495777	4	5.877934
31	1.545403	1.314561	1.431231	1.589199	4	5.880393
32	1.365325	1.306444	1.46698	1.507757	4	5.646506
33	1.360081	1.288262	1.275584	1.485909	4	5.409837
34	1.605542	1.571156	1.491912	1.537175	4	6.205786
35	1.540627	1.505974	1.57136	1.545564	4	6.163524
36	1.635369	1.626113	1.627297	1.648544	4	6.537323
37	1.601539	1.590038	1.596025	1.641629	4	6.429231
38	1.684587	1.697176	1.659151	1.665372	4	6.706285
39	1.66211	1.59834	1.449803	1.681139	4	6.391393
40	1.66397	1.628725	1.561865	1.623871	4	6.478431
41	1.692259	1.698698	1.627396	1.612072	4	6.630426
42	1.721621	1.684652	1.579893	1.632036	4	6.618201

table continued on next page

<i>Frame</i>	<i>Unprocessed</i>	$\sigma = 5mm$	<i>ZMedian</i>	<i>XYMedian</i>	<i>Count</i>	Y_j
43	1.711275	1.631721	1.522265	1.677063	4	6.542323
44	1.658368	1.634557	1.614367	1.509462	4	6.416754
45	1.603529	1.586091	1.646947	1.427036	4	6.263602
46	1.72227	1.699994	1.504094	1.643407	4	6.569765
47	1.612905	1.63375	1.692506	1.60844	4	6.547601
48	1.592909	1.689073	1.654631	1.729527	4	6.66614
49	1.523063	1.580031	1.565798	1.761755	4	6.430646
50	1.536246	1.588004	1.359394	1.585719	4	6.069363
51	1.496268	1.565695	1.387556	1.606982	4	6.056501
52	1.708768	1.606553	1.497343	1.709508	4	6.522171
53	1.607513	1.580342	1.612011	1.598948	4	6.398814
54	1.594618	1.609695	1.621052	1.622267	4	6.447632
55	1.637081	1.646495	1.560182	1.615285	4	6.459044
56	1.524087	1.585672	1.601476	1.67083	4	6.382065
57	1.573422	1.607394	1.585175	1.653975	4	6.419965
58	1.520061	1.510316	1.515728	1.547006	4	6.093112
59	1.596217	1.613342	1.613933	1.638634	4	6.462126
<i>Average</i>	1.594722	1.576017	1.541977	1.540315		
<i>Count</i>	55	55	55	55		
Y_i	87.70973	86.68092	84.80873	84.71735	$Y_{..}$	343.9167
$\sum Y_{ij}^2$	541.6681					
$\sum Y_i^2$	29576.13					
$\sum Y_j^2$	2156.758					

Table B. 11 Count of Undetected Defect Pixels of Defect Images from Unprocessed and Processed Echo-Planar Images

<i>Frame</i>	<i>Unprocessed</i>	$\sigma = 5mm$	<i>ZMedian</i>	<i>XYMedian</i>	<i>Count</i>	<i>Y_j</i>
5	1087	1174	964	773	4	3998
6	836	809	594	712	4	2951
7	584	899	665	591	4	2739
8	846	977	791	723	4	3337
9	747	978	851	529	4	3105
10	967	967	703	1225	4	3862
11	798	728	673	908	4	3107
12	676	598	528	641	4	2443
13	543	510	427	568	4	2048
14	626	566	535	518	4	2245
15	790	685	660	697	4	2832
16	724	646	574	583	4	2527
17	784	693	646	606	4	2729
18	464	405	349	461	4	1679
19	716	587	663	388	4	2354
20	993	907	882	418	4	3200
21	794	668	511	294	4	2267
22	711	744	539	485	4	2479
23	691	898	823	402	4	2814
24	1031	1149	717	623	4	3520
25	1242	1431	893	1150	4	4716
26	647	945	789	929	4	3310
27	901	858	1154	996	4	3909
28	874	805	854	934	4	3467
29	1027	875	790	1046	4	3738
30	1246	975	921	966	4	4108
31	1074	932	1246	1089	4	4341
32	1167	1028	1470	1225	4	4890
33	1215	1002	1144	1398	4	4759
34	879	858	849	1047	4	3633
35	1135	996	1164	1058	4	4353
36	919	843	856	850	4	3468
37	795	821	788	866	4	3270
38	833	816	778	727	4	3154
39	1006	945	773	902	4	3626
40	988	1052	1024	1042	4	4106
41	873	876	862	867	4	3478
42	739	693	889	675	4	2996

