A Computer Vision Based Tree Ring Analysis and Dating System

by

W. Steven Conner

A Thesis Submitted to the Faculty of the

DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING

In Partial Fulfillment of the Requirements
For the Degree of

Masters of Science

In the Graduate College

The University of Arizona

1 9 9 9

Statement by Author

This thesis has been submitted in partial fulfillment of requirements for an advanced degree at The University of Arizona and is deposited in the University Library to be made available to borrowers under rules of the Library.

Brief quotations from this thesis are allowable without special permission, provided that accurate acknowledgment of source is made. Requests for permission for extended quotation from or reproduction of this manuscript in whole or in part may be granted by the head of the major department or the Dean of the Graduate College when in his or her judgment the proposed use of the material is in the interests of scholarship. In all other instances, however, permission must be obtained from the author.

Signed: _____

Approval by Thesis Director

This thesis has been approved on the date shown below:

_____ _____

Robert A. Schowengerdt
Professor of ELECTRICAL &
COMPUTER ENGINEERING

Date

Acknowledgments

I would like to thank Martin Munro for his technical support and assistance throughout the project. I thank Paul Sheppard for providing algorithms and assistance for computerized skeleton plot generation, and Matt Kopala for the initial port of the skeleton plotting algorithm to Tcl/Tk. My gratitude goes out to Jim Burns for testing the software and providing advice and recommendations on how to make the system more useful and intuitive for dendrochronologists. I wish to thank Dr. Robert Schowengerdt and Dr. Malcolm Hughes for devising the concept for TREES and providing invaluable advice and ideas. Finally, a special thanks to all of my friends and family for their support and encouragement.

This work was supported by the National Science Foundation, *Grant SBR9601867*, and the University of Arizona, Office of the Vice President for Research.

Dedication

I dedicate this thesis to my lovely wife Mary, whose love, support, and friendship have made it all worthwhile.

Table of Contents

TABLE OF CONTENTS—*Continued*

List of Tables

List of Figures

Abstract

This thesis describes the design and implementation of a computer vision based analysis system for dendrochronology. The primary issues covered are software engineering design, automatic focus detection, illumination distortion, spatial and radiometric registration of a sequence of images into a single mosaic, edge detection and linking, and automated measurement. A modification of the Canny edge detection algorithm that adapts to the variable characteristics of tree ring images is also described. These issues are not unique to the application, but are likely of interest to anyone developing automated image analysis systems.

A Computer Vision Based Tree Ring Analysis and Dating System

W. Steven Conner, M.S.
The University of Arizona, 1999

Director: Robert A. Schowengerdt

This thesis describes the design and implementation of a computer vision based analysis system for dendrochronology. The primary issues covered are software engineering design, automatic focus detection, illumination distortion, spatial and radiometric registration of a sequence of images into a single mosaic, edge detection and linking, and automated measurement. A modification of the Canny edge detection algorithm that adapts to the variable characteristics of tree ring images is also described. These issues are not unique to the application, but are likely of interest to anyone developing automated image analysis systems.

Chapter 1
INTRODUCTION

1.1 Problem Statement

This thesis describes the software engineering and image processing issues involved in the design of a semi-automated, computerized tree ring analysis and dating system [4]. It is based on cooperative work between the Electrical and Computer Engineering Department and the Laboratory of Tree-Ring Research at the University of Arizona.

1.2 Overview of Thesis

The methods described in this thesis provide a means for increasing the efficiency of tree ring analysis. Rather than attempting to produce a fully automated solution, the philosophy of this project is to create a computer-assisted environment that integrates analyst intervention with algorithmic decisions. This approach allows the development of a robust system that can emulate the complex human vision analysis of tree samples, which often tend to have unpredictable features.

The thesis begins by presenting an overview of dendrochronology and a description of the requirements of the system. Several examples of previous systems that were developed for the purpose of computerized image analysis of tree rings will be discussed and contrasted to the work described in this thesis. Software engineering approaches used in the development of the system are explained. Finally, a description of the requirements for data acquisition is provided, followed by a description of the computer vision techniques used to identify and measure tree rings.

1.3 Overview of Dendrochronology

The science of tree ring dating, known as *dendrochronology*, provides techniques for the precise dating of trees. During each year in the lifetime of many trees, a single tree ring is created. The outer-most ring before the tree's bark corresponds to the current year of a live tree or the last year of growth of a dead tree, and by counting the rings inward toward the center of the tree it is possible to tell how many years the tree lived. Moreover, the widths of tree rings vary from year to year, creating patterns of variation that are present across different trees in a geographical region. These patterns make it possible to cross-date between tree samples, as described by Douglass [6] and Stokes and Smiley [27].

1.3.1 The Douglass Method of Cross-Dating

The techniques practiced at the Laboratory of Tree-Ring Research used to cross-date tree samples, collectively known as the Douglass method, were pioneered by Andrew Douglass at the University of Arizona early in the twentieth century [6]. The Douglass method of tree ring dating involves the use of *skeleton plots*, which are a normalized representation of tree ring widths that allow straightforward comparisons to be made between tree ring width patterns of multiple tree samples. Skeleton plots are generally created on graph paper, where each vertical line on the graph paper represents a single year in the lifetime of the tree sample. If a particular ring is much smaller than neighboring rings, a line is drawn at the corresponding location on the skeleton plot. The narrower a ring is (relative to its neighbors), the longer the line will be. In addition to ring widths, other unique tree ring features such as the existence of a "frost ring," which may be caused by a very hard freeze, can be included symbolically on a skeleton plot. An example skeleton plot is shown in Figure 1.1.

Skeleton plotting allows the graphical representation of the patterns of relative tree ring width variation. Relative ring widths have a stronger correlation between

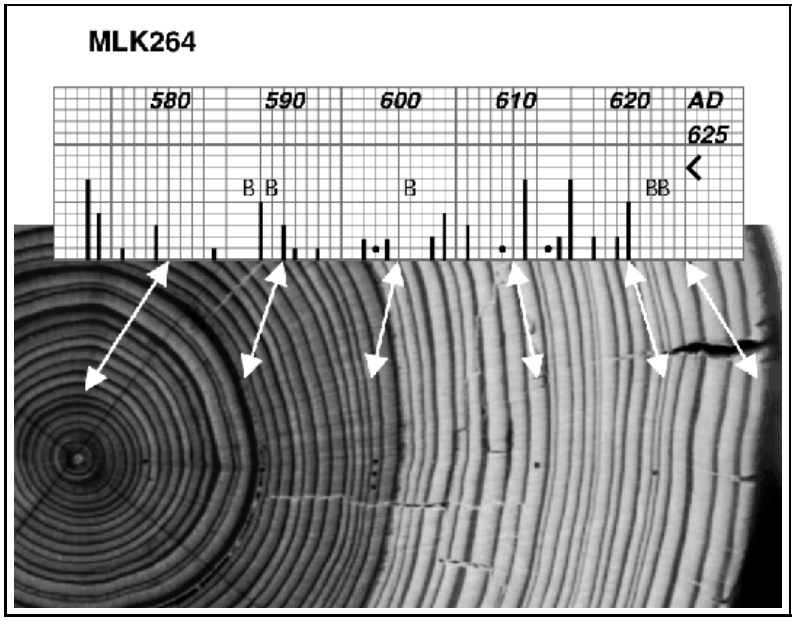


FIGURE 1.1. Example skeleton plot[1].

samples than absolute tree ring widths. Dendrochronologists who apply the Douglass method of cross-dating manually create and compare skeleton plots from multiple tree samples. Samples are dated by visually matching the pattern from one skeleton plot to another. One of the requirements of the Douglass method of tree ring dating that sets it apart from other dating techniques is the fact that the original wood sample must remain available throughout the dating process for reference. After measuring a tree sample and producing a skeleton plot, it is not a good idea to consider the skeleton plot to be a 100% accurate representation of the tree's chronology. This is because physical anomalies may occur at different times during the lifetime of a tree. For example, *micro* rings which are so small that they are not identified during the initial tree ring width measurement procedure may be present in a sample. In this case, the years corresponding to micro rings will be missing from the skeleton plot. Likewise, unusual climate conditions in a particular year might cause the existence of a *false* ring in the middle of a single annual growth ring. If this false ring is mistaken

[1]Courtesy the Laboratory of Tree-Ring Research, the University of Arizona.

for an actual tree ring during width measurement, an extra non-existent year will be included in the skeleton plot. Because a single extra or missing ring will cause a skeleton plot to be undateable, it is imperative to keep the original wood sample available to reconcile any problems that are identified during cross-dating.

Researchers invest many hours manually examining tree ring samples under a microscope. By tediously measuring tree ring features such as ring boundaries and widths and correlating data from different samples, they are able to perform dating so precise that Carbon-14 dating is calibrated from their results. Despite advancements in computational speed and image processing algorithms, manual measurement techniques remain the primary approach for collecting tree ring data since no previously developed automated technique is reliable enough for widespread use.

1.3.2 TREES System Overview

The computerized tree-ring dating system described in this thesis, known as *TREES*, includes a CCD camera attached to a microscope that is used to acquire images of a tree ring sample. The sample to be imaged is placed on an x-y positioning table under the microscope. The camera, microscope focus, and positioning table position are all controlled by a workstation. A diagram of the TREES system is shown in Figure 1.2.

TREES is designed to resolve small tree rings while limiting the amount of memory required to image a tree sample. Tree rings can range from 0.05 mm in width to several centimeters, although most are around 1 mm or less. Typical tree sample sizes range from 500 mm to over 100 cm in length. The hardware chosen for acquiring tree ring image data consists of a 1317x1035 Kodak Megaplus 1.4i CCD digital camera attached to a Nikon $SMZ-U$ microscope for imaging a cross section or core sample of a tree situated on a positioning table under the microscope. The microscope is generally set to a magnification of 1.1X. In order to increase the field of view of

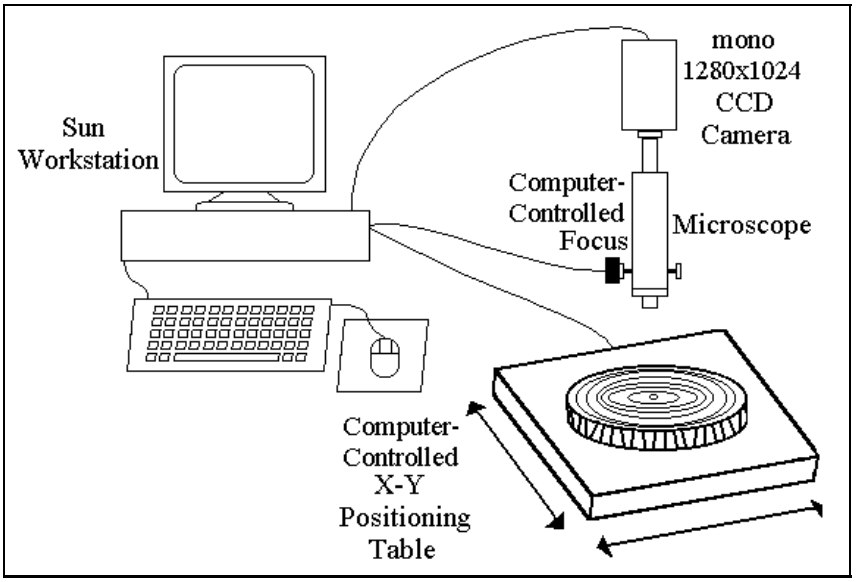


FIGURE 1.2. Diagram of TREES system.

captured frames, a video coupler lens with a magnification of 0.63X is used, creating a combined magnification of 0.693X and a diagonal field of view of approximately 16.4 mm for each frame. This results in a pixel size on the sample of approximately 10 μm, which is sufficient to resolve most small tree rings. For illumination, a fiber-optic ring illuminator with a diameter of approximately 75 mm is mounted on the objective lens of the microscope. The positioning table includes two stepper motors under computer control for motion in the X- and Y-directions of up to 75 cm, with a precision of about 1mm. The table is belt driven (rather than lead screw driven) for faster positioning over many centimeters. Software correlation techniques are applied to make up for the low precision of the positioning table when combining multiple captured frames into a single mosaiced image. Finally, a stepper motor is also used to control the Z-axis for focus adjustments. The camera, microscope, focus, and positioning table are all controlled by a Sun ULTRA 1 workstation with 256 MB of memory, 8.2 GB disk capacity, and a 4 MB frame buffer. With the current configuration, a monochrome mosaiced image of a 500 mm by 5 mm tree core sample created from 50 overlapping frames occupies approximately 54 MB. The time required

to acquire the frames and automatically create the mosaic is under ten minutes.

For the first step of the tree ring analysis process, a series of overlapping images are acquired across the tree sample, from the center of the tree (the pith) to the outer edge (the bark). These images must be corrected for any illumination nonuniformity and normalized for any variation in camera gain and brightness offset during capture. The individual frames are correlation-registered into a single mosaic. It is then possible to apply edge detection techniques to find the locations of tree ring boundaries in the sample mosaic, and then tree ring measurements can be made. Ring widths and other characteristics may be used to match ring patterns from one tree to another, allowing cross-dating to be performed. A flow chart of the data processing steps used in TREES is shown in Figure 1.3.

Several broad philosophies guide the development of this system. First, the goal is to create a *computer-assisted*, not automated, tree ring dating system. When dealing with natural phenomena such as tree rings, it is virtually impossible to predict all special cases. Because *every* tree ring must be accounted for to accurately cross-date a tree sample, it is better to allow analyst intervention when difficulties arise rather than attempting to develop a complex algorithm to make decisions with a low probability of error under all conditions. Also, to improve accuracy and reliability, the system is *wood-centered*, rather than image-centered. The analyst can visually explore the wood surface through the microscope at any point in the tree ring analysis process to resolve problems of ring identification and dating. The system is primarily meant to increase the efficiency of tree ring analysis by detecting and measuring the majority of the tree rings in a sample with minimum operator intervention. However, the operator may use expertise in dendrochronology for verification, to provide suggestions to the system, or to detect subtle or "false" rings.

At the time of writing, the computer vision problems described in this paper are only partially solved. The algorithms and techniques included here represent the best methods developed to date to solve the challenges of tree ring analysis.

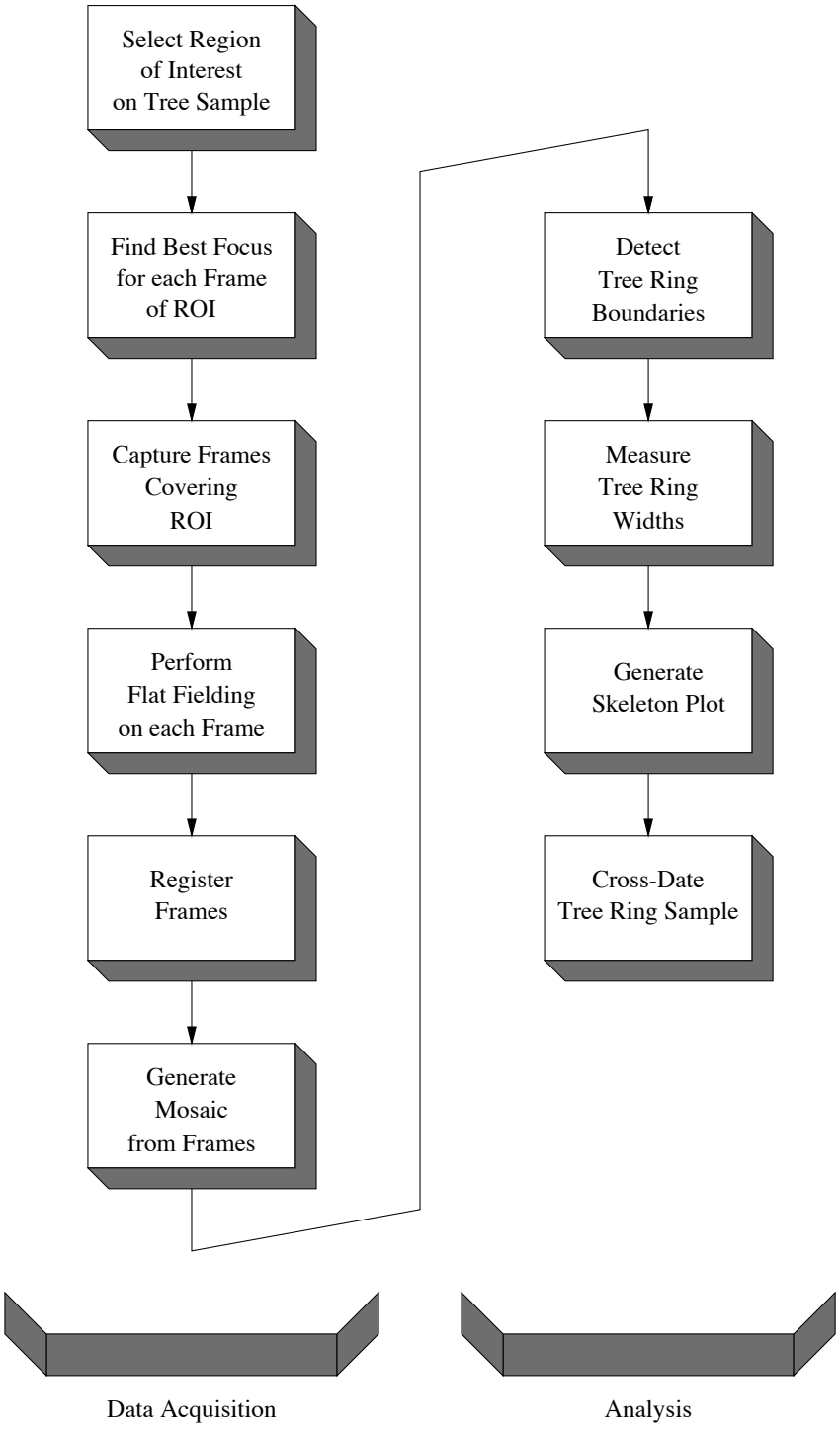


FIGURE 1.3. TREES data processing flow chart.

1.4 Review of Earlier Systems

TREES is not the first system to apply image analysis to the field of dendrochronology. However, to the knowledge of the TREES project design team, it is the first image analysis system to apply the Douglass method of tree ring dating. For completeness, an overview of three earlier image analysis systems for dendrochronology is provided.

1.4.1 DENDRO

The DENDRO image analysis system for dendrochronology has been implemented on the Macintosh platform as MacDENDRO [19, 7] and the Windows platform as WinDENDRO [8]. This system accepts input image data from either an input image file, image data acquired from scanning a tree sample on a flatbed scanner, or from a digital camera. Tree ring detection in DENDRO is accomplished by analyzing one or more radial "paths" from the pith to the bark of the sample. These paths must be defined by the analyst before tree ring boundaries can be identified. A maximum of 32 paths may be defined.

Once paths have been defined, DENDRO automatically detects tree rings via one of two available identification algorithms. The first algorithm is a means of completely automated tree ring detection along each path. It is performed by analyzing the grayscale intensity differences along a given scan line. Any major jumps in intensity are considered to be tree ring boundaries. The second tree ring detection algorithm uses a "teach and show" technique. This method requires the analyst to manually select a valid tree ring boundary in the image. This tree ring boundary is treated as a model in a form of template matching, and any regions along a scan path that have similar features to the model are also considered to be tree rings. Once the tree rings along a path have been identified, their widths are computed along the scan path. If more than one scan path is created, the measured tree ring widths can be averaged.

DENDRO appears to be the primary commercial image analysis system available

to the dendrochronology community. According to the authors of the software, it produces very good results for typical conifer samples. However, there are several limitations in DENDRO. First, DENDRO requires an analyst to define radial scan paths, and therefore, limits the data analyzed from the sample. These scan paths are one-pixel wide lines and are sensitive to any anomalies which may cause local variations in tree ring widths. Also, because there is no easy way to return to the wood in DENDRO, it lacks the requirements of the Douglass method of tree ring dating that are central to the TREES system.

1.4.2 MacDRUID

MacDRUID is an image analysis system for dendrochronology that runs on the Macintosh [28]. It uses the commercial Prism Image Analysis System from Dapple Systems, Inc., plus the addition of custom Fortran code. The purpose of the system is to apply image analysis to tree ring densitometry measurements. The first step of data acquisition for the MacDRUID system is to microtome a 300 μm thick section of a sample core. This is a standard procedure used when preparing samples for X-ray analysis, whether or not it is for the purpose of acquiring image data. Next, an X-ray negative is created from the microtome and the negative is projected and digitized with a CCD frame grabber.

In order to perform image analysis on the resulting X-ray image, a region of interest must be explicitly selected by an analyst. This region must be created with great care to ensure that all tree ring boundaries within the region of interest are exactly vertically aligned. This is necessary because each column of pixels in the region of interest is averaged in order to produce a 1-dimensional plot of the data. Finally, the tree ring boundaries are estimated from the 1-dimensional plot at the locations where the derivative of the data is maximum. From this data, ring widths and density parameters can be estimated. The authors of the software claim that

the results produced by this system are not significantly different from the results obtained by traditional measurement methods.

1.4.3 Reflected-Light Image Analysis System

The Reflected-Light Image Analysis System, designed at the Laboratory of Tree Ring Research at the University of Arizona, measures the brightness of conifer tree rings and then uses brightness as a substitute for density [24, 25, 26]. Ring density is a very useful measurement of tree ring patterns when tree ring widths show little variation. However, because it must be measured through X-ray densitometry, it is very slow and expensive. By using a very careful calibration of the imaging system, the techniques used in the Reflected-Light Image Analysis System are able to produce a very accurate estimation of wood density, without the difficulty of X-ray densitometry.

The image analysis algorithms used in this system require the rings to be dated prior to measuring. Any unwanted noise features such as sap ducts are manually erased from the image before analysis. When an image of a tree ring sample is acquired, care is taken to ensure that tree ring boundaries within the image are vertically aligned. If it is known that N tree rings exist in a given tree ring image, the N largest first difference locations found while horizontally traversing the image are considered to be the positions of the rings. Once the positions of the tree rings are computed, the information is reduced from the 2-dimensional image to a 1-dimensional representation by computing the average of each of the horizontal scans between each pair of rings. From this information, tree ring widths, earlywood and latewood widths, and other intra-ring characteristics useful in dating are computed by measuring the density changes within rings.

The Reflected-Light Image Analysis System accurately estimates the data obtained by the X-ray densitometry method in a fraction of the time. The primary disadvantages of this system relative to TREES are that it is unable to determine the

locations of tree rings without knowing the tree ring count up front, and orientations of tree rings are constrained to be vertical.

Chapter 2
SOFTWARE ENGINEERING

One of the initial requirements of the TREES system was that it must be as portable as possible across different software / hardware environments. The initial version of TREES has been developed under the UNIX environment on a Sun Solaris workstation. In the future, it may become necessary to port TREES to a PC environment, running either Windows or another variant of UNIX such as Linux, in order to reduce the overall cost of the system and to make it as widely available to dendrochronology labs around the world as possible.

Tcl/Tk was chosen as the graphical user interface (GUI) development language because programs written in Tcl/Tk will run under UNIX, Windows, and Macintosh operating systems with minimal changes to the source code. For algorithm development, C was chosen because it is an efficient, widely used, and standardized language for systems development. Another advantage of using C is that a pre-existing library of image processing routines, known as SADIE, could be adapted as the core of the image processing and computer vision portions of the system.

2.1 Algorithm Development

2.1.1 SADIE Image Processing Library

SADIE is a library of C image processing routines developed and maintained at the Digital Image Analysis Laboratory of the University of Arizona. It includes a relatively complete set of standard image processing routines, including contrast enhancement algorithms, gradient filters such as Sobel, Roberts, and Prewitt, Fourier and convolution filters, and classification algorithms. SADIE was chosen as the image

parameter	*variable name*	*data type*
# bands/image	nbnd	short integer
# lines/band	nlin	short integer
# columns/band	npix	short integer
pixel value array	data[nbnd][nlin][npix]	PIXEL

TABLE 2.1. SADIE Image Data Structure. These are the data structure variables that are relevant to the discussion of the integration of SADIE into TREES. In reality, the data structure includes additional variables.

processing core for the initial development of TREES because it contains many of the basic image processing routines necessary in an image analysis system. In addition, since it is actively maintained by researchers at the University of Arizona, it is possible to modify and extend the library without the licensing fees and copyright issues that would be present if a commercial image processing package were used. An implementation of the SADIE library for Macintosh systems is described by Schowengerdt and Mehldau in [13], [21], and [22].

The basic data structure used to store images in SADIE is shown in Table 2.1. Each SADIE image contains size identifiers to store the number of bands, number of lines, and number of pixels (or columns) in the image. The actual image data is stored in a three dimensional array of type PIXEL. In the SADIE library, PIXELs are single precision floating point numbers. In the actual SADIE implementation, more variables are contained in the image data structure than are shown in Table 2.1. However, since these additional variables are used in features of the SADIE library that are not relevant to this project, they are not described here.

2.1.2 Limitations of SADIE

While the SADIE library provides a large set of image processing routines, it has several limitations in the context of an image analysis system such as TREES that requires very large image arrays to be processed. First, as shown in Table 2.1, the

maximum number of lines and/or pixels in a SADIE image is constrained by the maximum value of a short integer. Assuming that a signed 16-bit short integer is used to store these variables, the maximum number of lines or pixels in an image is 32,767. If frames of 1280 pixels per line are captured during the image acquisition stage, a maximum width mosaic can hold only around twenty-five horizontally aligned frames. Based on the typical image capture resolution described in Section 1.3.2, this will allow approximately 32cm of a tree ring sample to be imaged in one mosaic.

The second disadvantage of using the SADIE library for processing very large images is related to the fact that all image data is stored as 32-bit per pixel floating point numbers. The 32-bit precision is useful during many image processing routines, especially when round-off errors can cause problems. However, outside of SADIE routines, many images do not require this much precision. For example, the camera used to capture digital images for TREES returns only 8-bits of data per pixel. Edge maps require only one-bit per pixel in the unlabeled case, and an edge label map only requires enough bits to store the integer range from zero to the number of labeled fragments in the image (typically less than 10,000). For this type of image data, the 32-bit per pixel values are very costly, especially in the case of very large image mosaics.

2.1.3 Modifications to SADIE for TREES

During edge detection and image processing, it is often necessary to work with multiple versions of an image mosaic simultaneously, such as the original image data, gradient information, and edge maps. In order to accommodate varying levels of precision, depending on the type of data in a given image, the single 32-bit image format of the SADIE library has been extended to three image formats for use in TREES. The normal SADIE image format uses 32-bit per pixel floating point values, as described above. This type of image is most useful for storing gradient and convo-

lution output data, as well as intermediate pixel values during image processing. The SADIE_SHORT image type is identical to the normal SADIE image format, except for the fact that image data is stored as 16-bit unsigned short integers. This type of image is ideal for storing edge label maps, where the background pixels have a value of 0 and foreground edge fragments can have labels ranging from 1 to 65,535. Finally, the SADIE_BYTE image format stores pixels as 8-bit unsigned characters, which is useful for storing raw image data from the frame grabber.

In order to maintain compatibility with the existing SADIE library routines, the new SADIE_SHORT and SADIE_BYTE image types require their own versions of any related library routines. This is due to the fact that all of the existing SADIE library routines expect all SADIE images to use 32-bit per pixel data.

In addition to adding the ability to store SADIE images at multiple precisions, a format has been added for the efficient storage of a mosaic of image frames. This MOSAIC_IMAGE format is described in detail in Section 3.6.

2.2 Graphical User Interface Design

In addition to the development of algorithms for performing image processing and analysis, the design of a graphical user interface is an imperative task in the software engineering process. The GUI is the interface to the algorithms and data that makes the system useful to dendrochronologists. Regardless of how powerful a software package may be, if it is not easy, reliable, and intuitive to use it will not be accepted in the field.

A discussion of the software implementation of the graphical user interface for the TREES project follows.

2.2.1 Tcl/Tk

The graphical user interface for TREES has been developed in Tcl/Tk. Tcl is a high level scripting language written by John Ousterhout at UC Berkeley [16]. Tcl is typically packaged with Tk, an extension that provides a toolkit of Tcl commands for building and manipulating user interface components such as buttons, menus, dialogs, and image display windows. Because Tcl itself is written in C, it provides seamless integration with existing C and C++ libraries, making it an ideal interface to the SADIE library routines described above. Descriptions of the current features of Tcl/Tk are available from Welch [30] and Harrison and McLennan [9].

The primary advantage of choosing Tcl/Tk as the interface development language for TREES is the fact that it allows cross-platform compatibility. Tcl/Tk was originally developed for use under the UNIX environment, but it has since been ported to both Windows and Macintosh. The initial version of TREES is being developed on a Sun workstation under X-windows. However, in the future Tcl/Tk will allow the vast majority of the program to be ported to another variant of UNIX or even to a Windows or Macintosh system with few or no changes required in the interface code. This would not be possible under alternate development languages such as Visual Basic or XView.

In addition to portability, Tcl/Tk has an added advantage of allowing rapid interface prototype development. Tcl is a very high level scripting language with a simple syntax that allows interface prototypes to be developed with a minimal amount of code. Since Tcl code is interpreted by a run-time compiler, there is no need to recompile the code when changes are made during the development and debugging stages, which saves time as well.

2.2.2 Interfacing Tcl/Tk to SADIE

A set of C libraries are included with Tcl that accommodate the sharing of information between a Tcl/Tk interface and the underlying C or C++ application libraries. The means of communication is generally based on the sharing of variables between Tcl and C or C++. Variables can either be passed by value between the two languages, or they can be linked so that a change in the variable shows up in both Tcl and C simultaneously.

Unlike C, Tcl is a very loosely typed language and stores variables internally in a flexible *object* format. When creating a variable in Tcl, there is no need to declare its type. Tcl will automatically store it in the appropriate object type. The default object types built into Tcl are boolean, integer, double precision floating point, and string types. Tcl can be extended to add any arbitrary data structure as a valid object type. However, the default object data types are sufficient for storing and sharing typical information such as SADIE image addresses and image processing algorithm parameters.

In an image analysis environment, it is imperative that a graphical user interface be capable of displaying images on the screen. This is accommodated in Tcl/Tk through *photo images* and the *canvas widget*. A Tk photo image is a RGB raster image that can be displayed to the screen on a canvas, with scrollbars if the image is very large. Routines are built into Tcl/Tk that allow GIF and PNM images to be converted to photo images. A Tk extension has been developed which takes the address of a SADIE image that is loaded into memory as input and generates a photo image representation for display purposes.

In order to allow C functions, such as those available in the SADIE library, to be called from Tcl, they must be registered with the Tcl interpreter. This is typically accomplished by creating a special C procedure for each library function that is capable of receiving a set of standard arguments from a Tcl call. This procedure can

receive Tcl objects as input, convert them to the corresponding C data types, use them as input parameters to a specific library routine, and return the results to Tcl as one or more objects. This procedure can be registered with the Tcl interpreter via a Tcl library function that associates a new Tcl procedure name with a corresponding C function.

As an example of how a Tcl/Tk interface can communicate with the C SADIE library, suppose that a SADIE image is currently loaded into memory, and that its memory address is stored in a Tcl variable. When the analyst instructs the program to process the image by clicking on a button or choosing a menu command, the address of the SADIE image is passed to the corresponding C function that was registered with the interpreter. This C function then passes the image address as the input to a SADIE image processing routine, and receives as output the address of a newly generated output image. After returning the address of the output image to Tcl, a Tk photo image is generated from the SADIE image as described above. Finally the photo image is displayed to the analyst for feedback in a canvas widget on the screen.

2.2.3 Example Graphical User Interfaces from TREES

Figure 2.1 shows the main Tcl/Tk graphical user interface from the TREES program. The left column of the interface provides buttons to allow the analyst to command TREES to begin a particular step of analysis. These buttons provide a simplified means of access to the data processing steps outlined in the flow chart from Figure 1.3.

When the analyst clicks the mouse on the *Capture Frames* button, the interface shown in Figure 2.2 is opened on the screen to aid in the data acquisition stage of analysis. This interface allow an analyst to control the x-y positioning table while viewing the current field of view from the CCD camera in the upper-left hand corner of the GUI. The current position of the positioning table is tracked symbolically in a graphical coordinate system at the bottom of the GUI. The analyst is able to choose

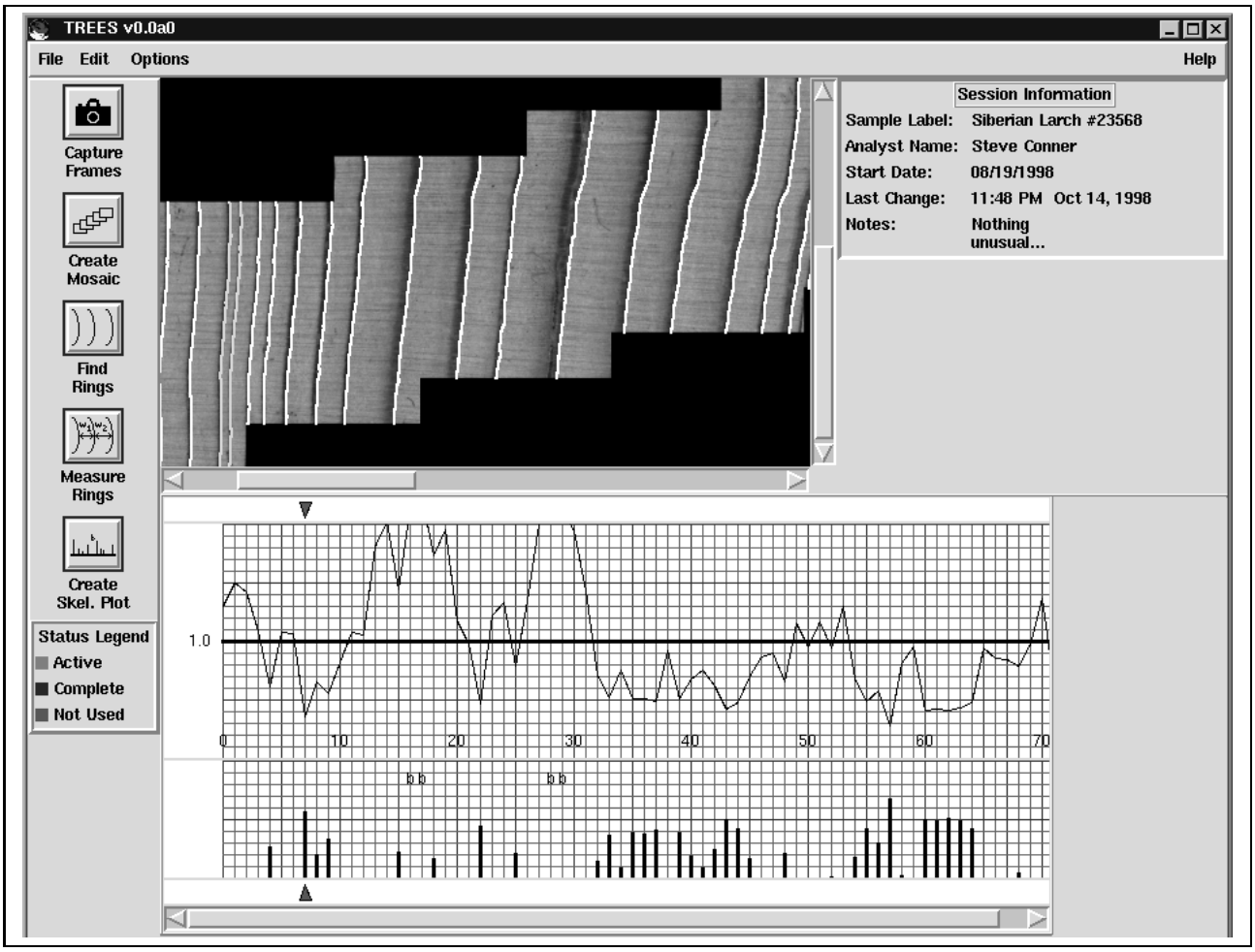


FIGURE 2.1. Main TREES graphical user interface.

the starting and ending points of a desired region of interest on the wood sample by placing *start* and *end* flags at the corresponding locations in the field of view in the top left corner. TREES then computes the required number of frames to capture the region of interest and symbolically displays them to the analyst in the coordinate system. After the analyst accepts these results, the system positions the stage at each frame location, finds the best focus position, captures the frame, and adjusts the frame for non-uniformity of illumination. This procedure is described in detail in Chapter 3.

After capturing the required frames for a sample, the main graphical user interface of Figure 2.1 returns to the screen. The analyst may then click on the *Create Mosaic* button in order to merge the individual frames into a single mosaic. The procedure to create a mosaic is described in Chapter 3. Once a mosaic is created, it is displayed as a low resolution browse image in a Tcl/Tk canvas at the top center of the GUI. The analyst may continue analyzing the sample by clicking on the *Find Rings* button. This commands TREES to perform edge detection, as described in Chapter 4. The detected ring boundaries are overlayed on the browse image for verification by the analyst. Next, the ring widths are measured using the techniques described in Chapter 5 by clicking on the *Measure Rings* button. Finally a computer-generated skeleton plot is provided at the bottom of the GUI when the analyst clicks on the *Create Skeleton Plot* button.

The lines in the skeleton plot are graphically linked to the tree ring boundaries shown in the mosaic browse image of the main TREES GUI. By pressing the right and left arrow keys on the keyboard, the arrows above and below the skeleton plot are shifted in the appropriate direction, and the corresponding tree ring boundaries are highlighted on the mosaic. This linking allows an analyst to resolve any problems in the skeleton plot by easily returning to the wood.

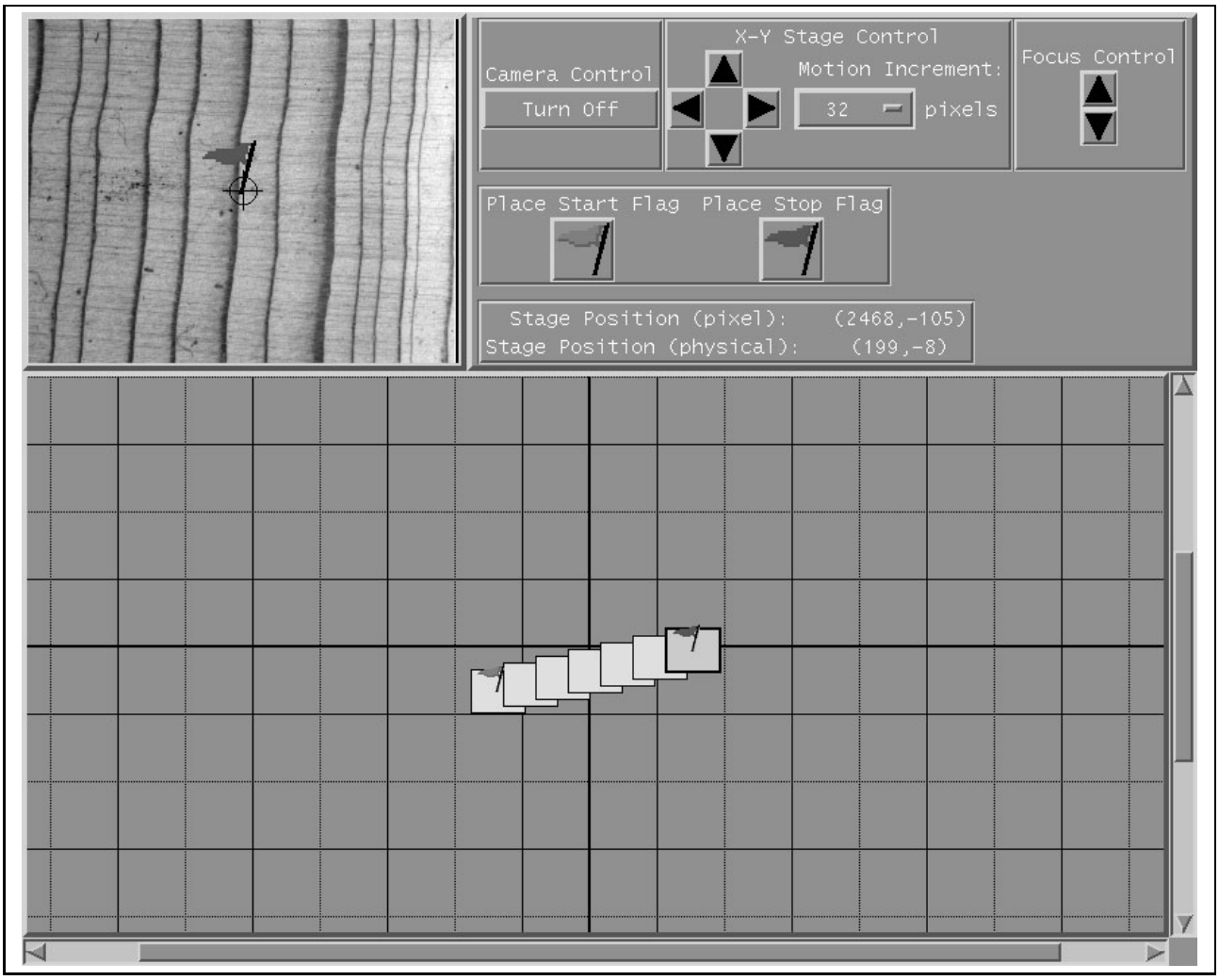


FIGURE 2.2. Graphical user interface used to capture a sequence of frames. The frame in the upper-left corner displays a live video image of the current microscope field of view, while the coordinate system at the bottom of the interface provides a reference to allow the analyst to track the position of the stage. After the analyst selects a starting and ending point on the sample, a graphical representation of the N frames required to capture image data of the selected region of interest is displayed in the coordinate system.

2.3 Shared Database

After tree ring edges are detected via computer vision and image processing algorithms, a number of features need to be stored for each tree ring in the sample. For this reason, a database of tree ring information needs to be created that can be shared and updated by various components of the TREES software. The image processing algorithms that process and measure the rings will generate an initial version of the database. Graphical user interface routines must access the database in order to create skeleton plots, and have the ability to update the database when an analyst modifies a skeleton plot. In the future, automated cross-dating and quality control procedures will need to have read and write access to the database in order to perform correlation and pattern matching algorithms on the data and fix any identified problems.

Since the functionality of Tcl is already built into TREES for graphical user interface purposes, the initial shared database has been developed in Tcl. The primary advantages of building such a database in Tcl are that it allows heterogeneous data types to easily be assembled into a single data structure, it simplifies the sharing of the database between the various components of the system, and the file I/O procedures built into Tcl make it fairly easy to store and retrieve the database from disk.

The Tcl constructs used to create the shared database include a combination of lists and arrays. A global Tcl list is used to store a list of variable names, where each variable represents a single tree ring. Each tree ring variable is itself a global Tcl array. Each array contains an element for the measured tree ring width as well as an element representing the normalized tree ring index value. The normalized index value is a representation of the measured ring width after low frequency trend has been removed from the series. Next a skeleton plot line length element is included in the array. This value represents the length of the symbolic line, if any, that is drawn on the skeleton plot for the given year, as described in Section 1.3.1. A list of tags is

included as an element of the tree ring array, representing such features as frost rings, micro rings, and exceptionally wide rings, which can be useful for pattern matching purposes. Finally, a free-format string of arbitrary length is included as an element of the tree ring array to store textual notes from the analyst for future reference.

When an analyst is working with a skeleton plot, missing rings may be added and false rings may be removed from the tree ring sequence. Since the tree ring sequence is stored as a Tcl list, which is similar to a linked list in C, this is a simple task. When a missing ring is added to the tree ring sequence, it can simply be added to an arbitrary location in the list. As long as the ring is tagged as a missing ring that has been inserted, it can be identified and removed from the sequence at a later time without difficulty. When removing false rings on the other hand, it is necessary to take an extra step in order to maintain the ability to return the ring to the list at a later time. Any time a tree ring is removed from the *main* list, it is moved to a second *removed ring* list rather than deleted. Each item in the *removed ring* list should contain both the removed tree ring array and the index at which it would be inserted if it was to return to the *main* tree ring list. Each time a tree ring is added or removed from the *main* tree ring list, the indices of each ring in the *removed ring* list must be updated accordingly.

Although the shared tree ring database has initially been developed in Tcl for simplicity, this is not a particularly efficient database implementation. In the future, it may be beneficial to develop a more efficient database implementation in C or C++.

Chapter 3
DATA ACQUISITION

3.1 Sample Preparation

Before beginning a new analysis session, the tree sample to be analyzed must be acquired and prepared. The two primary forms of tree ring samples that are used in analysis are cross-sections and core samples.

Tree cross-section samples are produced by cutting down a tree and cutting a cross-sectional disk from the trunk or a branch. In order for such a cross-sectional disk to be usable in the TREES system, it must be relatively flat on both sides so that it can rest on the stage with its surface approximately parallel to the imaging plane of the microscope. Secondly, it must be no more than two to three inches thick so that it can fit between the stage and microscope, with enough distance remaining to allow the surface to be brought into focus in the microscope.

Tree core samples are a less destructive form of a tree ring sample that are produced by drilling a small hole from the bark of the tree to the pith and extracting the resulting core. This type of sample tends to be rather fragile, and should be mounted on a stable base that can be anchored to the system positioning table without causing damage to the sample.

In order to allow analysis of the tree rings in either cross-sectional or core samples, the surface of the sample must be sanded and polished. This step is necessary to remove scratches and grooves from the wood surface that otherwise might be interpreted as tree ring boundaries. Before imaging the samples, they should be free of any large saw-dust particles, hairs, etc. that might impede analysis of the tree rings in the sample.