table continued on next page

<i>Frame</i>	<i>Unprocessed</i>	$\sigma = 5mm$	<i>ZMedian</i>	<i>XYMedian</i>	<i>Count</i>	<i>Y_j</i>
43	764	722	710	795	4	2991
44	770	762	784	774	4	3090
45	974	967	1071	1053	4	4065
46	853	885	732	655	4	3125
47	701	710	600	642	4	2653
48	566	530	480	601	4	2177
49	570	569	532	684	4	2355
50	685	710	771	856	4	3022
51	793	801	626	810	4	3030
52	788	829	783	853	4	3253
53	833	1013	932	1188	4	3966
54	977	971	748	1085	4	3781
55	843	963	1004	1065	4	3875
56	732	760	876	826	4	3194
57	810	873	914	982	4	3579
58	1316	1306	1268	1323	4	5213
59	1144	1358	1284	1223	4	5009
<i>Average</i>	856.127273	855.7818	808.2545	824.1273		
<i>Count</i>	55	55	55	55		
<i>Y_i</i>	47087	47068	44454	45327	<i>Y_{..}</i>	183936
$\sum Y_{ij}^2$	164613560					
$\sum Y_{i.}^2$	8463277238					
$\sum Y_{.j}^2$	648754370					

Table B. 12 Count of Undetected Defect Regions of Defect Images from Unprocessed and Processed Echo-Planar Images

<i>Frame</i>	<i>Unprocessed</i>	$\sigma = 5mm$	<i>ZMedian</i>	<i>XYMedian</i>	<i>Count</i>	<i>Y_j</i>
5	0	2	1	0	4	3
6	1	0	1	0	4	2
7	1	1	1	0	4	3
8	0	0	1	0	4	1
9	2	1	2	1	4	6
10	2	1	1	1	4	5
11	1	1	1	1	4	4
12	2	1	1	1	4	5
13	0	0	0	0	4	0
14	1	1	1	0	4	3
15	2	1	0	1	4	4
16	1	0	1	2	4	4
17	1	1	1	1	4	4
18	1	1	0	0	4	2
19	2	1	2	0	4	5
20	3	1	1	0	4	5
21	2	1	0	0	4	3
22	1	1	0	0	4	2
23	1	3	1	0	4	5
24	2	3	0	0	4	5
25	2	2	1	2	4	7
26	1	1	1	1	4	4
27	2	1	3	1	4	7
28	2	1	0	1	4	4
29	2	1	0	1	4	4
30	2	1	2	1	4	6
31	0	1	3	2	4	6
32	1	0	2	1	4	4
33	1	0	2	2	4	5
34	1	1	2	4	4	8
35	2	2	2	2	4	8
36	1	2	2	2	4	7
37	1	2	1	2	4	6
38	1	2	1	1	4	5
39	2	3	3	2	4	10
40	1	1	2	1	4	5
41	3	4	1	4	4	12
42	2	2	2	2	4	8

table continued on next page

<i>Frame</i>	<i>Unprocessed</i>	$\sigma = 5mm$	<i>ZMedian</i>	<i>XYMedian</i>	<i>Count</i>	<i>Y_j</i>
43	2	2	1	1	4	6
44	2	2	2	2	4	8
45	1	2	2	1	4	6
46	3	4	2	2	4	11
47	2	2	0	1	4	5
48	2	2	1	1	4	6
49	2	1	1	1	4	5
50	1	1	2	2	4	6
51	1	0	1	2	4	4
52	1	1	0	1	4	3
53	1	1	2	3	4	7
54	2	2	0	3	4	7
55	1	2	2	2	4	7
56	2	2	1	2	4	7
57	0	0	1	0	4	1
58	1	1	1	1	4	4
59	1	3	3	3	4	10
<i>Average</i>	1.418182	1.381818	1.236364	1.236364		
<i>Count</i>	55	55	55	55		
<i>Y_i</i>	78	76	68	68	<i>Y_{..}</i>	290
$\sum Y_{ij}^2$	558					
$\sum Y_{i.}^2$	21108					
$\sum Y_{.j}^2$	1846					

Table B. 13 Count of False Alarm Defect Pixels of Defect Images from Unprocessed and Processed Echo-Planar Images