All of the sample preparation steps listed here are typically performed before

traditional visual analysis of tree rings. Thus the TREES system does not require much additional time to prepare samples than is already being spent on a daily basis in dendrochronology labs.

3.2 Sequential Capture of Images

In most cases it is not necessary to examine the entire cross-sectional disk of a tree in order to measure tree ring widths. Generally one, or a few, radial sections of the tree will suffice, as long as enough of each ring is visible to avoid distortions from resin ducts, branch scars, etc. An example of a series of frames which comprise a radial section of a tree are shown in Figure 3.1.

In TREES, the technique for specifying the portion of a sample to be captured involves the creation of a coordinate system for simultaneously tracking the physical location of the positioning table and the pixel position of a live video image. In order to reduce hardware costs of the system, no digital control of the microscope zoom function is available. Thus, in order for the system to be able to reliably relate pixel distances to positioning table distances, it is necessary for the microscope zoom setting to remain fixed throughout the data acquisition stage of the TREES analysis process. An analyst must choose an appropriate optical zoom setting at the beginning of a session based on the physical properties of the sample being studied. The microscope zoom cannot be changed during the capture of a sequence of images to be mosaiced. However, after the mosaic has been created, the analyst is free to optically zoom on a region of the wood that needs to be visually analyzed in more detail as long as no additional digital images need to be captured and registered with the mosaic.

The first step in implementing this coordinate system to relate pixel distances to positioning table distances is to calibrate the number of pixels per positioning table motor step. This is accomplished with the help of a pre-fabricated target consisting of a uniform background with a single, small foreground object such as a circle.

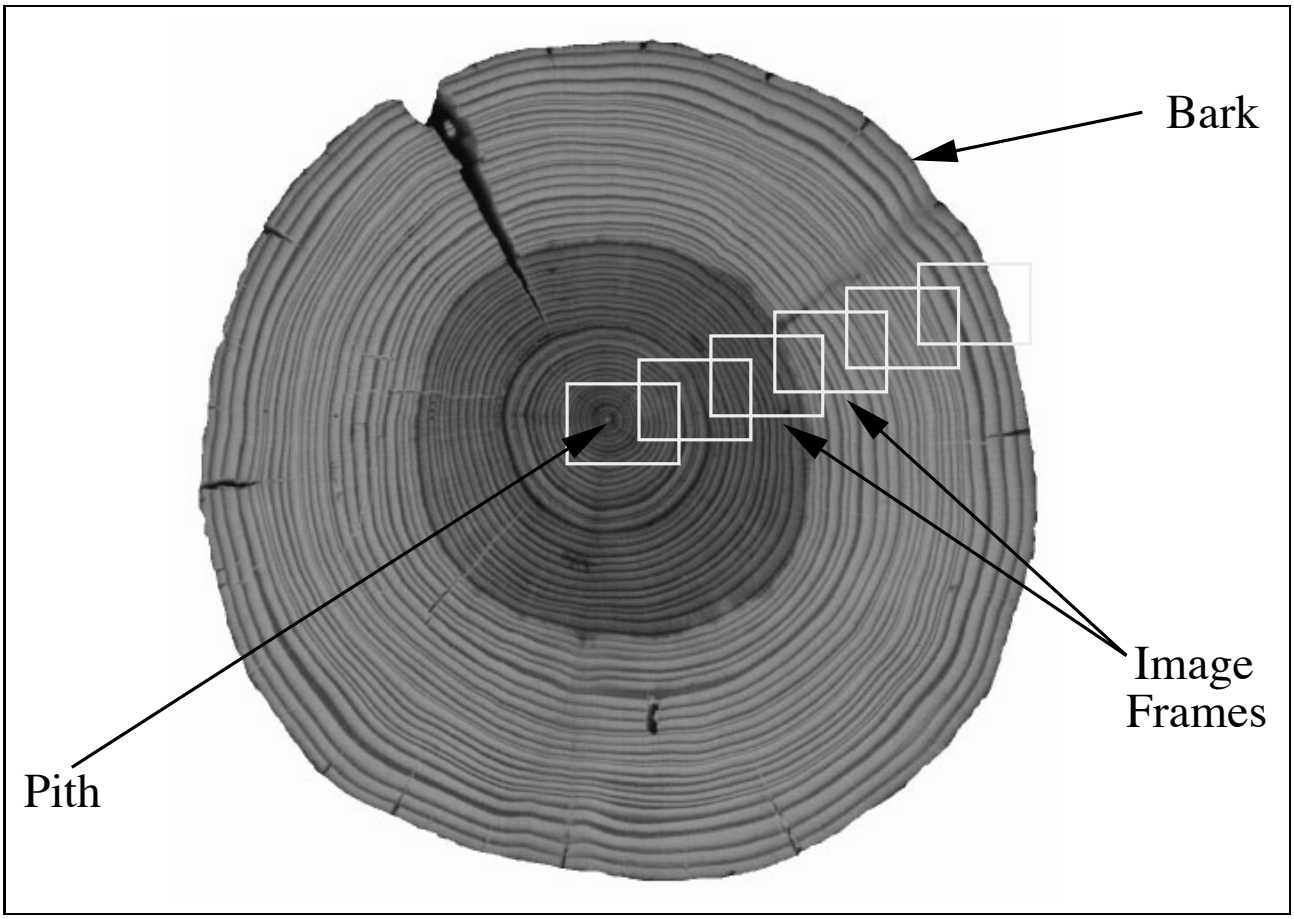


FIGURE 3.1. Example set of captured image frames.

This target is centered on the positioning table under the microscope and the system captures a digital image of the target before and after moving the positioning table by n-steps in the x and y directions. The center of the foreground object in each image is computed using a center of mass calculation as described in [12]. If the computed difference in position of the foreground object between the first and second images of the target is $(\Delta x, \Delta y)$, then the corresponding x and y pixels per positioning table step are $\Delta x/n$ and $\Delta y/n$. This experiment can be repeated multiple times to obtain average calibration values. A typical value from our system, using a magnification of 0.693X, is approximately 12.4 pixels per step in both the x and y directions.

Figure 2.2 shows an interface that allows an analyst to view a live video image while moving the positioning table to find the starting and ending points of a linear sequence of images to capture. Once these points have been determined, the required number of frames, N, to fully capture the area of interest is computed, assuming a minimum amount of overlap between adjacent images. The minimum required overlap is defined to be $overlap_{min}$ pixels and the total pixel offsets between the starting and ending frame positions as specified by the analyst through the image capture GUI are Δx_{total} and Δy_{total}. Specifying the width and height of each frame as w and h, the equation used to determine the minimum number of required frames to capture, when $\Delta x_{total} > \Delta y_{total}$, is

$$N = \frac{\Delta x_{total}}{w - overlap_{min}} + 1, \tag{3.1}$$

or if $\Delta y_{total} \geq \Delta x_{total}$,

$$N = \frac{\Delta y_{total}}{h - overlap_{min}} + 1. \tag{3.2}$$

Once N has been determined, the actual amount of overlap between consecutive frames is computed as:

$$overlap_x = w - \frac{\Delta x_{total}}{N - 1}, \tag{3.3}$$

$$overlap_y = h - \frac{\Delta y_{total}}{N - 1}. \tag{3.4}$$

Finally, the pixel offsets which are actually used to determine how many x and y steps the positioning table must take between captured frames are computed as:

$$offset_x = w - overlap_x, \tag{3.5}$$

$$offset_y = h - overlap_y. \tag{3.6}$$

Once the necessary offsets are determined, the positioning table is commanded to move to the locations of each of the frames to be captured, and the corresponding images are obtained from the camera.

An advantage of keeping such a coordinate system is that the approximate pixel offsets of consecutive captured frames are determined while the images are being captured. This is useful later when mosaicing the individual frames into a single data set. At this stage, the analyst should also be prompted for which end of the sequence of frames is closest to the pith and which end is closest to the bark of the tree. This information will help to simplify the tree ring boundary identification algorithms described in Chapter 4 with minimal additional burden placed on the analyst.

During the image capture stage, it is the analyst's responsibility to ensure that valid tree ring data representing a radius of the tree sample is located between the starting and ending frames.

3.3 Automatic Focus Detection

Tree cross-sections must be polished smooth before analysis in order to enhance the visibility of rings. However, in many cases the surface across a sample is not perfectly flat. In addition, the front and back sides of a cross-section are often not exactly parallel. These factors often make it difficult to precisely align the surface of a tree ring sample with the microscope. Thus, as the position of a tree sample is changed by moving the positioning table, the distance between the microscope and wood surface may vary enough to require a focus adjustment.

If a large number of frames are captured during a session, manual adjustment of the focus at each capture position may become a tedious burden. For this reason, an automated focus detection method is used. The technique used to determine whether or not an image is in focus uses the image power spectrum. The power spectrum returns information about the frequency-domain characteristics of the image. As the image goes out of focus, high frequency features will become blurred and the corresponding high frequency power spectrum characteristics will decrease in magnitude.

The power spectrum of images is approximated by the periodogram technique [31]. The largest power of two number of pixels and number of lines within a captured image are used to compute the periodogram. For instance, if an image is 1280x1024 in size, the first 1024x1024 pixels will be analyzed. If the width to be analyzed is W and the height is H, the data is partitioned into nine overlapping regions of size $(W/2)$x$(H/2)$, spaced at horizontal offsets of $W/4$ and vertical offsets of $V/4$. The magnitude squared of the Fourier Transform is computed for each of the nine overlapping regions. The two-dimensional power spectrum is then estimated by averaging the magnitude squared output of each of the overlapping regions. For the purposes of detecting focus, we are only interested in the magnitude of the power spectrum at high frequencies and do not care about the orientations of various features in the power spectrum. For this reason, the two-dimensional power spectrum is reduced to one dimension by performing radial averaging, as shown in Figure 3.2.

In order to test the sensitivity of the image power spectrum to focus, a series of 32 image frames of a typical tree ring sample were captured at different focus settings. Periodograms were computed for each frame for comparison. In this case, the camera was manually focused and image frames were captured at 15 focus intervals below and above this point, resulting in frame number 15 having the best focus and frames 0 and 32 having the worst focus. Figure 3.3 shows the radially-averaged power spectra for a set of frames, with the maximum at higher frequencies occurring near frame 15. However, it is difficult to compare the power spectrum response at all frequencies

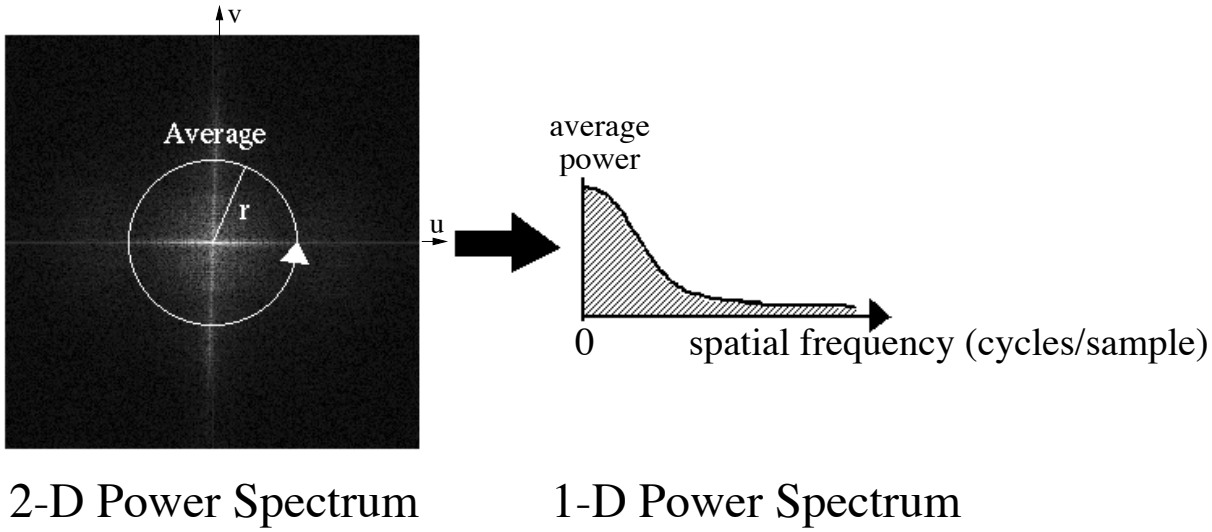


FIGURE 3.2. Radial averaging of a 2-D power spectrum into 1-D profile, where (u, v) are the 2-dimensional spatial frequency components.

because the overall power spectrum for all frames rapidly decreases in magnitude as the frequency increases. Thus it is necessary to normalize the power spectrum in order to perform a valid comparison of the sensitivity to focus variation across different frequency ranges.

The first normalization step converts each power spectrum value to a percentage of the maximum power spectrum value for the given image. Since the zero-frequency power spectrum value is far greater than any higher frequency values, each frequency value is divided by this zero-frequency value. The low frequency range will have a value near one, while high frequencies quickly fall off toward zero. The more out of focus an image is, the faster its power spectrum should decrease at higher frequencies.

Once the power spectrum from each frame has been divided by the zero-frequency value, the individual frequencies must be further normalized across the set of images in order to determine which image has the best focus. This is accomplished by treating the power spectrum values for all frames at a given frequency as a data set. Each power spectrum value at the given frequency is divided by the mean value over all frames for that frequency, resulting in a normalized mean value of one at

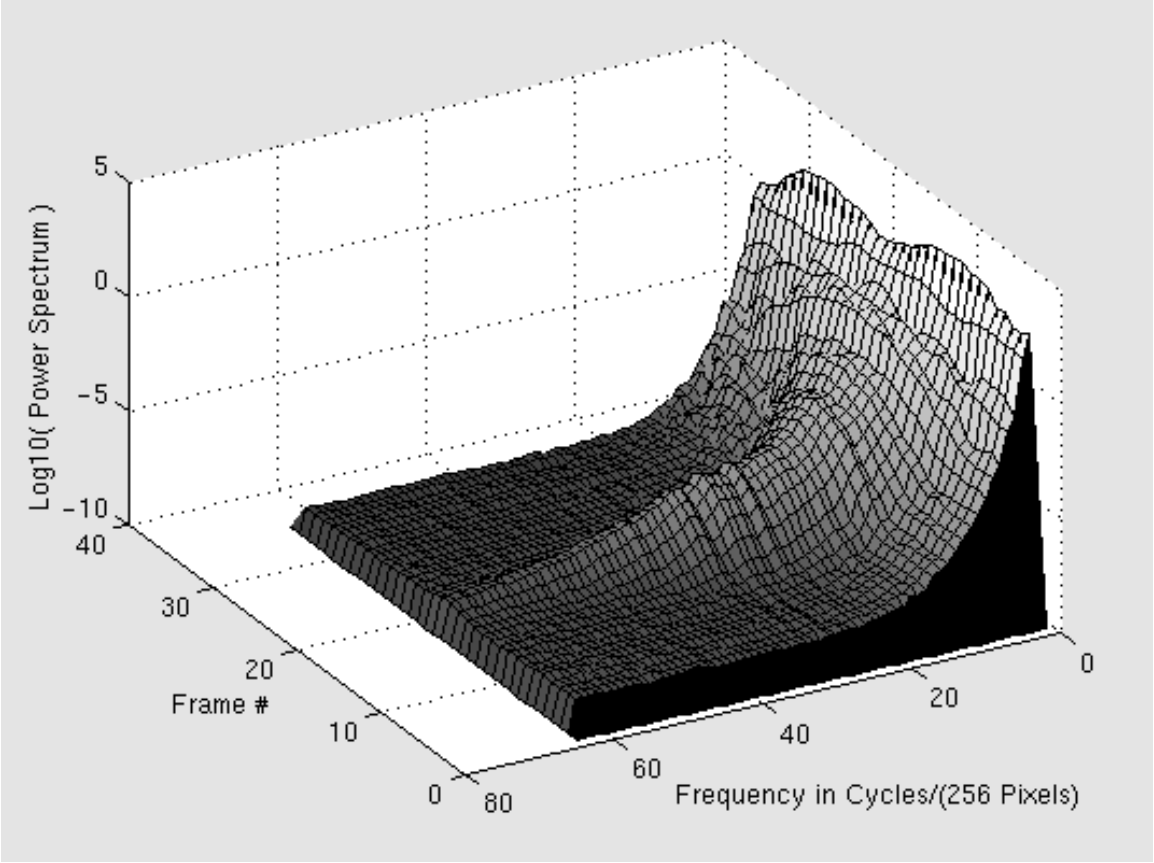


FIGURE 3.3. Power spectrum versus focus for tree ring images. Note that the lowest three frequencies are not shown because they are much larger in magnitude than the higher frequencies. Frame 15 has the best visual focus, while the rest of the frames are at stepped focus settings on either side of the best focus position.

all frequencies. An example normalized power spectrum for the set of 33 images described above is shown in Figure 3.4. In contrast to the non-normalized power spectrum of Figure 3.3, this version does not have a global decrease in magnitude as frequency is increased, but reveals a middle-frequency peak for the frame of best focus. This maximum is poorly defined at low frequencies since large-scale features are relatively unaffected by changes in focus. At very high frequencies, on the other hand, the power spectrum magnitude is nearly zero due to round off error, regardless of the focus setting. For this reason, it is appropriate to use a frequency range that is high enough to become blurred from defocus, but low enough to avoid excessive round off error. In the example from Figure 3.4, the optimal frequency range of interest is approximately 20/256 to 40/256 cycles per pixel ($W = H = 256$).

This analysis is applied in the data acquisition stage of the tree ring analysis system to help automate the process of capturing a sequence of images. Once the positioning table has been moved to a specified location to capture a given image, frames are captured at a range of focus settings. After computing a normalized power spectrum for this set of images, the output for each frame is summed across an optimum frequency range. The frame with the largest output sum is considered to be the best focused image, which is then retained for inclusion in the mosaic. If none of the acquired images are satisfactorily in-focus, the operator is alerted and must manually focus the system.

3.4 Flat Fielding

When capturing images that are to be mosaiced together, it is important that each is uniformly illuminated. As image magnification and field of view are varied from one wood sample to another, the distribution of light will vary within the field of view. At sufficiently high magnification, it is possible to view only the center of the illuminated area, where the lighting is brightest and the most uniform. However, as

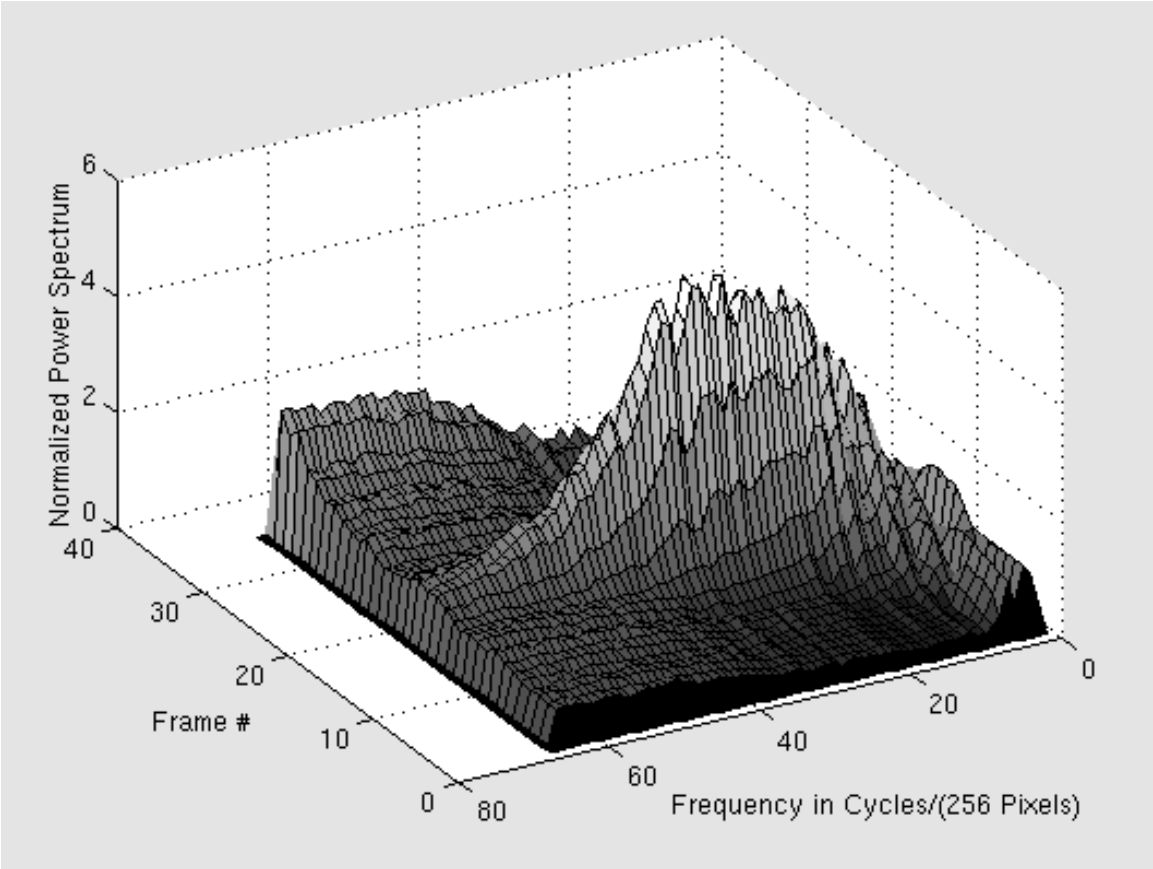


FIGURE 3.4. Normalized power spectrum versus focus for tree ring images. Note that the frequency range of 20/256 to 40/256 cycles per pixel seems to be ideal for the detection of the best focus at frame 15.

the magnification is decreased and the field of view increases, a larger portion of the illuminated area will become visible. The further the distance from the center of the illuminated area, the more the illumination varies. One technique that may be used to decrease the effects of nonuniform illumination is to use multiple illumination sources. However, this will not eliminate all illumination variation under all conditions. A software solution is more robust and allows for any illumination variation.

The software procedure used to remove illumination nonuniformity uses the "flat fielding" technique [3]. Once the magnification setting has been determined and locked for a particular session, a uniform calibration target is placed in the field of view. A sequence of frames of this calibration target is captured and averaged to produce a representation of the background lighting nonuniformity. An example of a target image showing bright illumination near the center of the field of view with a fall-off near the corners and edges is shown in Figure 3.5. This calibration image is then normalized by performing a linear stretch to set its mean value to one. The illumination nonuniformity is eliminated from images captured during a session by dividing each frame by the normalized calibration image. This typically will have the effect of increasing the brightness of the regions near the edges and corners while suppressing the brightness of the center of the field of view.

For best results, a new background illumination image should be captured each time a new sample is analyzed by the system. A given background illumination image is only valid at a particular magnification setting with the target placed at a particular distance from the microscope lens. Since tree ring sample sizes and thicknesses vary largely, even if the magnification of the microscope is held constant, the distance to the surface of the wood that is being imaged will likely change from one sample to another.

FIGURE 3.5. Example of illumination nonuniformity. The center-to-corner difference in graylevel is about 25%.

3.5 Image Registration

After individual frames have been captured at appropriate focus settings and corrected for any non-uniformity of illumination, they must be precisely registered to one another. This is necessary in order to fuse the individual frames into a single continuous mosaic image. In order to assure that no "seams" are visible at the locations of overlap, adjacent frames must be registered both geometrically and radiometrically.

3.5.1 Spatial Registration

The positioning table is controlled by independent horizontal and vertical motion. Thus, any pair of captured images from a given sample are related to each other by simple vertical and horizontal offsets. Approximate pixel offsets between consecutive frames are determined while the images are being captured. The images are then

precisely registered through correlation techniques [17, 20]. The cross-correlation coefficient provides a robust indication of correlation between two images. The primary advantage of the cross-correlation coefficient over other correlation techniques is that it is insensitive to bias and gain variation, the importance of which will be explained in Section 3.5.2.

Using the approximate pixel offset between a consecutive pair of images (obtained during pixel scale calibration described in Section 3.2), the estimated overlapping areas can be found. In the overlapping area of the first image, an N-by-N "target" area, T, is defined and a corresponding M-by-M "search" area, S, is defined in the second image, with M greater than N. In order to determine the position of best registration, the target region is shifted around the search region, computing the correlation at each location, with the maximum defining the point of best registration. If μ_S and μ_T are the mean pixel values of the search and target areas, respectively, the cross-correlation coefficient is defined as:

$$r_{ij} = \frac{\sum_{m=1}^{N}\sum_{n=1}^{N}(T_{mn}-\mu_T)(S_{i+m,j+n}-\mu_s)}{[\sum_{m=1}^{N}\sum_{n=1}^{N}(T_{mn}-\mu_T)^2]^{\frac{1}{2}}[\sum_{m=1}^{N}\sum_{n=1}^{N}(S_{i+m,j+n}-\mu_s)^2]^{\frac{1}{2}}}. \tag{3.7}$$

A large target area increases the accuracy of the correlation registration method. A large search area relative to the size of the target area allows for a less accurate initial estimated offset between two images. However, larger target and search areas require more computation time. These are important trade-offs that must be considered when implementing an automated registration technique that will be used to register a large sequence of images.

When constraining the size of the target area in order to minimize computation time, the correlation method of automated registration works best when both the search and target regions contain unique, high contrast features. However, searching for ideal search and target locations before beginning the correlation adds complexity and computation time to the procedure. On the other hand, if a single predetermined search and target location are used for every set of images, it is possible that

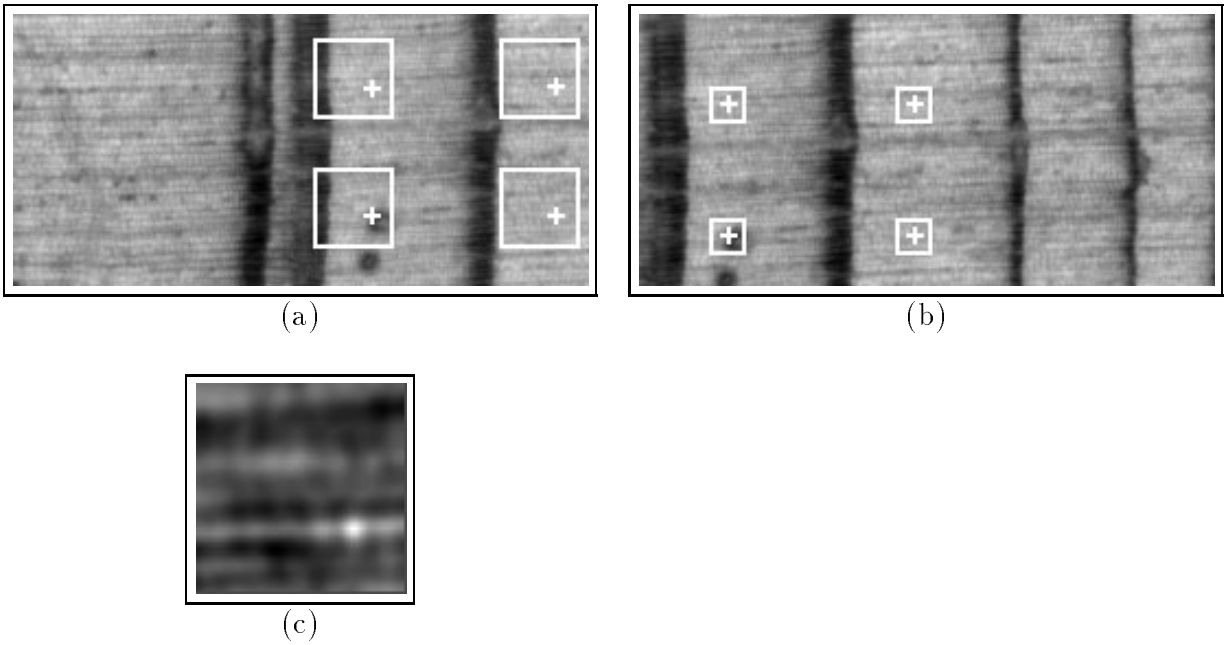


FIGURE 3.6. Search and target areas for registration correlation. (a) Search areas with markers at point of highest correlation. (b) Target areas with markers at centers. (c) Correlation results with best correlation at brightest point. Note that the correlation results are not to scale. In reality, the correlation matrix is approximately the same size as the search area.

these search and target areas may contain nearly uniform, low-contrast data that is not conducive to correlation. A compromise used in TREES involves using multiple preselected search and target areas distributed across the assumed overlapping areas, as illustrated in Figure 3.6. With this technique, n target areas are simultaneously shifted through the corresponding n search areas and the average correlation is used to determine the best registration position. This method avoids the additional complexity of searching for an ideal search and target area, while increasing the accuracy of the registration by averaging the correlations of multiple neighborhoods that are distributed throughout the overlapping areas of the images.

3.5.2 Gain and Bias Registration

In addition to spatial registration, overlapping images must be adjusted for any bias and/or gain variations due to voltage fluctuation and CCD camera electronic properties. As illustrated in Figure 3.7, it is important to perform contrast adjustments between overlapping frames to avoid the presence of "seams" at the edges of overlapping regions, which could be mistaken for tree ring boundaries during edge detection.

Figure 3.8 shows results from an experiment in which a set of 32 images of a single field of view were captured from a typical tree ring sample at 15 second intervals, and the mean and standard deviation were computed for each image. A linear gain/offset relationship exists between the captured images. This correspondence allows the contrast and brightness of overlapping images to be adjusted to one another through a linear stretch by computing the mean grayscale value, μ, and standard deviation, σ, of the overlapping region from each image. As an example, assume image *m-1* and image m overlap. If g_m and b_m are the relative gain and bias adjustments between the overlapping images *m-1* and m, then:

$$\sigma_{m-1}x + \mu_{m-1} = g_m(\sigma_m x + \mu_m) + b_m. \tag{3.8}$$

Grouping the additive and multiplicative terms results in:

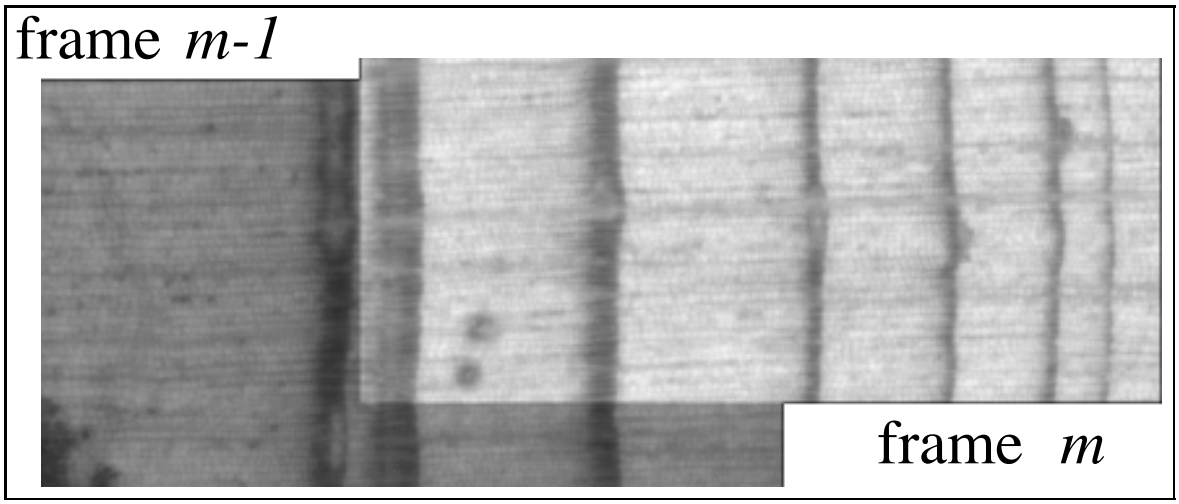
$$\sigma_{m-1}x + \mu_{m-1} = (g_m\sigma_m)x + (g_m\mu_m + b_m). \tag{3.9}$$

Thus the gain and bias relationship between the two images is:

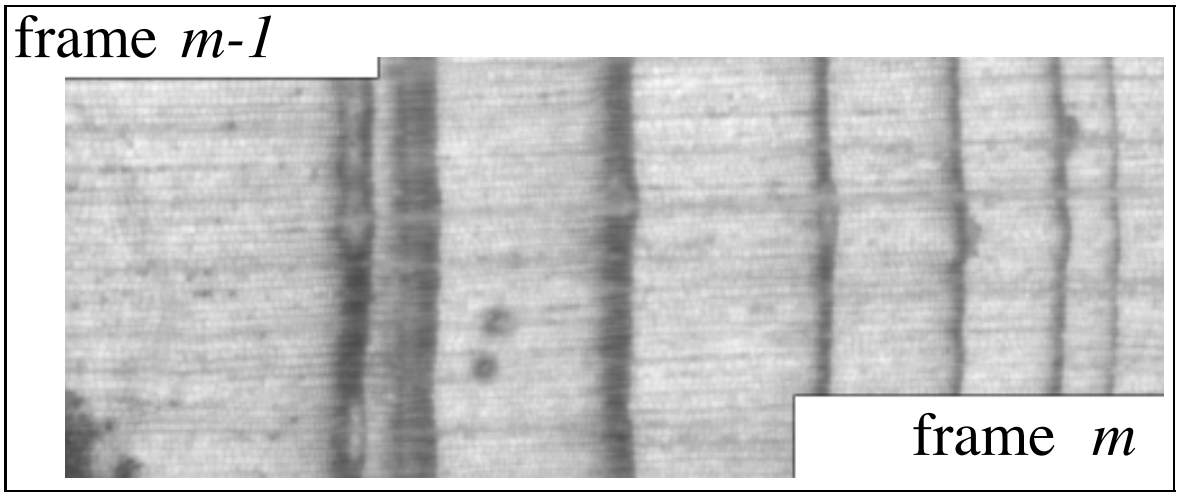
$$g_m = \frac{\sigma_{m-1}}{\sigma_m}, \tag{3.10}$$

$$b_m = \mu_{m-1} - g_m\mu_m. \tag{3.11}$$

Once the relative gain and bias adjustments have been determined between each pair of captured images, they may be converted to global gain and bias adjustments relative to a single reference image. For example, if the first image in a sequence of n



(a)



(b)

FIGURE 3.7. Effects of gain / bias adjustment during registration. (a) Mosaic of two frames without gain/bias adjustments. Note that a "seam" exists between the two frames. The problem has been exaggerated for illustration purposes. (b) Mosaic of the same two frames with frame m adjusted for gain/bias variation to match frame $m-1$. Note that no x or y offset changes are made between (a) and (b). Note that the the images of both (a) and (b) have been linearly stretched for maximum contrast, thus the image in (b) appears brighter than frame $m-1$ from (a).

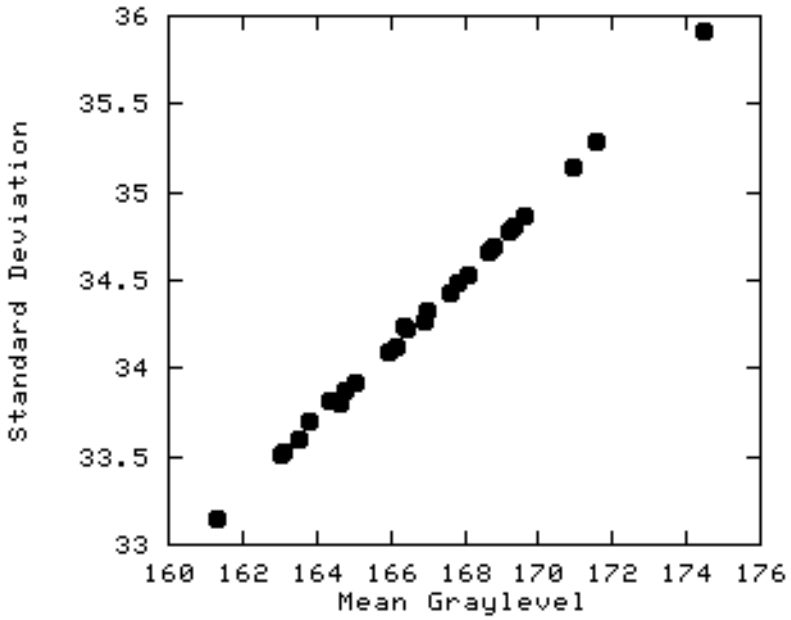


FIGURE 3.8. Mean and standard deviation variation during image capture. 32 frames were acquired of a stationary sample. The plot corresponds to the mean grayscale and standard deviation values for each frame. The linear nature of the plot indicates that a linear grayscale stretch can be used to match the gain and bias of one frame to another.

images is the reference and the relative gain and bias for image m refer to the gain and bias adjustments between frame m and frame *m-1*, the global gain, γ, and global bias, β, adjustments are:

$$\gamma_1 = 1, \tag{3.12}$$

$$\beta_1 = 0, \tag{3.13}$$

$$\gamma_m = \gamma_{m-1} g_m, \quad m = 2, 3, ..., n, \tag{3.14}$$

$$\beta_m = (b_m \gamma_{m-1}) + \beta_{m-1}, \quad m = 2, 3, ..., n. \tag{3.15}$$

Note that this frame-to-frame adjustment must be performed after the field of each frame has been adjusted for illumination non-uniformity (Section 3.4).

3.6 Creation of Mosaic

Once the individual frames have been properly registered to one another, both spatially and in gain and bias, they are combined into a single continuous mosaic image. There are several advantages to replacing the individual frames with a single mosaic file. First, ambiguities may exist in the overlapping area between any pair of frames. In a mosaic image, image data from any given overlapping area is taken from a single frame, guaranteeing that all future processing applied to the region is performed on a single set of data. Also, as long as the frames are properly registered, both spatially and for gain and bias variation, noticeable seams between adjacent frames can be largely removed. This provides a smooth, continuous transition between frames, which is especially beneficial at the edge detection stage, when we want to avoid mistaking the transition between frames for a tree ring boundary. Finally, a single mosaic file simplifies software implementation, since only one source of input data needs to be considered.

The primary disadvantage of a single, large mosaic file is that it can be very inefficient in terms of memory requirements if stored in a simple raster format. This

problem is illustrated in Figure 3.9 (a). Analysts are encouraged to orient a tree ring sample on the positioning table such that the sequence of frames to be captured will be as horizontal as possible. However, in some samples a horizontal linear sequence of frames is not feasible. As the angle of orientation of the sequence of frames increases, a larger and larger percentage of the raster format mosaic becomes wasteful fill pixels that do not contain any useful image data.

In order to reduce the overhead required when storing a large mosaic image, a new MOSAIC_IMAGE format has been added to the SADIE image processing library. This format reduces the number of fill pixels in the mosaic by reducing the mosaic to its minimum required vertical size, while maintaining the original width. This is accomplished by aligning each column of image data vertically at the top of the MOSAIC_IMAGE. In Figure 3.9, (b) is a MOSAIC_IMAGE format version of the example mosaic shown in (a). It is necessary to maintain a small number of filler pixels below the non-overlapping columns of image data in order to maintain a rectangular array structure. However, the percentage of filler pixels is significantly reduced from the full-size raster format mosaic layout, especially for a large number of frames that are captured in a sequence at an angle.

As a measure of the reduction of overhead pixels required in the MOSAIC_IMAGE, consider Figure 3.10. In (a), assume that F filler pixels are included in the rectangle below the upper-right frame. In the case of two frames in the mosaic, $2F$ filler pixels are present. In the case of three frames, as shown in (b), a total of $6F$ filler pixels are present in a normal raster image. Finally, in the case of four frames, shown in (c), a total of $12F$ non-image data pixels are included in a normal raster representation of the mosaic. Note that this example assumes the average offsets between any pair of images remains relatively constant throughout the mosaic (i.e. the scan path is linear), which is the case for mosaics created with the TREES system. In general, for a mosaic of N frames, the total number of overhead pixels in a normal raster mosaic

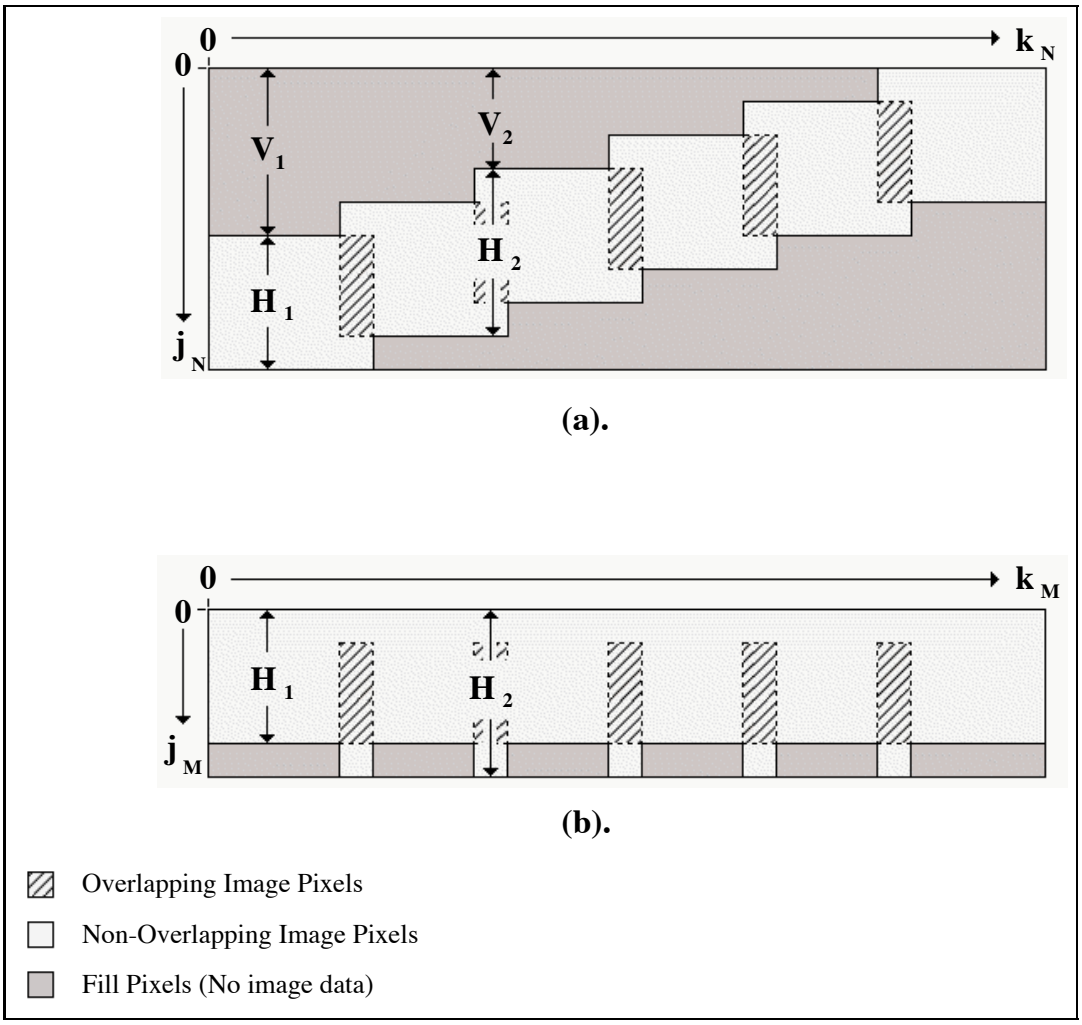


FIGURE 3.9. Comparison of a normal raster mosaic to the MOSAIC_IMAGE format. (a). A mosaic image consisting of six frames stored in a normal raster layout. (b). The same mosaic image stored in a more efficient MOSAIC_IMAGE format. Note that the width of both mosaics are identical, but the height of the MOSAIC_IMAGE is significantly reduced by aligning the top of each frame in the mosaic. The pixels within a given column of image data are not rearranged in any way, but complete columns of image data may be shifted vertically in order to reduce the number of overhead "filler" pixels in the mosaic. In order to extract data from the MOSAIC_IMAGE format, a table of vertical offsets, V_i, and heights, H_i, of each column, i, of image data in the mosaic must be stored. Note that $k_M = k_N$, $j_M \leq j_N$.

image is:

$$overhead_{raster} \approx (N^2 - N)F. \tag{3.16}$$

Thus, although F can be reduced linearly by making the orientation of the mosaic as horizontal as possible, the overhead grows at a rate of N^2 as the number of frames in the mosaic becomes large. The overhead required by the MOSAIC_IMAGE format, on the other hand, is:

$$overhead_{mosaic\_image} \approx NF. \tag{3.17}$$

By using the MOSAIC_IMAGE format, the number of overhead filler pixels in the image is reduced by a factor of $N - 1$.

The MOSAIC_IMAGE format has the ability to dramatically reduce the amount of memory required to store a mosaic image without adding the complexity of image compression. However, it is only useful if the coordinates of its image data can be easily converted to the equivalent full-size raster coordinates. When a MOSAIC_IMAGE is created, it is necessary to simultaneously create a table to store the vertical offsets, V_i, and heights, H_i, of each column in the mosaic. The relationship between the coordinates of the full-size raster image and the corresponding MOSAIC_IMAGE is:

$$column_{raster} = column_{mosaic\_image}, \tag{3.18}$$

$$line_{raster} = line_{mosaic\_image} + V_{column}. \tag{3.19}$$

While the MOSAIC_IMAGE format allows a convenient and efficient means of storage for a large mosaic image, it is not necessary to process the entire mosaic at one time during image processing and computer vision analysis. Sub-sections of image data corresponding to a given region of interest can be windowed sequentially out of the mosaic for processing. The advantage of windowing these sections from the mosaic rather than the original frames is that data can easily be acquired into a single window from multiple frames, without seams existing at the boundary between the image data from the different frames.