<i>Frame</i>	<i>Unprocessed</i>	$\sigma = 5mm$	<i>ZMedian</i>	<i>XYMedian</i>	<i>Count</i>	<i>Y_j</i>
5	628	1140	872	893	4	3533
6	1091	1281	521	474	4	3367
7	994	589	956	714	4	3253
8	1030	370	560	2510	4	4470
9	578	350	594	733	4	2255
10	520	302	815	475	4	2112
11	667	624	2331	507	4	4129
12	672	749	762	478	4	2661
13	630	1514	1586	506	4	4236
14	751	825	1057	480	4	3113
15	756	1126	1321	1104	4	4307
16	529	839	910	793	4	3071
17	342	618	1394	738	4	3092
18	1057	943	833	2156	4	4989
19	484	969	976	3240	4	5669
20	390	681	861	7261	4	9193
21	644	1349	1064	9304	4	12361
22	706	1332	1078	5570	4	8686
23	701	511	751	4766	4	6729
24	653	585	1033	1927	4	4198
25	412	568	1256	1772	4	4008
26	2287	536	1332	688	4	4843
27	1171	1161	1466	929	4	4727
28	1644	1335	1705	1126	4	5810
29	747	1211	2232	847	4	5037
30	645	1124	1816	1077	4	4662
31	753	2186	1037	566	4	4542
32	1490	2037	496	681	4	4704
33	1556	2272	2223	675	4	6726
34	683	834	1145	780	4	3442
35	599	885	473	656	4	2613
36	520	627	610	546	4	2303
37	770	786	797	561	4	2914
38	427	408	558	590	4	1983
39	347	622	1433	393	4	2795
40	325	372	637	398	4	1732
41	350	329	558	604	4	1841
42	409	557	702	736	4	2404

table continued on next page

<i>Frame</i>	<i>Unprocessed</i>	$\sigma = 5mm$	<i>ZMedian</i>	<i>XYMedian</i>	<i>Count</i>	<i>Y_j</i>
43	395	670	1081	459	4	2605
44	543	625	669	1076	4	2913
45	466	532	232	1109	4	2339
46	287	345	1152	712	4	2496
47	750	673	608	824	4	2855
48	797	718	871	536	4	2922
49	1244	1022	1112	363	4	3741
50	1024	807	1797	669	4	4297
51	1109	820	1817	664	4	4410
52	384	654	1124	317	4	2479
53	623	537	509	297	4	1966
54	547	501	686	345	4	2079
55	556	406	666	406	4	2034
56	1106	835	662	485	4	3088
57	838	651	690	387	4	2566
58	579	632	646	458	4	2315
59	382	109	181	161	4	833
<i>Average</i>	737.96364	819.7091	1004.618	1227.673		
<i>Count</i>	55	55	55	55		
<i>Y_i</i>	40588	45084	55254	67522	<i>Y_{..}</i>	208448
$\sum Y_{ij}^2$	396196738					
$\sum Y_i^2$	1.129E+10					
$\sum Y_j^2$	1.009E+09					

Table B. 14 Count of False Alarm Defect Regions of Defect Images from Unprocessed and Processed Echo-Planar Images

<i>Frame</i>	<i>Unprocessed</i>	$\sigma = 5mm$	<i>ZMedian</i>	<i>XYMedian</i>	<i>Count</i>	<i>Y_j</i>
5	3	2	0	1	4	6
6	2	3	0	1	4	6
7	5	1	2	1	4	9
8	6	0	2	4	4	12
9	4	0	0	0	4	4
10	3	1	1	0	4	5
11	3	4	2	0	4	9
12	1	4	0	0	4	5
13	0	5	2	0	4	7
14	3	3	3	0	4	9
15	4	5	2	4	4	15
16	3	2	1	3	4	9
17	1	2	2	1	4	6
18	2	2	0	5	4	9
19	1	2	1	6	4	10
20	0	2	0	8	4	10
21	5	8	0	8	4	21
22	3	6	2	8	4	19
23	3	1	0	7	4	11
24	3	1	1	2	4	7
25	0	2	1	4	4	7
26	4	2	2	2	4	10
27	1	2	3	0	4	6
28	10	8	4	1	4	23
29	0	8	5	0	4	13
30	1	6	6	2	4	15
31	9	11	0	0	4	20
32	10	10	0	0	4	20
33	10	10	6	2	4	28
34	2	3	3	1	4	9
35	4	4	2	1	4	11
36	0	1	1	0	4	2
37	6	4	1	0	4	11
38	1	0	0	0	4	1
39	0	0	2	0	4	2
40	0	0	1	0	4	1
41	1	0	1	1	4	3
42	0	0	3	4	4	7