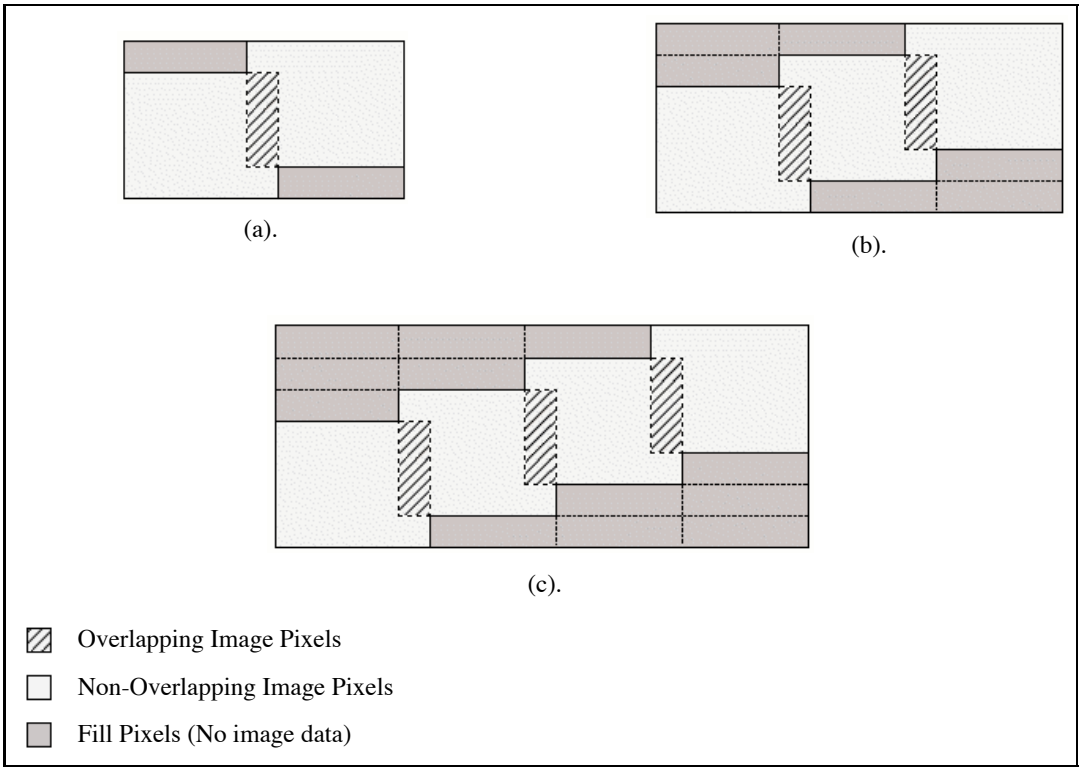


FIGURE 3.10. Measure of inefficiency of normal raster storage of a mosaic. (a). In this simple mosaic composed of two frames, assume that the number of overhead fill pixels below the upper-right frame is F. A total of $2F$ fill pixels are required in this case. (b). For three frames in the mosaic the overhead increases to $6F$. (c). For four frames, $12F$ filler pixels are required, and in general the number of overhead pixels required in a N-frame mosaic is approximately $(N^2 - N)F$.

Chapter 4
RING BOUNDARY DETECTION

Once a fully registered mosaic of image data is obtained, the next step is to automatically detect the locations of tree rings through edge detection algorithms that are optimized for the physical characteristics of tree ring samples.

The most common features in the image of a cross-section of a tree are summarized in Figure 4.1. In this image, tree rings run from the top to bottom along primarily vertical lines. These are the edges that must be detected in the image, while ignoring extraneous features. Resin ducts are dark circular structures which are distinguished from near-linear tree rings without too much difficulty simply by their shape. However, lines that run from the center of a tree outward toward the bark, referred to as "rays", and aligned cell boundaries are also nearly linear, and therefore less easily distinguished from rings.

4.1 Standard Edge Detection Algorithm Results

A number of edge detection algorithms exist in the computer vision literature, with various advantages and disadvantages. This section describes the evolution of the edge detection algorithms used in the the TREES project, which were designed primarily via trial and error of common edge detection techniques.

To begin experimentation, a set of images of typical tree ring samples was collected using the hardware described in Section 1.3.2. Example images of typical conifer samples are shown in Figure 4.2. A number of edge detection algorithms were applied to the images in this set. Initially, very simple gradient magnitude computations were applied to the images. Later, gradient direction information was used in addition to

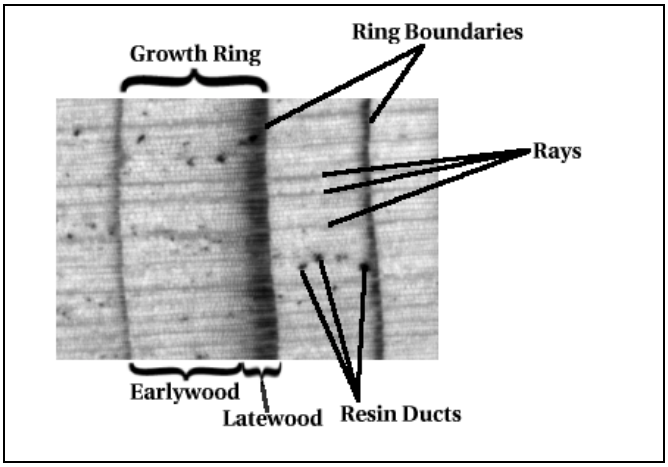


FIGURE 4.1. Typical characteristics of a conifer tree ring sample.

the magnitude to improve results. Finally, the Canny edge detector was applied to the images, and a variant of the Canny edge detector was developed and chosen as the edge detection algorithm of choice for the TREES project. The following subsections describe the experiments and their results.

4.1.1 Gradient Computation

The first edge detection algorithms applied to the sample of tree ring images were based on simple gradient computations. For a two-dimensional image, the gradient at each pixel location is equivalent to a vector including the first partial derivatives in both the x and y directions. However, a more useful representation of the gradient is given by its magnitude and direction. The gradient magnitude and direction can be computed from the x and y partial derivatives as:

$$\|G\| = \sqrt{\frac{\delta}{\delta x}^2 + \frac{\delta}{\delta y}^2} \quad , \tag{4.1}$$

$$\angle G = tan^{-1}\left(\frac{\frac{\delta}{\delta y}}{\frac{\delta}{\delta x}}\right) , \tag{4.2}$$

where $\frac{\delta}{\delta x}$ and $\frac{\delta}{\delta y}$ are the partial derivatives in the x and y directions, respectively. In a discrete digital image, the partial derivatives are approximated by first differences,

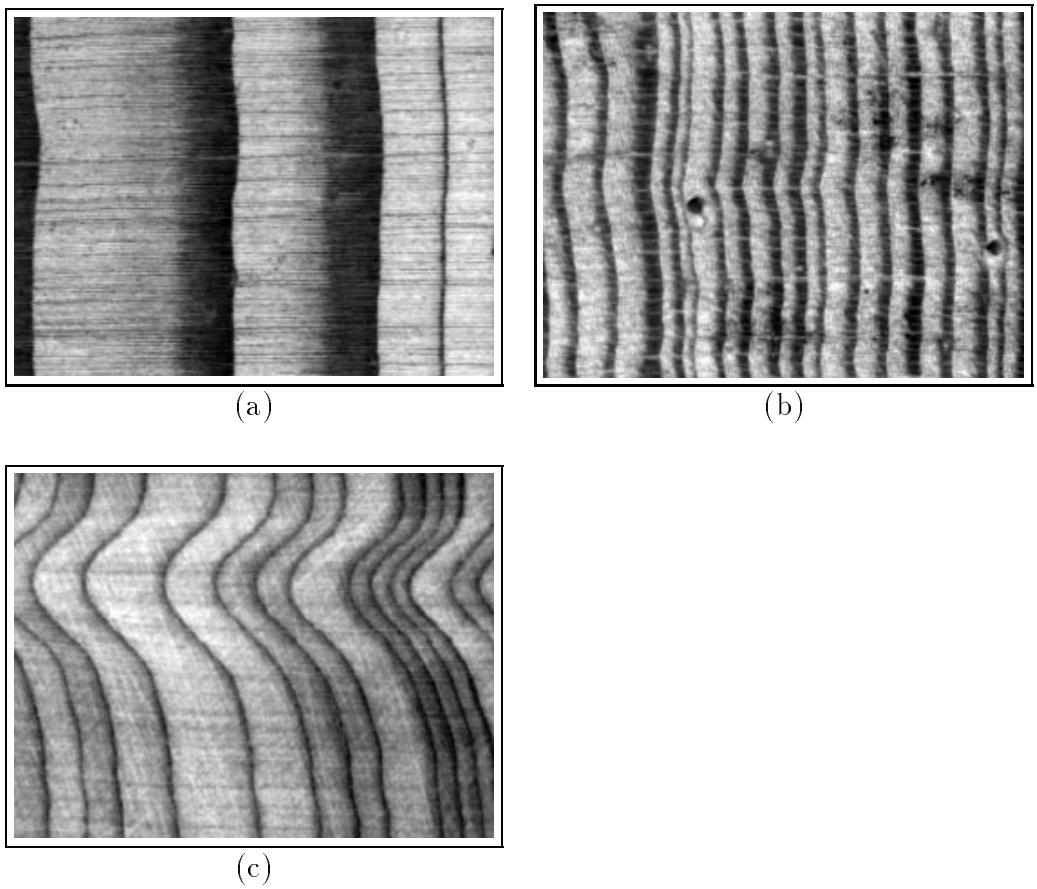


FIGURE 4.2. Typical tree ring image samples. (a) Ponderosa Pine sample. (b) Siberian Larch sample. (c) Juniper sample.

$$\Delta x = \begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline -2 & 0 & 2 \\ \hline -1 & 0 & 1 \\ \hline \end{array} \qquad \Delta y = \begin{array}{|c|c|c|}\hline 1 & 2 & 1 \\ \hline 0 & 0 & 0 \\ \hline -1 & -2 & -1 \\ \hline \end{array}$$

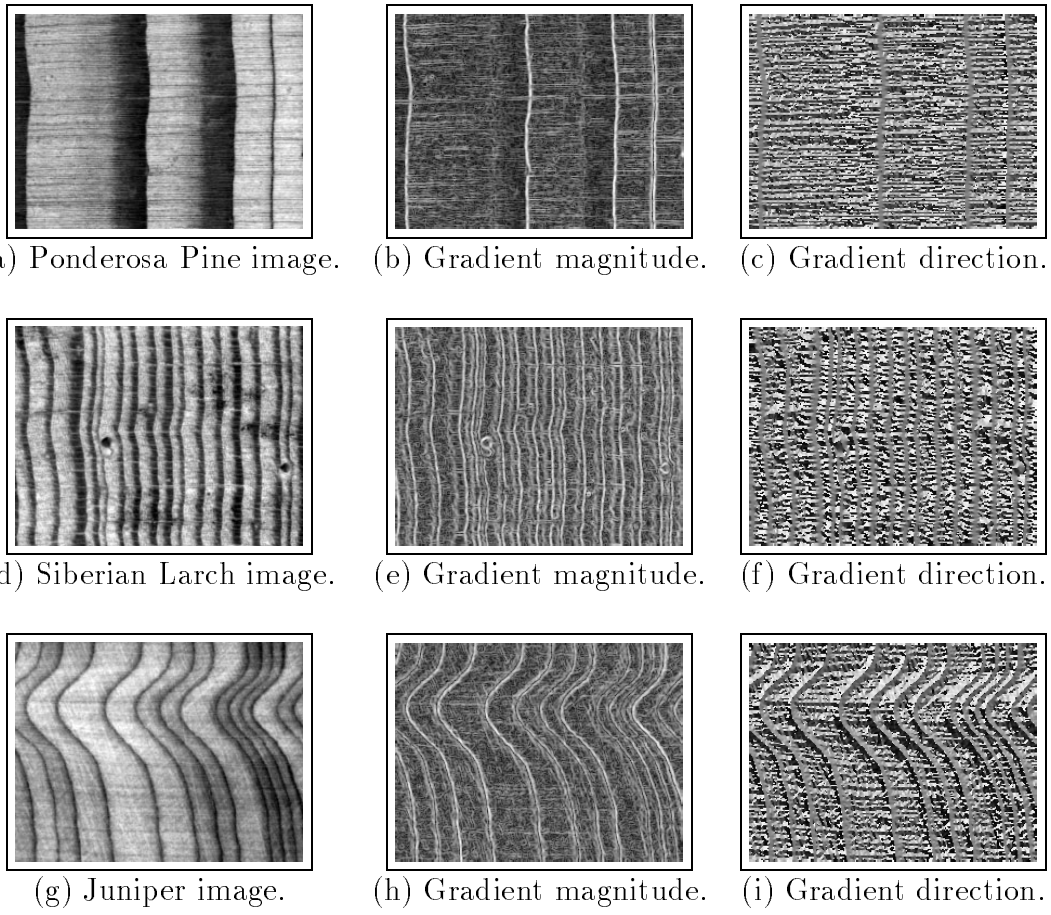
FIGURE 4.3. Gradient masks used in Sobel filter.

which are computed by convolving first difference masks with the image. The gradient masks for the Sobel filter are shown in Figure 4.3. The Sobel filter is a good choice for gradient computation in an image because it provides some noise averaging characteristics while giving pixels near the center of the 3x3 masks the highest weights. Figure 4.4 shows the gradient magnitude and gradient direction images that result from applying the Sobel filter to the test images of Figure 4.2.

The gradient magnitude is often used to find the edges of features in an image due to the fact that the boundaries between features normally have high contrast edges. This is the case in tree sample images, where the boundary between two tree rings is generally at the high contrast boundary between the dark latewood of one ring and the light earlywood of the next.

In order to extract the high contrast edges from a gradient magnitude image, it is necessary to apply a threshold. The threshold value must be high enough to exclude low contrast noise pixels that have a relatively low gradient magnitude, while maintaining the pixels with a high gradient magnitude that correspond to high contrast edges in the image. The choice of an appropriate threshold value varies largely between images, and is normally chosen visually. Jain, et. al., describe several methods of automated threshold selection [12].

Figure 4.5 (b), (e), and (h) show the edge maps that result from applying a visually adjusted threshold to the gradient magnitude of the sample images in Figure 4.2. These edge maps indeed contain the highest contrast edge features from the input images. However, there is no discrimination between the edge pixels at tree ring boundaries and other high contrast features. Since the tree rings in the images are aligned approximately vertically, a second processing step can be applied to filter

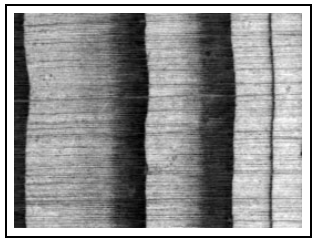


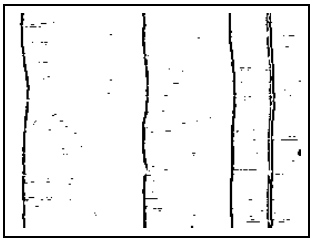
(a) Ponderosa Pine image. (b) Gradient magnitude. (c) Gradient direction.

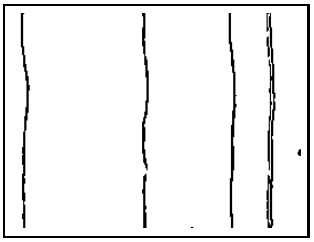
(d) Siberian Larch image. (e) Gradient magnitude. (f) Gradient direction.

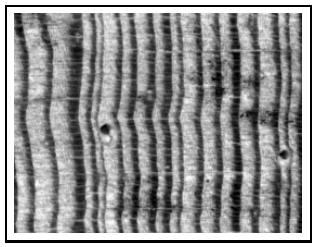
(g) Juniper image. (h) Gradient magnitude. (i) Gradient direction.

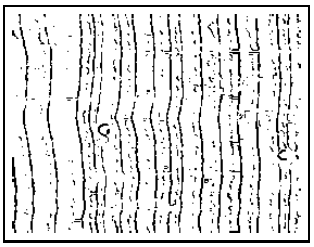
FIGURE 4.4. Gradient results.

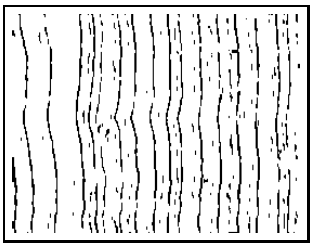

(a) Ponderosa Pine image.

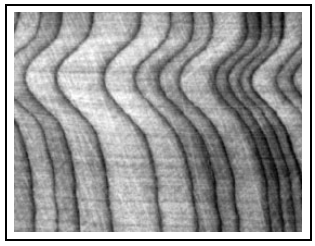

(b) Thresholded gradient magnitude.

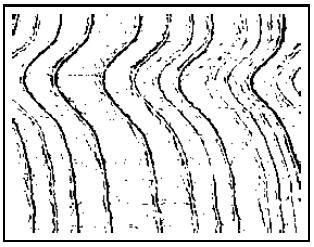

(c) Median filtered version of (b).


(d) Siberian Larch image.


(e) Thresholded gradient magnitude.


(f) Median filtered version of (e).


(g) Juniper image.


(h) Thresholded gradient magnitude.

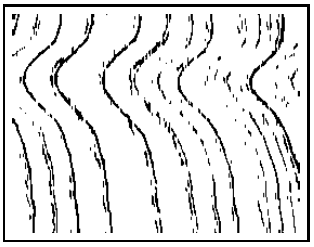

(i) Median filtered version of (h).

FIGURE 4.5. Gradient magnitude results. A median filter of size 1x7 was used to produce the filtered edge maps of (c), (f), and (i).

much of the noise from the thresholded edge map. By applying a vertical median filter (e.g., of size 1x7) to the edge map, any edge pixels lacking neighbors above and below will be discarded. Figure 4.5 (c), (f), and (i) show the results from filtering the edge maps with a median filter. The larger the median filter size, the more the noise pixels are suppressed. However, this also severely degrades the natural curves in the actual tree ring edges.

While the gradient magnitude combined with thresholding and median filtering provide a good first attempt at approximating tree ring boundaries, there are several disadvantages to this approach. First, the boundaries returned by the Sobel filter are

imprecise due to the fact that they are generally several pixels wide. Second, efforts to remove noise from the image result in severely degraded tree ring boundaries.

4.1.2 Canny Edge Detector

Edge detection results can be improved by applying information contained in the gradient direction as well. The Canny edge detector [2] applies both gradient magnitude and direction information in its implementation of edge detection. The Canny edge detector is a robust edge detection algorithm that produces good results on a wide range of image types.

One of the primary advantages of the Canny edge detector over other edge detection techniques is that it is guaranteed to produce one pixel wide, fully connected edge boundaries. This is especially useful in the detection of tree rings, because complete ring boundaries across the image are necessary to distinguish one ring from another. Also, a precise edge location increases the accuracy of other measured features such as tree ring widths.

A straightforward implementation of the Canny edge detector is described by Jain, et. al. [12]. The first step is smoothing the input image by convolving it with a Gaussian filter. A Gaussian with a standard deviation of three pixels seems to work well at the magnification used for the tree samples investigated to-date. This suppresses noise while maintaining the locations of high contrast edges. Next, the gradient magnitude and direction are computed at each pixel location, as described in Section 4.1.1. The most important step in Canny edge detection is nonmaxima suppression, which creates thin, fully-connected boundaries. Nonmaxima suppression is implemented by quantizing the gradient direction of each pixel into one of four sectors, shown in Figure 4.6. Each pixel is compared to the two neighboring pixels located in its sector. If the center pixel has a greater magnitude than either of its neighbors, it is considered to be a local maximum and the output value is set to

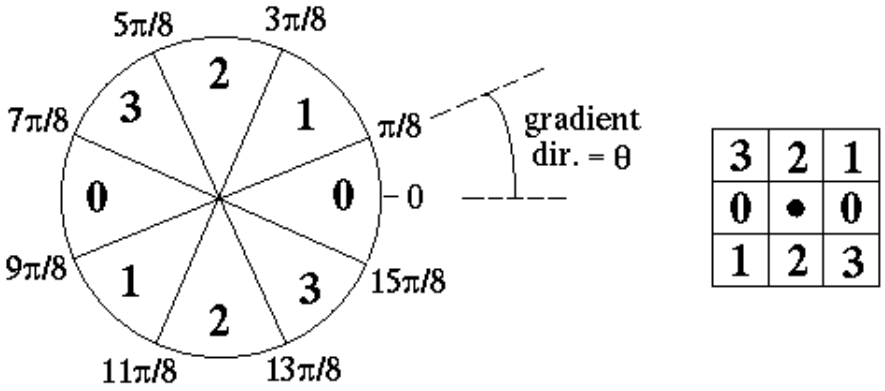


FIGURE 4.6. Direction sectors used in Canny edge detection. (a) Continuous directions. (b) Discrete directions.

its magnitude. Otherwise the pixel is suppressed, and its output is set to zero. The result is an edge map in which each edge pixel represents the location of the maximum gradient magnitude at that point on the edge.

Figure 4.7 (b) shows the result of the Canny edge detector applied to an example tree ring image, which appears to show very accurate ring boundaries. However, in addition to the latewood-earlywood boundary that represents the edge between two tree rings, it also detects the intra-ring earlywood-latewood boundary. In addition, prominent rays and other horizontal features in the image cause connections between tree ring edges. These shortcomings of the Canny edge detector in this application are eliminated by applying knowledge of the physical characteristics of tree rings.

4.2 Modified Canny Edge Detector

This section describes the design of an edge detection algorithm that is based on the Canny edge detector, but optimized for the detection of tree rings. The primary modification to the original Canny edge detector algorithm is that tree ring orientation within a region of interest is applied to suppress inter-ring connections.

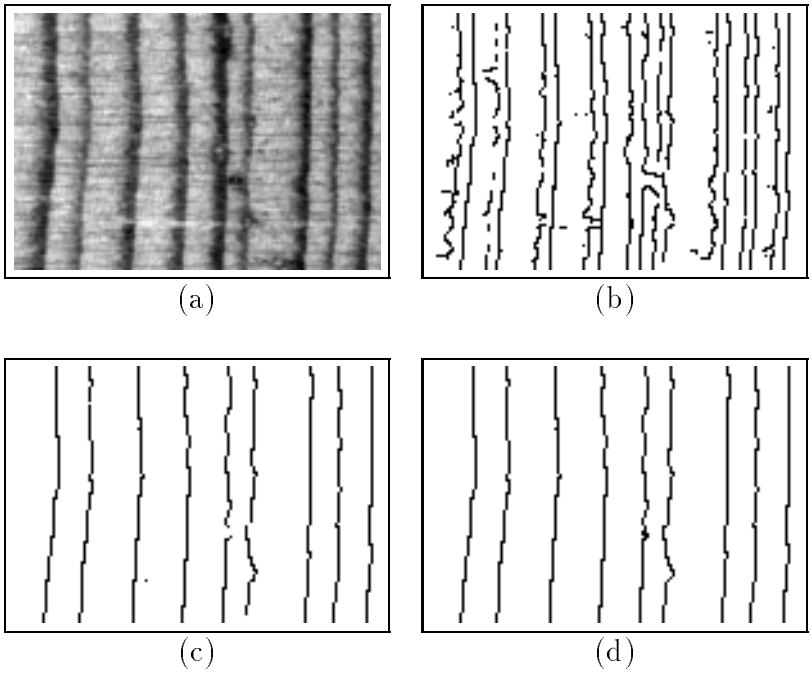


FIGURE 4.7. Canny edge detector results for an example tree ring image. (a) Sample Siberian larch image. (b) Ring boundaries detected with the Canny edge detector, with an appropriate threshold. (c) Ring boundaries detected with modified Canny edge detector. (d) Ring boundaries from (c) after performing fragment linking.

4.2.1 Tree Ring Orientation Tracking

It is useful to first attempt to determine the overall orientation of tree rings within a region of interest. This simplifies the process of ignoring rays and other features that are oriented orthogonally to tree rings during edge detection. Because ring orientations may not remain constant across an entire data sample, it is important to treat each sample as a set of regions in which the orientation of tree rings is fairly consistent; orientations within a region of interest are determined by computing the gradient with a directional filter such as Roberts or Sobel, as described in Section 4.1.1.

Tree ring boundaries are defined to be the boundary between the latewood of one year and the earlywood of the following year. Within a given ring, the transition between earlywood and latewood tends to be a gradual change from light colored wood to dark colored wood. At the boundary between rings, however, there is generally a very steep transition from the dark colored latewood of one year to the light colored earlywood of the next year. On average, the gradient magnitude of this latewood to earlywood transition is much higher than the gradient magnitude of earlywood to latewood boundaries and the edges of rays.

By quantizing the continuous gradient direction into eight discrete directions shown in Figure 4.8, the average gradient magnitude for each of the discrete directions can be computed. While traversing a region of interest, at each pixel location the discrete gradient direction is computed. A counter corresponding to this direction is incremented and the gradient magnitude of the current pixel is added onto a gradient magnitude sum for this direction. After performing this update at each pixel in the region of interest, the average gradient magnitude for each of the eight discrete directions can be computed by dividing the gradient magnitude sum for each direction by the corresponding count.

Based on the above characteristics of tree ring images, the direction with the largest average magnitude corresponds to the average direction of transition from the

4	3	2
5	●	1
6	7	8

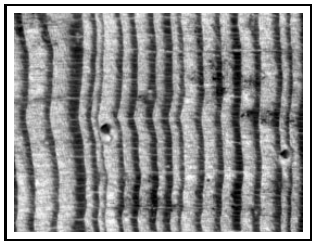
FIGURE 4.8. Discrete directions for assumed tree ring orientations.

latewood of one ring to the earlywood of the next within the region of interest. The maximum gradient magnitude direction corresponds to the direction from the pith of the tree to the bark, and it is orthogonal to the average tree ring edge within the region of interest. If the frame closest to the pith and the frame closest to the bark of the tree were identified by the analyst in the data acquisition stage (Section 3.2), this information can be used to verify that the computed tree ring orientations in a region of interest are reasonable.

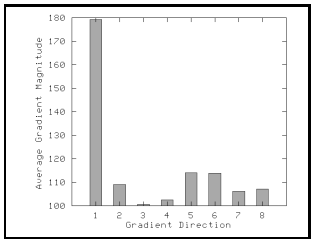
Figure 4.9 illustrates several examples of the average gradient magnitude calculation for each discrete gradient direction. In each case, this technique yields a strong correlation with the actual visible tree ring orientation.

4.2.2 Modified Non-Maxima Suppression

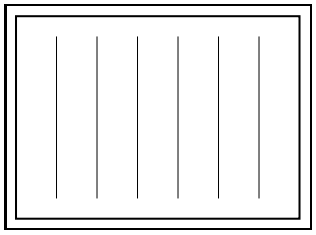
Once the general orientation of tree rings within a region of interest is known, this information may be applied to a variant of the Canny edge detector algorithm for efficient tree ring boundary detection. Modifications at the nonmaxima suppression stage of the Canny edge detector algorithm [12] allow it to be optimized for the detection of tree ring boundaries. While the original Canny edge detector performs directional nonmaxima suppression from local pixel data, it is more appropriate to use a global direction for nonmaxima suppression in tree ring images if the average orientation of tree rings within a region of interest is known. This will reduce sensitivity to local pixel variations. If tree rings are known to have one of the orientations shown in Figure 4.8, edge detection should be optimized to detect edges in this direc-



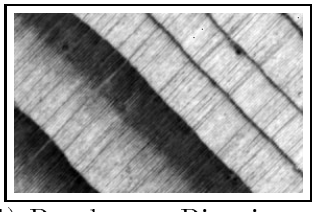
(a) Siberian Larch image.



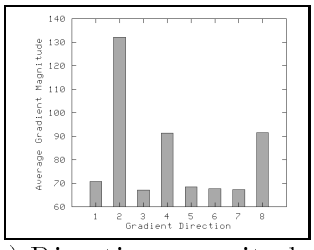
(b) Direction magnitudes.



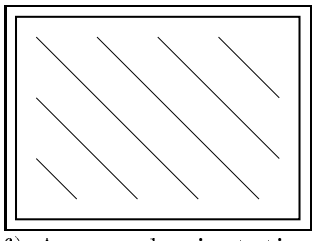
(c) Assumed orientation.



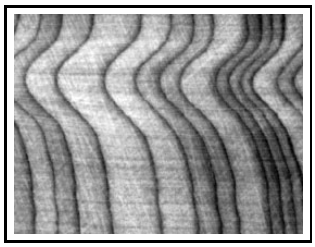
(d) Ponderosa Pine image.



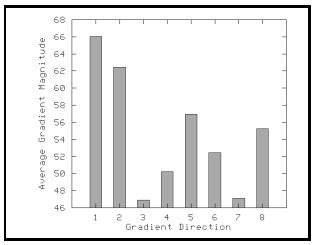
(e) Direction magnitudes.



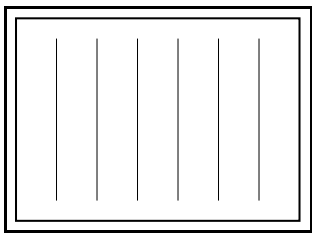
(f) Assumed orientation.



(g) Juniper image.



(h) Direction magnitudes.



(i) Assumed orientation.

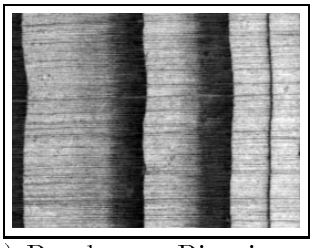
FIGURE 4.9. Example computed tree ring orientations based on the directions defined in Figure 4.8.

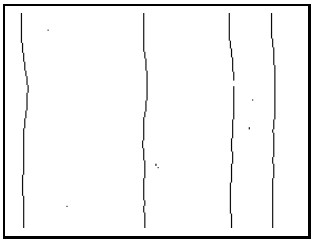
tion. This is accomplished through a two step process. First, at each pixel location, nonmaxima gradient magnitude suppression is implemented in the direction of the average orientation of tree rings, rather than in the local gradient direction at the pixel. For instance, if tree rings are determined to be approximately vertical within a region of interest as described in Section 4.2.1, the gradient magnitude of a pixel is only compared to that of its left and right neighbors. If the pixel has a larger gradient magnitude than either of these neighboring pixels, it is not suppressed. Second, assuming that the transition between the latewood of one ring and the earlywood of the next ring takes place from left to right, only edge pixels where the right neighbor has a larger grayscale value than the left neighbor are retained. This corresponds to the transition between dark latewood pixels and light earlywood pixels, and allows intra-ring earlywood to latewood transitions to be suppressed.

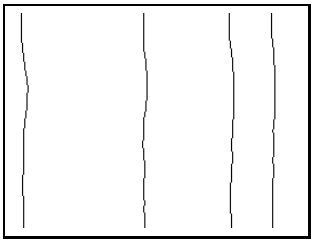
Figure 4.7 (c) and Figure 4.10 (b), (e), and (h) illustrate several examples of tree ring edges that are detected by this modified Canny edge detection algorithm, with an appropriate threshold. Although several breaks exist in some of the ring boundaries, earlywood to latewood transition edges and ring interconnections have successfully been eliminated. Furthermore, it is possible to produce fully connected tree ring boundaries from these results via a fragment linking postprocessing step.

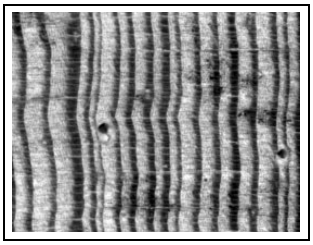
4.3 Producing Fully Connected Tree Ring Boundaries from Edge Detection Results

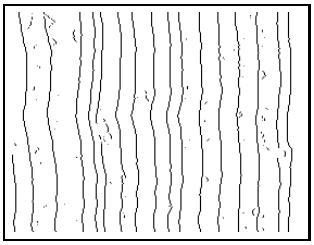
The raw data returned by the modified Canny edge detector algorithm is an edge map with each edge pixel representing the gradient magnitude value at that point. Any edge pixels that meet the criteria of being locally maximum in the average direction of orientation of tree rings will be included in the edge map, regardless of their absolute magnitude. This means that, in addition to the tree ring edges (which generally have a relatively high gradient magnitude), the edge map will also contain low contrast

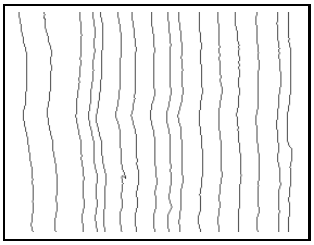

(a) Ponderosa Pine image.

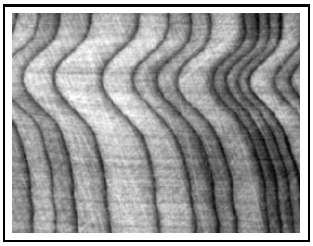

(b) Fragmented ring boundaries.

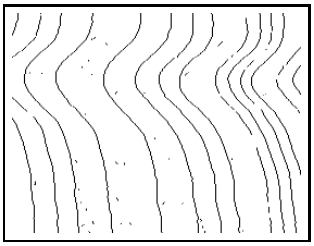

(c) Fully linked ring boundaries.


(d) Siberian Larch image.


(e) Fragmented ring boundaries.


(f) Fully linked ring boundaries.


(g) Juniper image.


(h) Fragmented ring boundaries.

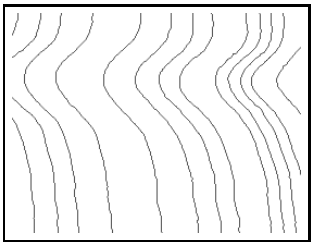

(i) Fully linked ring boundaries.

FIGURE 4.10. Results of the modified Canny edge detector. Note that these results were obtained assuming that tree rings are nearly vertical in the image. This edge detection technique is robust enough to maintain good edges, even in the areas where the rings are very curved, as in the case of the juniper sample.

edges representing noise features within the rings themselves. It is necessary to decide which edge pixels are tree ring boundaries that should be kept, and which are simply noise that should be removed.

An appropriate gradient magnitude threshold will remove most of the noise edges while maintaining the majority of tree ring edge pixels. The choice of a threshold value is aided by the fact that the histogram of the gradient magnitude values included in the edge map from most tree ring samples studied to-date is bimodal, as illustrated in Figure 4.11. An example of an appropriate threshold choice is shown in Figure 4.12. By choosing a threshold just short of the peak for high gradient magnitude edge pixels, a conservative estimate of which edge pixels belong to tree ring boundaries can be obtained. Unfortunately, the threshold will typically be high enough that some tree ring edge pixels will be discarded and also low enough that several noise edge pixels will remain. Currently, the threshold value is set visually, but in the future it should be determined by using the mode method of automatic thresholding [12] or a similar approach.

4.3.1 Labeling Edge Pixels

After applying a threshold to an edge map of gradient magnitude pixels, a binary image remains. In this binary edge map, zero-value pixels are considered to be background pixels that are not included in relevant edge boundaries. Any non-zero pixels are foreground edge pixels that may belong to valid tree ring boundaries or may simply be noise that must be filtered from the edge map. In order to distinguish between individual tree ring boundaries and noise edge pixels, it is helpful to label sets of edge pixels that are connected to each other as belong to a single foreground object. A connected-components labeling algorithm such as that listed as Algorithm 2.2 in [12], modified to detect 8-connected regions, is used to label foreground objects.

The "fuzziness" of the threshold decision region for rings versus noise results in tree

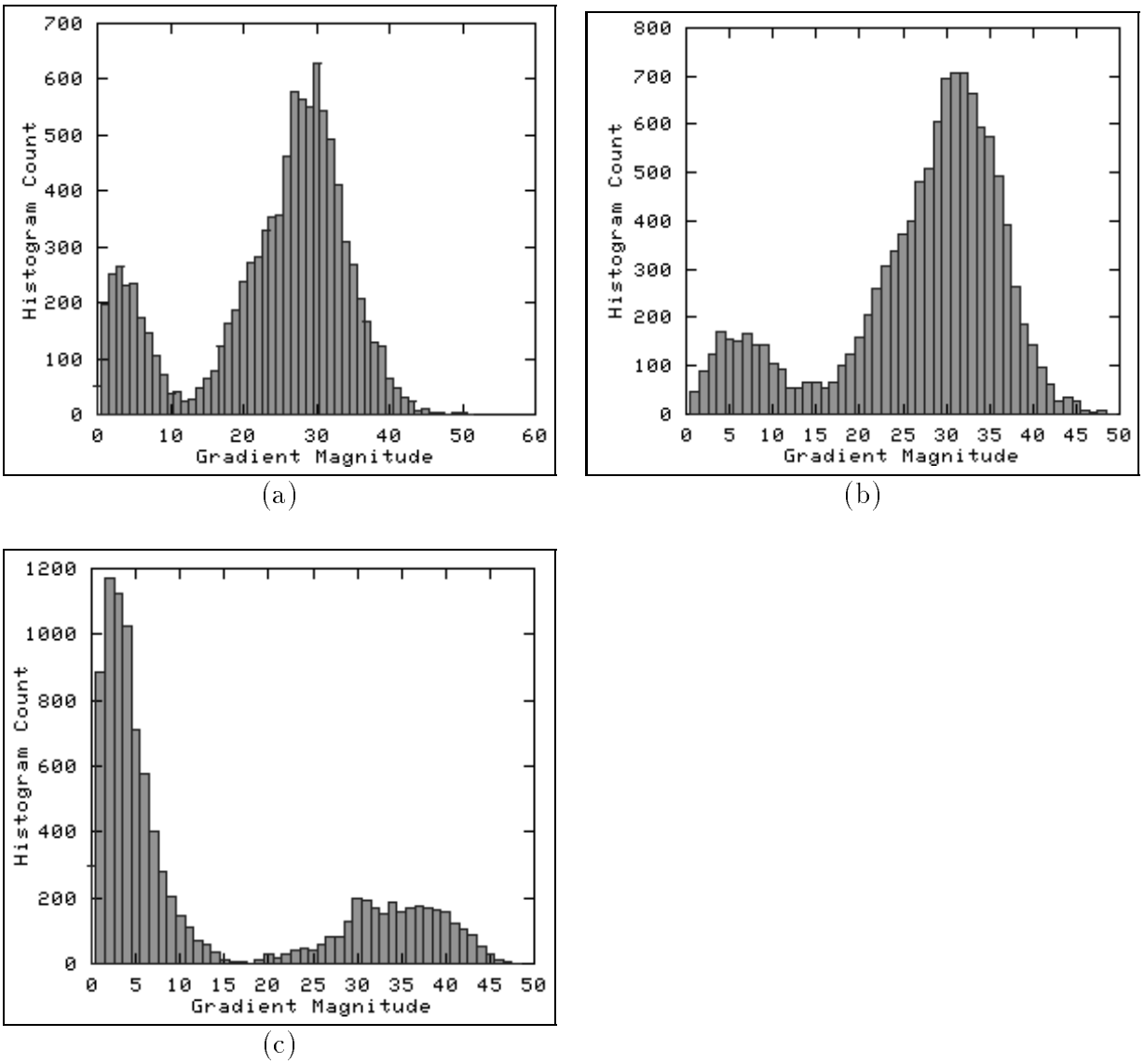


FIGURE 4.11. Example histograms of gradient magnitude in edges returned by modified Canny algorithm. (a) Siberian larch #1. (b) Siberian larch #2. (c) Ponderosa Pine. The sample in (c) has a large number of low contrast scratches, which accounts for the large counts at low gradient magnitude values. Note that despite the variation in counts for high versus low gradient magnitude values, all distributions are bimodal.

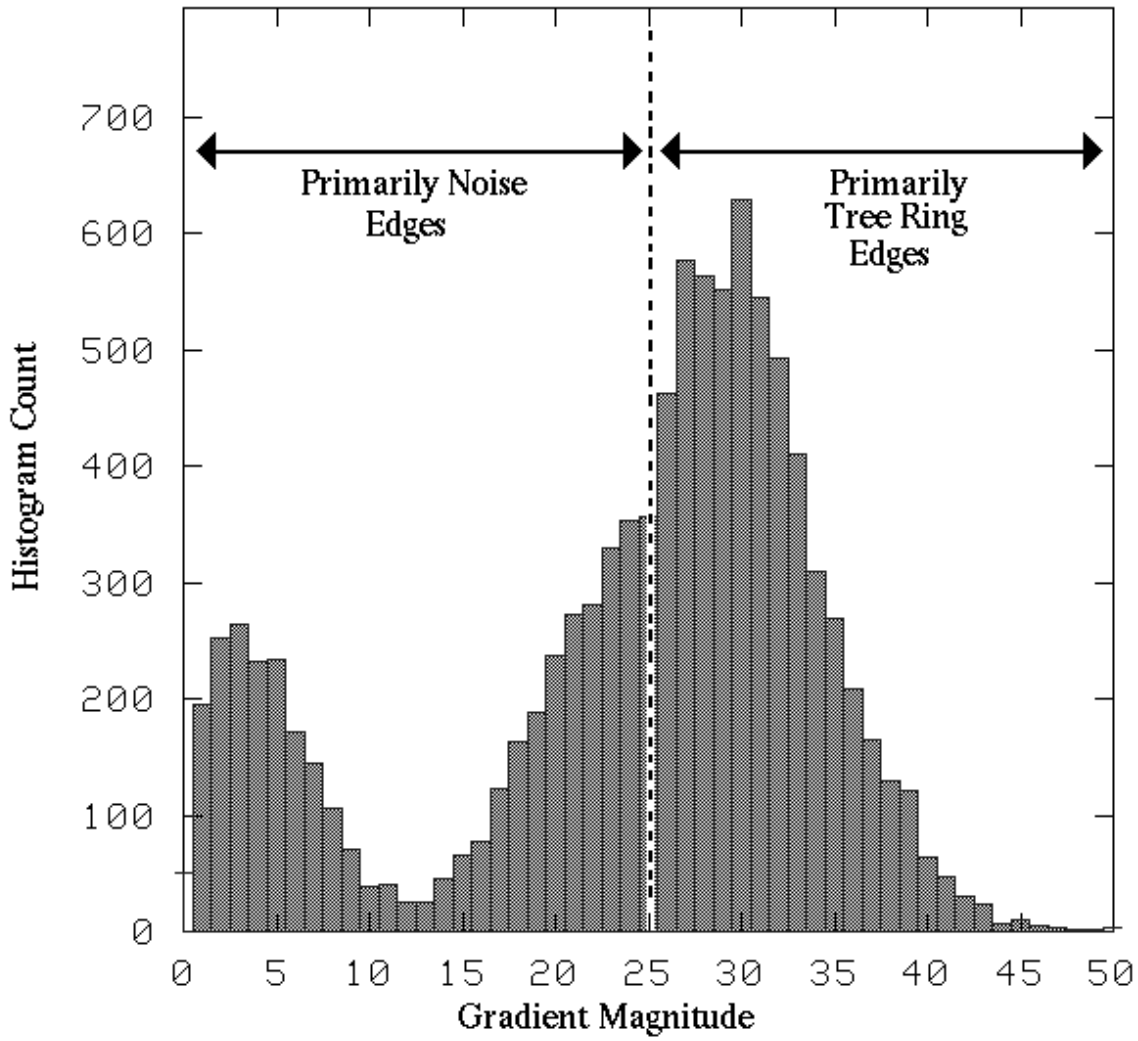


FIGURE 4.12. Threshold selection for gradient magnitude in edges returned by modified Canny algorithm.

ring edges that may be fragmented. In this case each fragment in the tree ring will be assigned its own label during connected components labeling. Visually, it is generally easy to determine which fragments belong to a given tree ring boundary. However, an intelligent fragment linking algorithm must be applied in order for a computer to link fragmented tree ring edges while discarding the remaining noise edges. In TREES, this is accomplished through a two stage linking approach of double threshold linking followed by linking optimized for tree rings.

4.3.2 Double Threshold Linking

The problems of degraded tree ring boundaries and the difficulty in eliminating all noise edges during the threshold stage may be addressed by a double threshold linking technique as described by Jain, et. al. [12]. Double threshold linking is accomplished by choosing a high threshold to remove noise edges while maintaining fragmented high contrast tree ring edges, and a low threshold that maintains full high contrast edge data as well as noise edge data. The entire edge map from the high threshold is considered to contain good edges, and any edges from the low threshold that are connected to the edges from the high threshold are also kept. This has the effect of filling in the "holes" in the fragmented high contrast edges where the gradient magnitude is below the threshold, while eliminating extraneous noise edge pixels. For the purposes of tree ring edge detection, the high threshold edge map should be the map returned after using an appropriate threshold as described above, which has been additionally reduced with a size filter to remove any fragments that are too small (and therefore have a high probability of being noise edges). The low threshold image should be the edge map from the modified Canny algorithm, with a zero threshold to maintain all edge data.

An example of double threshold linking is shown in Figure 4.13, including high and low threshold edge maps and the resulting edge map after double threshold linking.