table continued on next page

<i>Frame</i>	<i>Unprocessed</i>	$\sigma = 5mm$	<i>ZMedian</i>	<i>XYMedian</i>	<i>Count</i>	<i>Y_j</i>
43	1	2	3	2	4	8
44	1	1	2	3	4	7
45	0	1	2	2	4	5
46	1	1	2	1	4	5
47	1	1	0	1	4	3
48	1	2	2	0	4	5
49	3	4	1	1	4	9
50	3	1	2	1	4	7
51	8	2	2	2	4	14
52	0	1	2	0	4	3
53	1	1	0	0	4	2
54	1	0	0	0	4	1
55	0	0	0	0	4	0
56	4	5	2	0	4	11
57	4	1	0	2	4	7
58	4	0	0	0	4	4
59	5	0	1	0	4	6
<i>Average</i>	2.745455	2.690909	1.545455	1.672727		
<i>Count</i>	55	55	55	55		
<i>Y_i</i>	151	148	85	92	<i>Y_{..}</i>	476
$\sum Y_{ij}^2$	2324					
$\sum Y_i^2$	60394					
$\sum Y_j^2$	6042					

VITA

Eyler Robert Coates, Jr. was born in Baton Rouge, Louisiana, on January 3, 1956. He graduated from Istrouma High School in May 1974. After that he attended Louisiana State University in Baton Rouge and graduated with a bachelor's degree in Industrial Engineering in May 1979. He married Denise Slocum in 1975 and they had two children, George (born in 1975) and John (born in 1981). After graduation in 1979, he went to work as an industrial engineer at Cincinnati Milacron, a machine tool manufacturer, in the Industrial Robot Division in Cincinnati, Ohio. There in a pilot manufacturing plant, he planned the assembly methods and layout for a new production facility in Greenwood, South Carolina. In 1981, he transferred down to Greenwood to become the industrial engineer for the assembly portion of the manufacturing plant. After 5 years with Cincinnati Milacron, in 1984, he took a new job as industrial engineer at Hatteras Yachts in High Point, North Carolina. There he spent most of his time with time studies, plant layout changes and manufacturing budgeting. After 5 years at Hatteras Yachts, in 1989, he went to start an industrial engineering department at Davis Yachts on Roanoke Island off North Carolina. He spent most of his time, heading up the production control, bill of materials, and industrial engineering functions for the company. Two years later, in 1991, after a national luxury tax was passed, many small yacht builders including Davis Yachts became insolvent.

After applying for and receiving a Louisiana Board of Regents Dean's Fellowship at L.S.U. in Baton Rouge, he began his graduate studies in August 1991. There he married Brenda Ann Murphy. Having completed all coursework for the doctorate and obtaining a

master of science degree in Engineering Science in May of 1996, he accepted a position as Assistant Professor of Engineering Technology at the University of Southern Mississippi that began in August 1996. There he teaches junior, senior and master's level courses in industrial engineering technology such as Maintenance Engineering, Industrial Cost Control, Resources in Eng. Tech. (applied operations research) graduate course, Motion and Time Study, Industrial Simulation, and Plant Layout. He was inducted into Phi Kappa Phi, L.S.U. Chapter, on April 17, 1997. After one year at U.S.M., he was promoted to the coordinator of the Industrial Engineering Technology program and was elected to the Faculty Senate at U.S.M. by his peers to represent the School of Engineering Technology.

At U.S.M., he spends most of his time teaching (9 credit hours per semester) and the remaining time is spent in applied research. He has up to now published 12 articles in the following peer-reviewed journals: *International Journal of Production Economics*, *Computers and Industrial Engineering*, *Integrated Manufacturing Systems*, *Proceedings of the 1997 ASEE Southeastern Section Meeting*, *Proceedings of the 1998 ASEE Southeastern Section Meeting*, *Proceedings of 15th Annual Conference on Computers & Industrial Engineering*.

Eyler Coates is currently a candidate for the degree of Doctor of Philosophy in The Interdepartmental Programs in Engineering at L.S.U.

Eyler Coates also has some experience as an industrial engineering consultant specializing in computer simulation of manufacturing facilities and manufacturing productivity improvement. His clients include Fruit of the Loom and Masonite Corporation, a Division of International Paper. He is also a member of the Institute of

Industrial Engineers (IIE), the Institute for Operations Research and Management Sciences (INFORMS), and the American Society for Engineering Education (ASEE).