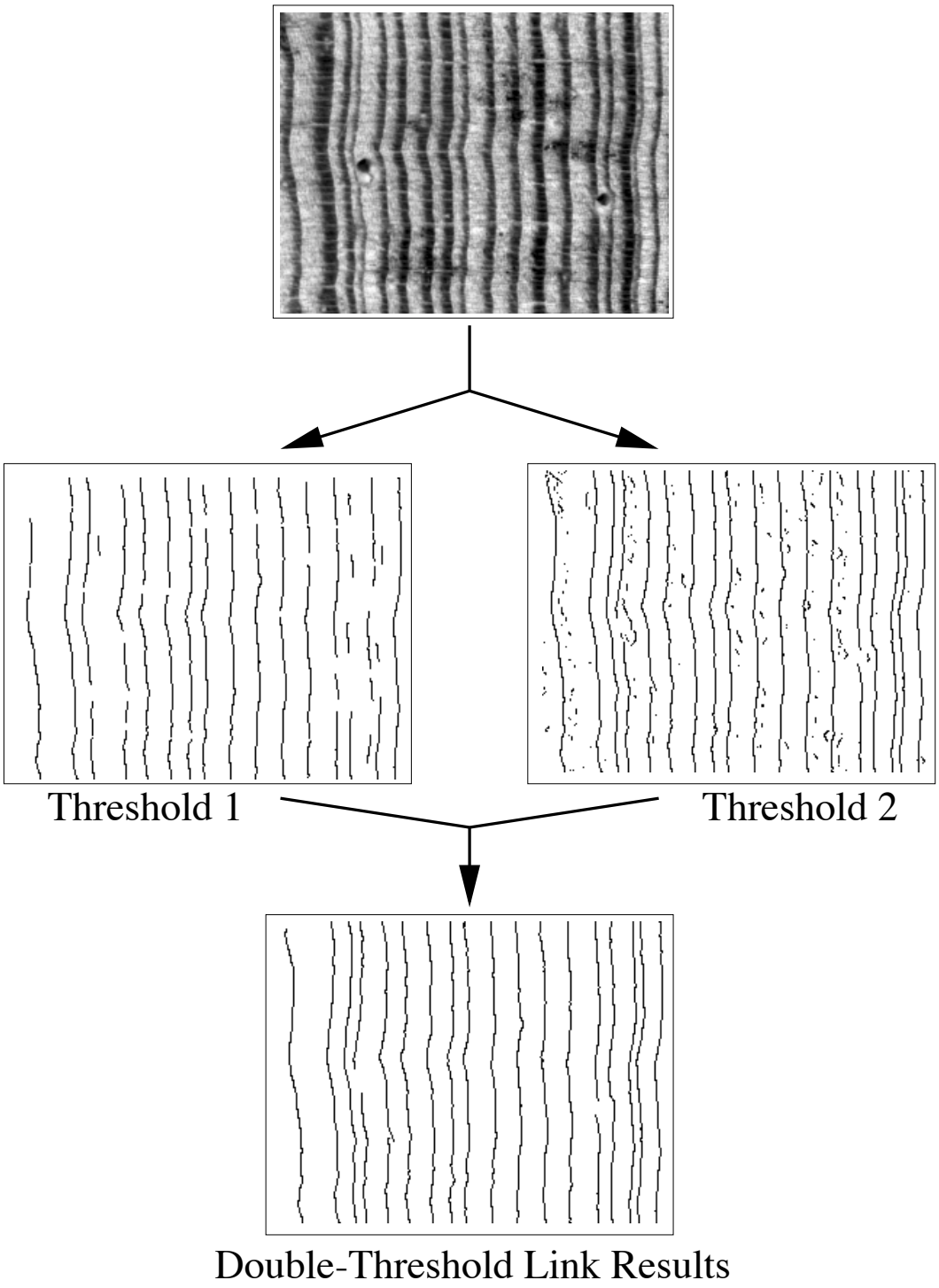


FIGURE 4.13. Double threshold linking.

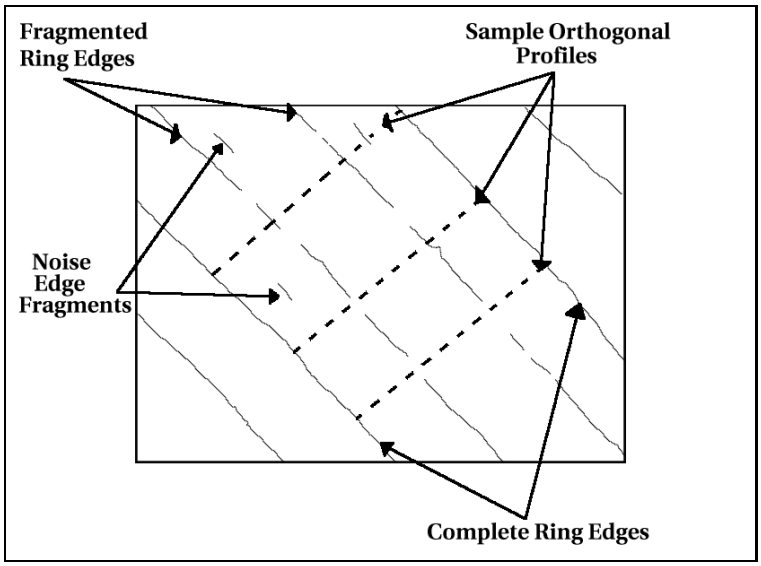
4.3.3 Second Linking Stage Based on Characteristics of Tree Rings

The double thresholding technique is not guaranteed to return all tree ring edges fully connected. A second postprocessing step that is capable of closing these gaps is based on the physical characteristics of tree rings, specifically the fact that the width of a tree ring relative to the widths of neighboring rings remains nearly constant along the length of the ring. This information makes it possible to decide whether a set of edge fragments belong to the same tree ring boundary.

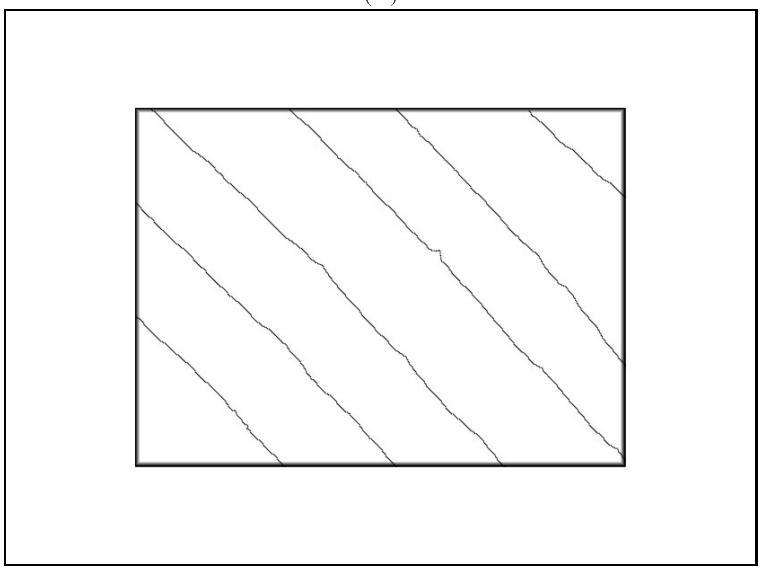
In order to implement this decision, it is necessary for several fully connected tree ring boundaries to be present after double threshold linking. In most cases, only a few rings with low contrast regions remain fragmented after the double threshold linking stage. Any fully connected tree ring boundaries are assumed to be accurate representations of tree ring boundary positions. Between each pair of complete rings, the edge map is searched for fragments.

Since most of the noise fragments were removed during double thresholding, the median number of fragments, n, located along each orthogonal profile between the two complete rings, as illustrated in Figure 4.14, is very likely equivalent to the number of tree rings that exist between the two known rings. It is then possible to compute the n most common positions of tree ring fragments, as a percentage of the distance between the two known rings. The possible locations are quantized into M values, equally spaced from 0% to 100%. M bins are used in a voting scheme to count the number of fragments that exist at each location. The n locations with the largest counts represent the most likely locations of tree ring boundaries. Any fragments that are located at a given position are labeled as belonging to the appropriate ring, while any fragments that are not near one of the positions are discarded as noise fragments. It is then possible to synthesize links between fragmented tree ring edges in order to fully connect them. These links are flagged so that they are not used in future analysis, such as when average tree ring widths are computed. Their purpose

is to simply produce a complete set of indexed fully-connected rings. Examples of linking applied to a tree ring edge map are shown in Figure 4.7 (d) and Figure 4.10 (c), (f), and (i).



(a)



(b)

FIGURE 4.14. Final step of tree ring fragment linking. (a) Tree ring edge fragments remaining after double threshold linking. (b) Fully connected tree ring boundaries after linking. In this example, the tree rings in the field of view are assumed to be aligned with the orientation of sector 3 from Figure 4.6. When examining the orthogonal profiles between the two completely connected rings, the most common fragment locations encountered will be at approximately 30% and 75% of the distance between them. This information can be used to link the two fragmented tree ring edges while eliminating noise fragments.

Chapter 5
MEASUREMENT AND ANALYSIS

After fully connected tree ring edges have been detected and sequentially labeled, it is possible to proceed with analysis of the tree ring sample. Although the core of the image processing algorithms used in TREES are designed to produce reliable, fully-connected tree ring boundaries, the acquisition of these edges is not the primary goal of the system. This is only the first step that enables further analysis of the features of the individual tree rings in a sample, which is of far more importance to dendrochronologists.

5.1 Tree Ring Widths

In dendrochronology, the most commonly measured features of tree ring samples are relative tree ring widths. Patterns of ring width variation are the primary feature of tree rings that make it possible to cross-date among different wood samples (from two different trees, for example). For the purposes of this system, tree ring widths are defined to be the average distance between each pair of ring edges. Assuming that the ring detection techniques described above are accurate, this definition of tree ring widths is superior to the manual measurement most commonly used in dendrochronology labs, where only a single line between two tree rings is used to measure the width. By averaging the width over the full length of each tree ring in an image, the measurements are less susceptible to small anomalies within tree rings and therefore more accurate.

5.1.1 Direction of Tree Ring Width Measurement

The direction of width measurement at each point along a ring is important because the rings may have curves and other variations in orientation. In general, a ring width is defined as the radial distance between two ring boundaries at a given point on a ring boundary. However, since the precise position of the tree pith may not be known, the radial distance is estimated by determining the distance between two ring boundaries along the normal vector at a given point on a ring boundary. The normal vector used here should be orthogonal to the low frequency curve of the tree ring rather than the high frequency curves in the form of small dips and bumps in the ring.

The gradient direction computed during the edge detection stage (section 4.1) provides a localized measurement of the direction of maximum grayscale change. In most cases, this corresponds to the direction of transition from the latewood of one tree ring to the earlywood of the next, which is orthogonal to the tree ring boundary at a given point along the boundary. Because this information is already available from a previous processing step, it might be considered a computationally inexpensive way of estimating the direction of the normal vectors for width measurement. However, the local gradient direction is extremely sensitive to small dips or bumps in the ring edge from features such as rays and sap ducts, and may misrepresent the orthogonal direction between two ring boundaries by ninety degrees or more. Thus gradient direction is not recommended for reliably estimating the orthogonal direction to tree ring boundaries.

Since the detected tree ring boundaries are already determined and considered to be a reliable representation of the tree ring boundaries, in most cases it is better to fit a polynomial to the tree ring edge to determine the normal direction at each point, rather than using the local gradient. Since tree rings are by definition circles of varying radii that are centered at the pith of a tree, the most accurate mathematical

model of a tree ring edge is of the form:

$$(x-a)^2 + (y-b)^2 = r^2. \tag{5.1}$$

However, due to the high resolution at which tree ring samples are imaged, an underlying premise of the algorithms that produce the tree ring boundaries in TREES is that the tree rings are approximately linear within a region of interest. This allows much simpler linear interpolation techniques to be used to model the tangent lines that represent the orientation at each point along a tree ring boundary. Note that the purpose of producing such models is not to replace the computed tree ring edges with models, but simply to determine the orthogonal direction at any point along the tree ring boundary.

When fitting a line to a particular point on a tree ring boundary, the number of edge pixels used must be large enough to suppress the effects of small twists and curves. However, the number of points must be small enough to maintain an accurate representation of the local orientation of the ring. At the magnification used by this system and for the samples tested to-date, twenty to thirty edge pixels are generally sufficient, with the point of interest at the center of the data set.

A least-squares fit algorithm is used to approximate the orientation of tree ring boundaries [12, 18]. This method involves minimizing the sum of squared perpendicular distances of each edge pixel in a segment from the line represented in polar form as:

$$\rho = x\ cos\ \theta + y\ sin\ \theta. \tag{5.2}$$

The first step used to compute the orientation of a segment of a tree ring boundary is to determine the center of the set of edge pixels, which can be substituted as a valid value for ρ in the equation of the line. If N edge pixels are included in the set and the line, y_i, and column, x_i, of each edge pixel, i, are known, the center of the set is

at:

$$\overline{x} = \frac{\sum_{i=1}^{N} x_i}{N}, \tag{5.3}$$

$$\overline{y} = \frac{\sum_{i=1}^{N} y_i}{N}. \tag{5.4}$$

As described by Jain, et. al., in [12], the angle of orientation of the segment of edge points that minimizes the sum of squared errors for the points in the segment can be computed as:

$$\theta = \frac{1}{2}\, tan^{-1}\, \left(\frac{b}{a - c}\right), \tag{5.5}$$

where the parameters a, b, and c are computed as:

$$a = \sum_{i=1}^{N} (x_i - \overline{x})^2, \tag{5.6}$$

$$b = 2 \sum_{i=1}^{N} (x_i - \overline{x})(y_i - \overline{y}), \tag{5.7}$$

$$c = \sum_{i=1}^{N} (y_i - \overline{y})^2. \tag{5.8}$$

Depending on the location of the pith and bark of the tree relative to a tree ring boundary, which is determined either during the data acquisition stage or during tree ring boundary detection, the orthogonal angle to the tree ring boundary is obtained by either adding or subtracting ninety degrees from the computed tangent angle, θ.

5.1.2 Measurement of Average Tree Ring Widths

Once the normal direction to a point on a ring edge has been determined, the distance between the current ring and the next ring is computed in this direction. The method used to compute this distance is to travel along the sector direction (Figure 4.6) closest to the normal angle at the current ring location until the next ring is reached. Pixel-by-pixel steps are then taken along the second ring, computing the angle between the starting point on the first ring and the current ending point on the second

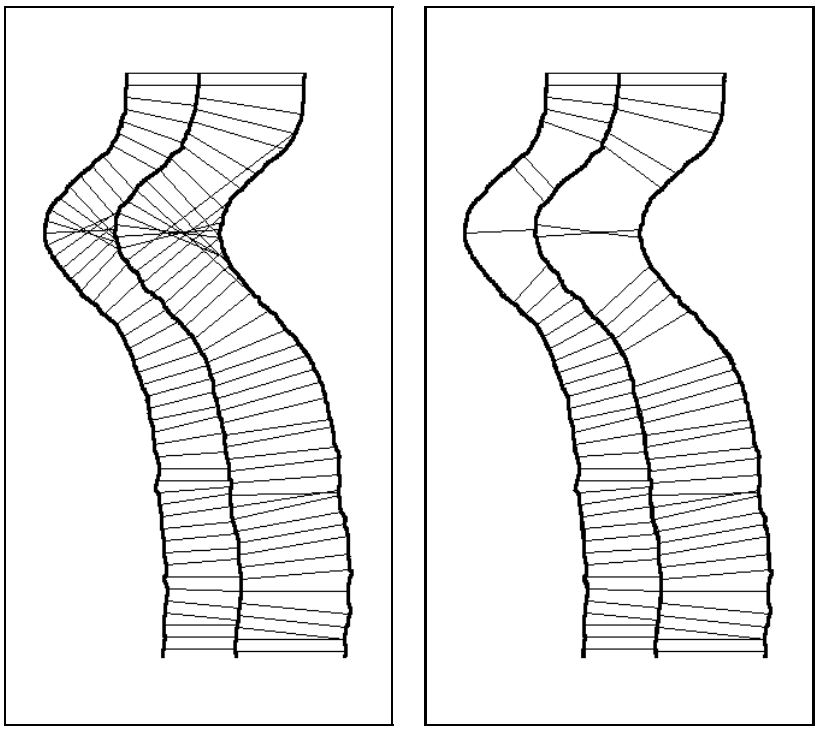


FIGURE 5.1. Example width measurements used for computing average ring widths. (a) No error checking. (b) Width measurements that remain after requiring the angle of the width measurement lines with respect to the left and right rings to match within ten degrees. Note that for illustration purposes, only 1/10 of the width measurement lines are shown.

ring. When the angle matches the angle of the normal vector, the two end points of the orthogonal line are known, and the width can be measured by computing the Euclidean distance. For error checking, the system checks that the orthogonal line is within ten degrees of being orthogonal to the second ring boundary at the ending point, as shown in Figure 5.1. If this is not the case, there is a good possibility that the width measurement at that point along the curve has a large error and should be discarded.

In order to determine the average width of a ring, the width at each point along the tree ring boundary should be measured, discarding any known erroneous measurements as described above. The average of these measurements is the average ring

width. *Absolute* ring widths are not as important in dendrochronology as *relative* ring widths, so this method is sufficient for defining the average width of a ring even if the width of the ring varies over the region of interest.

Computer generated tree ring width measurements are computed in pixels. Image analysis is the entire basis of the measurements obtained by the TREES system, and since pixels are the finest resolution units available in an image, they are the natural choice for width measurement units. A calibration between pixel widths and physical distances on the actual tree sample can be obtained by applying knowledge of the magnification of the system and the size of the CCD elements in the camera. For a system with a total magnification M and a CCD camera with square imaging elements of width w μm, the calibration between pixels and physical units is:

$$unit\ calibration = \left(\frac{1}{M}\right)\ \frac{w\ \mu\text{m}}{1\ \text{pixel}}. \tag{5.9}$$

The CCD camera used in the TREES system has square 6.8 μm elements. When used with a typical magnification of 0.693, the resulting calibration is 10.79 μm/pixel. This results in a typical 1mm wide tree ring having a measured width of around 93 pixels.

5.2 Additional Tree Ring Attributes

Tree ring widths are the feature that can most easily be extracted from tree rings via image analysis algorithms. However, other features such as the average grayscale profile between two ring edges, can be measured and recorded. Also, the analyst may attribute additional features for unusual rings, such as frost and micro rings. All of this information is useful when cross-dating between tree ring samples.

5.2.1 Average Grayscale Profiles of Tree Rings

After computing the average widths of each of the tree rings in a sample, it is possible to create an average grayscale profile of each tree ring. To generate such a grayscale

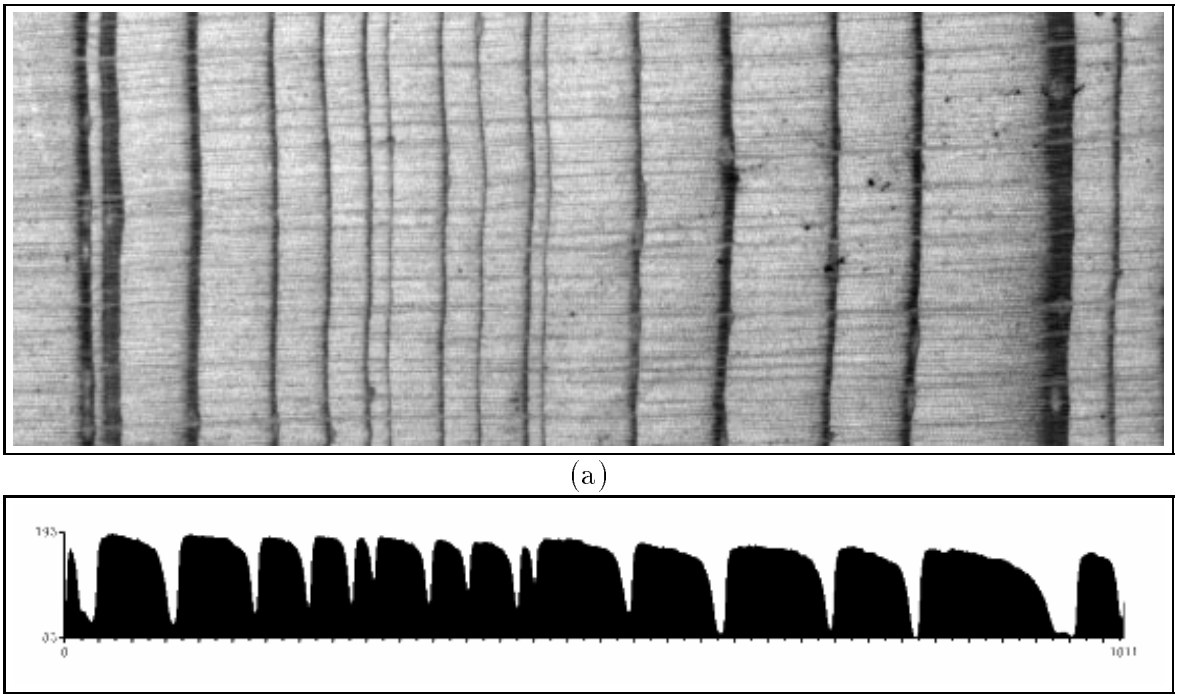
profile for each ring, a count array and a sum array whose lengths are the nearest integer of the average width of the ring in pixels, $\overline{W}$, are used to consolidate the grayscale values along each of the individual width lines that were used to compute the average width.

For each position, j, along a given individual width line i with length W_i, the corresponding position in the average width line of width $\overline{W}$ is computed as:

$$index_{avg}(i,j) = \left(\frac{j}{W_i}\right) \quad \overline{W}. \tag{5.10}$$

In other words, the absolute pixel position in the individual line is converted to a percentage of the individual width and then multiplied times the average width to find the position in the average width line. The grayscale value under the point j on the individual width line is added to the sum array at index $index_{avg}(i,j)$ and the count array value at the same index is incremented. After repeating this procedure for all points along each of the width lines used to compute the average width, each value in the sum array is divided by the corresponding value in the count array. The result is an array of length $\overline{W}$ containing the average grayscale profile of the ring. Figure 5.2 shows an example tree ring sample and the corresponding computed average grayscale profile.

One of the advantages of producing an average grayscale profile of a tree ring sample is that it allows internal characteristics of tree rings to be measured and analyzed. For example the internal earlywood to latewood boundary can be estimated as the location where the maximum rate of change from the high grayscale value earlywood to the low grayscale value latewood occurs in the average profile. This boundary can be used to divide each individual tree ring into its earlywood and latewood components. The widths of these intra-ring components can be used as cross-dating aids in addition to the full tree ring widths.



(a)

(b)

FIGURE 5.2. Average grayscale profile.