He can be contacted at Eyler Coates, Assistant Professor, The University of Southern Mississippi, School of Engineering Technology, Box 5137, Hattiesburg, Mississippi 39406-5137, E-mail : Eyler.Coates@usm.edu, Work phone (601) 266-6421, FAX: (601) 266-5717.


DOCTORAL EXAMINATION AND DISSERTATION REPORT

Candidate: Eyler Robert Coates, Jr.

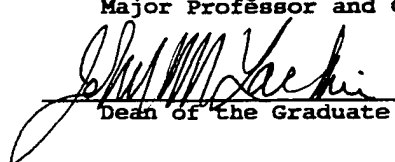
Major Field: Engineering Science

Title of Dissertation: Internal Defect Detection in Hardwood Logs
with Fast Magnetic Resonance Imaging

Approved:

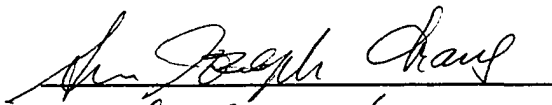


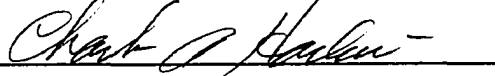
Major Professor and Chairman

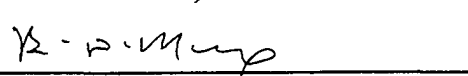



Dean of the Graduate School

EXAMINING COMMITTEE:





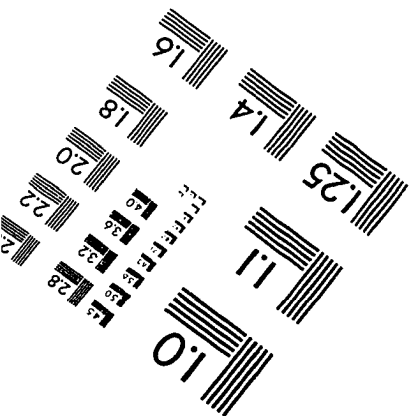
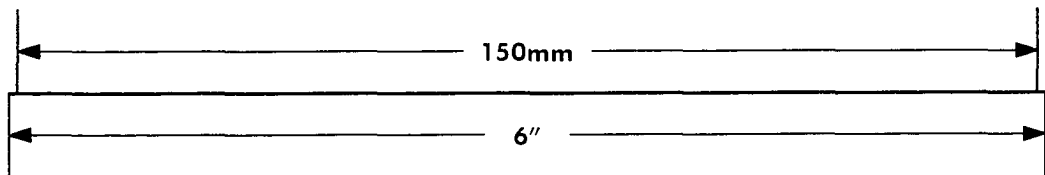
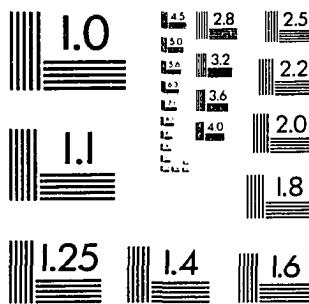
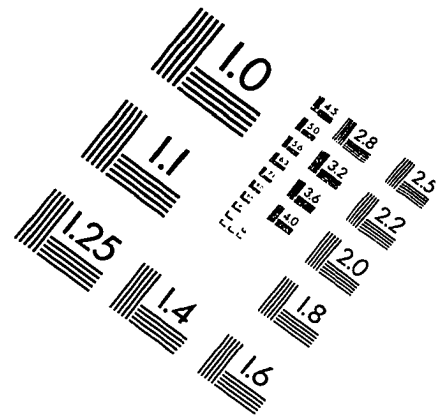
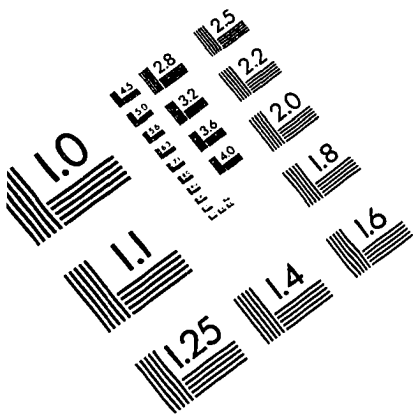




Date of Examination:

7 October 1998

IMAGE EVALUATION TEST TARGET (QA-3)



APPLIED IMAGE, Inc
1653 East Main Street
Rochester, NY 14609 USA
Phone: 716/482-0300
Fax: 716/288-5989

© 1993, Applied Image, Inc., All Rights Reserved