5.3 Pattern Matching and Cross-Dating

The primary goal of tree ring analysis in most cases is the ability to cross-date one sample to another. Cross-dating between two wood samples is based on the fact that one tree ring is grown for each year in the lifetime of most trees, and that the ring's properties are determined by environmental conditions at the time of growth. The first ring around the pith of a tree represents its first year of growth. The partial tree ring at the edge of the tree, just before the bark, represents either the current year in a live tree, or the last year of growth in a dead tree. Climatic changes tend to cause not only varying ring widths from one year to the next in a single tree, but also similar variation across different trees in a region. If the ring width patterns from one tree can be matched to the ring width patterns in another tree, there is a high probability that the tree rings in question were grown during the same years. This allows a tree

or log that has been dead for an unknown number of years to be cross-dated with a live tree to determine its age. Pairs of trees from the same species are obviously desirable, but in principal, cross-dating between species may be possible with the aid of skeleton plots.

5.3.1 Skeleton Plots

In the Douglass method of dendrochronology (Section 1.3.1), cross-dating involves the use of "skeleton plots," which are a simple way of representing tree rings in a compact, normalized form. Skeleton plots are typically drawn on graph paper, and each vertical line on the graph paper corresponds to a single year, representing a single tree ring in a sample. Skeleton plot line patterns are drawn on the vertical lines of the graph paper based on the relative widths of each tree ring and its neighbors. If a tree ring has a similar width to its neighboring tree rings, no skeleton plot line is drawn since the tree ring does not stand out. However, if a significant change exists because the width of a particular tree ring is much narrower than the widths of its neighbors, a skeleton plot line is included on the plot. The length of the line indicates how much smaller a tree ring is than its neighbors, with the smallest rings having the longest skeleton plot lines. In addition, wide rings, micro rings, frost rings, and other outstanding tree ring features can be symbolically tagged on a skeleton plot.

Cropper describes a method of numerically generating skeleton plots [5]. Cropper's method involves computing the running mean, μ_i, and standard deviation, σ_i, for a set of tree ring widths. A critical level, σ_c, is chosen in order to determine the relative narrowness of tree rings. A tree ring with width W_i will have a skeleton plot line if

$$W_i < \mu_i - \sigma_c \sigma_i, \tag{5.11}$$

or is marked as a particularly large ring if

$$W_i > \mu_i + \sigma_c \sigma_i. \tag{5.12}$$

The lengths of lines drawn in Cropper's computer generated skeleton plots are determined by setting the smallest ring's line to the maximum length for a particular plot scale, and scaling each of the remaining skeleton plot lines by the same scale factor.

Skeleton plots used in the TREES system are adapted from a modified Java version of Cropper's algorithm, implemented by Paul Sheppard of the Laboratory of Tree Ring Research at the University of Arizona [23]. This algorithm requires tree ring widths to be filtered such that low frequency trend is removed. This can be accomplished by fitting a cubic smoothing spline to the series to model the low frequency characteristics of the data, and dividing the series by the computed model [10]. The resulting filtered data set, which is referred to as a set of index values, I_i, will have a mean value of 1.0. The minimum and maximum index values of the filtered series are I_{min} and I_{max}, respectively. The skeleton plotting algorithm requires an absolute cutoff value, C_{ab}, and a first difference cutoff value, C_{diff}, to be specified in order to trigger the existence of skeleton plot lines. The lengths of skeleton plot lines are computed on a specified scale (a scale of ten is used for this discussion), with a default length of zero. A skeleton plot line will have a non-zero length if the following absolute value check is triggered:

$$I_i < 1.0 - \frac{1.0 - I_{min}}{C_{ab}}, \tag{5.13}$$

where $1.0 < C_{ab} < 10.0$. The corresponding length, L_i, of a skeleton plot line created by the absolute value trigger is

$$L_i = 10.0 - \left(\frac{I_i}{1.0 - \frac{1.0 - I_{min}}{C_{ab}}} \right) \times 10.0. \tag{5.14}$$

Likewise, the first difference trigger

$$I_i - I_{i-1} < \frac{I_{min} - I_{max}}{C_{diff}}, \tag{5.15}$$

where $1.0 < C_{diff} < 10.0$, results in a skeleton plot line of length

$$L_i = 10.0 - \left(\frac{(I_i - I_{i-1}) - (I_{min} - I_{max})}{\frac{I_{min} - I_{max}}{C_{diff}} - (I_{min} - I_{max})} \right) \times 10.0. \tag{5.16}$$

Figure 2.1 illustrates an example graphical user interface from TREES which displays the detected tree ring boundaries for a sample, and the corresponding computed skeleton plot.

Skeleton plots can be visually compared to one another by an analyst to determine matching ring width patterns. The Douglass approach of tree ring dating requires that the original tree sample remain available for reference throughout the cross-dating process. This is due to the fact that anomalies in the growth process of trees may cause very small micro rings to be present in a particular tree sample. Thus, even if image analysis algorithms produce perfect tree ring boundaries based on an input sample mosaic image, some tree ring boundaries may be missing or may not actually belong to true annual growth rings in the tree. For this reason, the cross-dating procedure does not simply consist of finding the "best fit" between two tree samples. If inconsistencies are found between two or more skeleton plots, they must be resolved by returning to the original tree sample for further analysis.

This basic cross-dating feature has been integrated into the TREES system. Skeleton plot lines are linked to the corresponding tree ring boundaries, so it is possible to return to the sample image for reference while analyzing a skeleton plot. As long as the wood sample is not removed from the positioning table after the mosaic is created, it is also possible to command the positioning table to center a particular region of the sample under the microscope by pointing and clicking the mouse on the corresponding area in the mosaic image that is displayed in the GUI. This allows an analyst to take a closer look at a particular tree ring if discrepancies arise when comparing skeleton plots.

In the current version of TREES, cross dating must be performed manually by the analyst. In the future, correlation algorithms could be implemented to allow the system to provide the analyst with a first best guess at a cross-dating solution, which the analyst will be able to accept or reject.

Chapter 6
SUMMARY AND CONCLUSIONS

6.1 Summary

Tree ring analysis has many applications in fields ranging from archaeology to climatology. Unfortunately, a large percentage of the research time of experienced dendrochronologists is spent manually analyzing each tree ring sample. This thesis describes a system that implements image processing and computer vision techniques to increase the efficiency of tree ring analysis.

The TREES system consists of a CCD camera connected to a microscope that images tree ring samples placed on an x-y positioning table under the microscope. The position of the positioning table, the focus setting of the microscope, and the CCD camera are all controlled by a Sun Ultra 1 workstation.

When an analyst chooses a starting and ending point of a region of interest on a tree ring sample, the system automatically captures a sequence of carefully focused images that covers the specified region of interest. Each frame is adjusted for nonuniformity of illumination, and precisely registered to its neighbors, both spatially and radiometrically. Based on the registration results, a single continuous mosaic image is constructed from the individual frames.

Chapter 4 describes the edge detection techniques used to identify tree ring boundaries in an image mosaic. Gradient magnitude computations, combined with thresholding and median filtering to reduce noise, are capable of identifying the high contrast edges that exist at tree ring boundaries. However, the boundaries lack precision and are severely degraded during noise removal attempts. A superior method is to use the Canny edge detector, which applies knowledge of gradient direction in addition to magnitude. This method produces precise, one-pixel wide boundaries but has the

unwanted characteristic of sometimes connecting adjacent tree rings.

A modified version of the Canny edge detection algorithm produces precise boundaries, while suppressing inter-ring connections. This is made possible by using global direction assumptions within a region of interest, rather than examining the local gradient direction at each pixel location. The disadvantage of this modified Canny-based algorithm is that it does not guarantee that fully connected tree ring boundaries will be created. Double threshold linking, followed by an intelligent linking algorithm that is optimized to common characteristics of tree rings, are used to produce complete representations of tree ring boundaries while suppressing noise features. The second linking algorithm assumes that the width of a tree ring relative to the widths of neighboring rings remains nearly constant along the length of the ring.

After the detection of tree ring boundaries, the average width of each tree ring in the sample is computed. Average grayscale profiles allow the internal characteristics of individual tree rings to be analyzed. This information is used to produce skeleton plots which allow an analyst to cross-date one tree ring sample to another.

6.2 Conclusions

This thesis presents computer vision approaches which have proven successful at accurately analyzing typical conifer tree ring samples in a semi-automated manner. The research described here indicates that image analysis may be capable of increasing the efficiency of dendrochronology research, while preserving the feature of human intervention. By allowing an analyst to concentrate on other research work while the TREES system captures and analyzes a sample, much of the tedium of tree ring analysis is removed, potentially increasing the number of samples that can be analyzed.

The accuracy of the edge detection performed by TREES has primarily been verified via visual analysis. By overlaying edge maps, such as those of Figure 4.10, on

the original tree ring sample images, it has been verified that the modified Canny algorithm produces results that are visually within one pixel of the latewood - earlywood boundaries. These results are at least as accurate as the manual techniques that are currently used to determine tree ring boundaries. Assuming that tree ring boundaries have been correctly identified by the system, the width measurement technique used in TREES is superior to manual width measurements due to the fact that the system's measurements are produced by averaging tree ring widths across the entire region of interest. This technique is less sensitive to noise than the manual measurement approach, which uses only a single profile to measure the width of each ring.

The system has to-date been tested on approximately twenty relatively typical samples from four conifer species, including ponderosa pine, Douglas fir, Siberian larch, and juniper. The results of these tests were similar to the examples of Figure 4.10 as long as no cracks or other major anomalies were present within the region of interest captured and analyzed. All testing to-date has been performed by members of the TREES project team, who are familiar with the algorithms and techniques used in the software. A version of the TREES system is currently being prepared for use by members of the dendrochronology community who are not familiar with the details of the operation of the system. Tests performed by such analysts will test the robustness of the system and aid in determining future enhancements that will hopefully make the system routinely usable for dendrochronology research.

6.3 Recommendations for Future Contributions

While the results presented in this thesis are very promising, much work remains to be done in order to increase the efficiency, reliability, and diversity of the TREES system. From the standpoint of software engineering, the efficiency and organization of the TREES software can be improved by modifying the structure of the SADIE

image processing library. In order to make TREES more robust across diverse tree ring samples, the edge detection algorithms could be made more intelligent. Also, to increase the efficiency of the cross-dating portion of the system, semi-automated correlation and pattern matching algorithms could be integrated into TREES. To increase flexibility when analyzing multiple samples, techniques should be developed to allow an analyst to return to a previous sample after it has been removed from the positioning table. Finally, while TREES has been primarily designed to aid in the measurement of tree ring widths for cross-dating purposes, enhancements could be made to allow reliable analysis of other features of tree rings to aid in climatology and other forms of research.

6.3.1 Further Modifications to SADIE

In Section 2.1.2, the 32,767 width / height limitation of SADIE images was described. This imposes a maximum of approximately 32cm of a tree ring sample to be imaged at typical resolution settings. For most samples this will be sufficient. However, in the future it will likely become desirable to have the ability to use TREES for the analysis of larger tree samples. In order to make this possible, the maximum height or width of an image can be increased by changing the short integer variables to unsigned short integers or normal integers. Unfortunately, in order to implement this change each routine in the SADIE library would need to be changed due to the fact that local variables in the routines that are used to loop through image data are currently implemented as signed short integers.

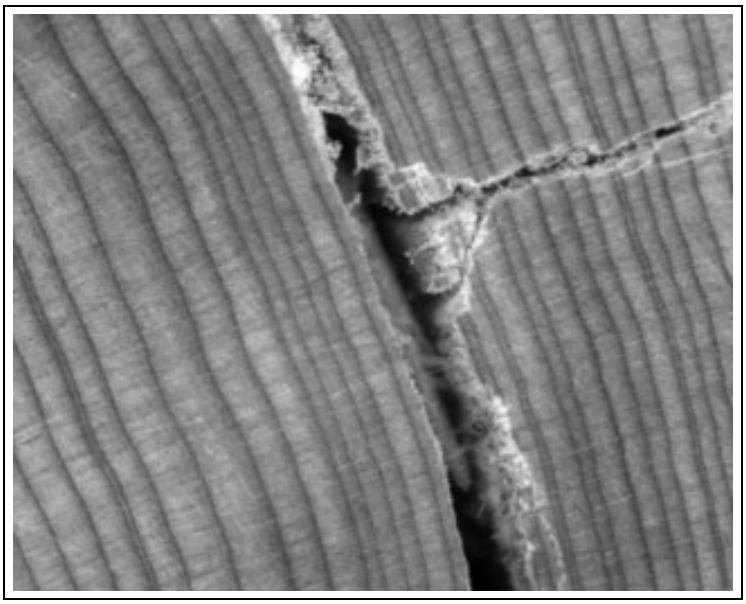
In order to accommodate varying levels of image data precision, SADIE_BYTE and SADIE_SHORT data types were added to TREES in addition to the standard 32-bit IMAGE format of the SADIE library, as described in Section 2.1.3. Currently, a separate version of any routine that uses these data types must be created for each individual data type. A future enhancement to the SADIE library would be

to integrate several precision types into a single SADIE image format. A tag could be added to the image data structure shown in Table 2.1 to indicate which format an image is in. This would require all SADIE routines to check the precision of the image data before processing an image.

6.3.2 Development of More Robust Edge Detection Techniques

The system has to-date been tested primarily on relatively well-behaved conifer samples. More complex tree ring patterns such as those of hardwoods, as well as cracks and other troublesome wood features, will test the robustness of the present design and will likely motivate modifications. Two particularly difficult examples are illustrated in Figure 6.1. Assumptions made in the second tree ring edge linking algorithm (Section 4.3.3) cause the tree ring edge detection techniques currently used in TREES to fail in both of these difficult cases. In the future, more intelligent tree ring detection algorithms should be developed to supplement the routines that are already in place in the system. As an example, an algorithm might be added in the future to analyze the "roughness" of a tree ring boundary. Most tree ring boundaries are relatively smooth, while cracks tend to be very rough and bumpy. This type of information could be useful in distinguishing true tree ring edges from crack edges.

In keeping with the design philosophy of the TREES system, it may prove more productive to allow the dendrochronologist to manually handle these unusual situations. Rather than attempting to automatically solve all difficulties encountered, which may require an unacceptable level of computation, analyst intervention should be allowed at critical stages. For example, the analyst should be able to manually connect tree ring fragments across cracks or guide the search for very low contrast edges.



(a)

(b)

FIGURE 6.1. Example difficult juniper tree samples. (a) Sample with large cracks, both parallel and orthogonal to tree ring edges. (b) Sample with rings wedged together due to a branch scar.

6.3.3 Automated Cross-Dating

The current version of TREES requires all cross-dating to be performed manually by the analyst. In the future, a major enhancement should be the integration of correlation and other pattern matching techniques to aid the analyst by computing a best first guess at a match. Techniques for computerized cross-dating are described by Baillie and Pilcher [1] and Munro [14].

In addition, quality control algorithms for verifying cross dating choices between samples exist at the Laboratory of Tree Ring Research in the COFECHA program [10, 11]. The functionality of this type of algorithm could be applied to aid in the identification of locally false and/or missing rings, as well as to verify the cross-dating results determined by algorithms or the analyst.

6.3.4 Multi-Session Memory

In the Douglass approach of tree ring dating, one of the most important requirements for reliable dating is the ability to return to the original wood sample in order to reconcile problems. As long as a sample is not removed from the stage after capturing a sequence of images and generating a mosaic, the TREES system is capable of repositioning the stage so that any region of the sample contained in the mosaic can be centered under the microscope for analysis. However, if a problem with a sample does not become evident until after the sample has been removed from the positioning table, there is no easy way to return to the wood via the system.

There are two possible techniques that might be applied in the future to eliminate this shortcoming. The first method involves physically mounting a sample on the positioning table by drilling a set of screws into the wood. If the sample is removed from the stage, it could be returned at a later date in its original orientation by remounting it with the same screws inserted into the same holes in the wood. In addition, an identifier must be marked at a unique position on the surface of the

wood to allow the system to resynchronize the position of the sample under the microscope with the mosaic image loaded into the GUI.

If no repeatable method of mounting the sample on the stage is practical, a software approach could be used to resynchronize the sample with the pre-generated mosaic. For this purpose, an identifier that is non-ambiguous in both position and orientation must be placed on the surface of the wood before beginning a new session. A one-headed arrow would suffice for this purpose. During data acquisition, an image of this identifier must be captured and registered with the system. When the sample is returned to the stage later, the analyst should be prompted to center the identifier under the camera. An image of the new orientation of the identifier could be captured and compared to the original orientation that was used in the creation of the mosaic. Rather than requiring precise rotation of the sample to match its original orientation, positioning table coordinates could be computed and converted between the current orientation and the previous orientation.

In either of the cases described above, the magnification of the system must be recorded at data acquisition time, and the identical magnification must be set when the tree sample is returned to the stage.

6.3.5 TREES as a Substitute for X-Ray Densitometry

TREES has been initially designed for the sole purpose of measuring tree ring widths, and is not optimized for accurately estimating X-ray densitometry data. However, in the future it would be wise to integrate the careful calibration steps used in the Reflected-Light Image Analysis System described in Section 1.4.3 into TREES. This would allow average grayscale profiles to be used to estimate wood density data that otherwise must be obtained by tedious X-ray densitometry techniques, and would create a more versatile TREES system.

References

[1] M. G. L. Baillie and J. R. Pilcher, "A simple cross-dating program for tree-ring research." *Tree Ring Bulletin*, 33, 7-14, 1973.

[2] J. F. Canny, "A Computational Approach to Edge Detection." *IEEE Trans. Pattern Analysis and Machine Intelligence*, 8(6), 679-698, 1986.

[3] K. R. Castleman, *Digital Image Processing*. Prentice-Hall, Inc., Englewood Cliffs, NJ, 1996.

[4] W. S. Conner, R. A. Schowengerdt, M. Munro, and M. K. Hughes, "Design of a Computer Vision Based Tree Ring Dating System", *Proceedings of the 1998 IEEE Southwest Symposium on Image Analysis and Interpretation*, 256-261, 1998.

[5] J. P. Cropper, "Tree-Ring Skeleton Plotting by Computer." *Tree-Ring Bulletin*, 39, 47-54, 1979.

[6] A. E. Douglass, "Tree Rings and Chronology." *University of Arizona Bulletin*, 8(4), 1957.

[7] R. Guay, R. Gagnon, and H. Morin, "MacDENDRO, a new automatic and interactive tree-ring measurement system based on image processing." *Tree Rings and Environment: Proceedings of the International Symposium, Ystad, South Sweden, 3-9 September, 1990*, Lund University, Department of Quaternary Geology. *Lundqua Report* 34, 128-131, 1990.

[8] R. Guay, *WinDENDRO V. 6.0 User's Guide*, Régent Instruments, Inc, Quebec, Canada, 1995.

[9] M. Harrison, M. McLennan, *Effective Tcl/Tk Programming*. Addison-Wesley, Reading, MA, 1998.

[10] Holmes, R. L., "Computer-Assisted Quality Control in Tree-Ring Dating and Measurement." *Tree-Ring Bulletin*, 43, 69-78, 1983.

[11] Holmes, R. L., *Dendrochronology Program Library User's Manual*. Laboratory of Tree-Ring Research, University of Arizona, Tucson, AZ, 1994.

[12] R. Jain, R. Kasturi, and B. G. Schunck, *Machine Vision*. McGraw-Hill, Inc., New York, NY, 1995.

[13] G. Mehldau and R.A. Schowengerdt, "MacSADIE 1.2 User's Manual," University of Arizona, Digital Image Analysis Lab, Technical Report No. 90-4, December 1990.

[14] M. A. R. Munro, "An Improved Algorithm for Crossdating Tree-Ring Series." *Tree-Ring Bulletin*, 44, 17-27, 1984.

[15] A. V. Oppenheim and R. W. Schafer, *Discrete-Time Signal Processing.* Prentice Hall, Englewood Cliffs, NJ, 1989.

[16] J. Ousterhout, *Tcl and the Tk Toolkit.* Addison-Wesley, 1994.

[17] A. Papoulis, *Probability, Random Variables, and Stochastic Processes.* Third Edition. McGraw-Hill, Inc., New York, NY, 1991.

[18] W. H. Press, S. A. Teukolsky, W. T. Vetterling, B. P. Flannery, *Numerical Recipes in C.* Second Edition. Cambridge University Press, 1992.

[19] J. Schou and E. Rytter, "Dendrochronological dating using scanning and imageprocessing." *Tree Rings and Environment: Proceedings of the International Symposium, Ystad, South Sweden, 3-9 September, 1990*, Lund University, Department of Quaternary Geology. *Lundqua Report* 34, 286-287, 1990.

[20] R. A. Schowengerdt, *Remote Sensing: Models and Methods for Image Processing.* Second Edition. Academic Press, San Diego, CA, 1997.

[21] R.A. Schowengerdt and G. Mehldau, "Engineering a Scientific Image Processing Toolbox for the Macintosh II," *International Journal of Remote Sensing*, Vol. 14, No. 4, pp. 669-683, 1993.

[22] R.A. Schowengerdt and G. Mehldau, "An Image Processing Toolbox for the Macintosh II," *Proc. of the Conference on Image Processing and the Impact of New Technologies*, Canberra, Australia, pp. 155-157, Dec. 1998.

[23] P. R. Sheppard, personal communication, 1999.

[24] P. R. Sheppard, L. J. Graumlich, "A Reflected-Light Video Imaging System for Tree-Ring Analysis of Conifers." *Tree Rings, Environment and Humanity, Proceedings of the International Conference*, Editors J. S. Dean, D. M. Meko and T. W. Swetnam. *RADIOCARBON*, pp. 879-889, 1996.

[25] P. R. Sheppard, L. J. Graumlich, L. E. Conkey, "Reflected-Light Image Analysis of Conifer Tree Rings for Reconstructing Climate." *The Holocene*, 6:62-68, 1996.

[26] P. R. Sheppard, *Reflected-Light Image Analysis of Conifer Tree Rings for Dendrochronological Research*, Dissertation, University of Arizona, 1995.

[27] M. A. Stokes, and T. L. Smiley, *An Introduction to Tree-Ring Dating.* University of Chicago Press, Chicago, IL, 1968.

[28] R. D. Thetford, R. D. D'Arrigo, and G. C. Jacoby, "An Image Analysis System for Determining Densitometric and Ring-Width Time Series," *Canadian Journal of Forest Research*, 21:1544-1549, 1991.

[29] S. E. Umbaugh, *Computer Vision and Image Processing: A Practical Approach Using CVIPtools*. Prentice Hall PTR, Upper Saddle River, NJ, 1998.

[30] B. B. Welch, *Practical Programming in Tcl and Tk*. Second Edition. Prentice Hall PTR, Upper Saddle River, NJ, 1997.

[31] P. D. Welch, "The Use of Fast Fourier Transform for the Estimation of Power Spectra: A Method Based on Time Averaging Over Short, Modified Periodograms." *IEEE Transactions on Audio and Electroacoustics*, vol. AU-15, 70-73, 1967.