

Portland State University

PDXScholar

Dissertations and Theses

Dissertations and Theses

10-25-1996

Adaptive Color Correlation of Knots in Wood Images and Weighted-value Product Selection Methods in a Machine Vision System

John Robert Goulding
Portland State University

Follow this and additional works at: https://pdxscholar.library.pdx.edu/open_access_etds



Part of the [Electrical and Electronics Commons](#)

Let us know how access to this document benefits you.

Recommended Citation

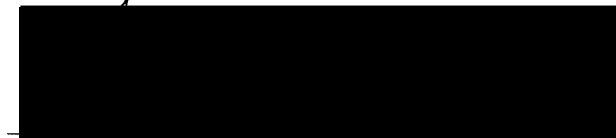
Goulding, John Robert, "Adaptive Color Correlation of Knots in Wood Images and Weighted-value Product Selection Methods in a Machine Vision System" (1996). *Dissertations and Theses*. Paper 5189.
<https://doi.org/10.15760/etd.7065>

This Thesis is brought to you for free and open access. It has been accepted for inclusion in Dissertations and Theses by an authorized administrator of PDXScholar. Please contact us if we can make this document more accessible: pdxscholar@pdx.edu.

THESIS APPROVAL

The abstract and thesis of John Robert Goulding for the Master of Science in Electrical Engineering were presented October 25, 1996, and accepted by the thesis committee and the department.

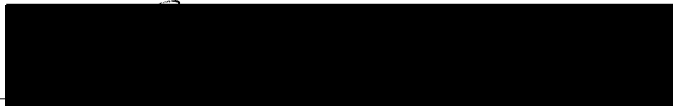
COMMITTEE APPROVALS:


George G. Lendaris, Chair


Robert W. Daasch



Tim R. Anderson
Representative of the Office of Graduate Studies

DEPARTMENT APPROVAL:


Rolf Schaumann, Chair
Department of Electrical Engineering

ACCEPTED FOR PORTLAND STATE UNIVERSITY BY THE LIBRARY

by

 on 2 December 1996

ABSTRACT

An abstract of the thesis of John Robert Goulding for the Master of Science in Electrical Engineering presented October 25, 1996.

Title: Adaptive Color Correlation of Knots in Wood Images and Weighted-Value Product Selection Methods in A Machine Vision System

The biggest obstacle to robust color image processing of wood is in developing a color model that represents all possible defect colors. When the color model is too general or too specific, defect recognition fails because too many or too few non-defect pixels match the model, respectively. Because a color image of wood contains far more clear and clear-grain colored pixels than grain-knot and knot colored pixels, it is beneficial to first statistically identify and remove the clear and clear-grain colors and to use the accumulated data to simultaneously enhance and normalize the remaining grain-knot and knot colored pixels. This process is here called adaptive color correlation. The normal image processing strategy is to search and test for defect features directly. The strategy proposed and developed here is to instead classify all wood pixels containing non-defect colors first, and then identify defect features. Once non-defect features are removed from an image, the task of finding candidate defects becomes easier and faster.

This improvement is realized in a sigmoid-shaped color correlation implemented as an adaptive look-up table.

As wood has become more expensive relative to manufacturing costs, more efficient methods of maximizing the recovery of clear wood in every board are sought. Optimization, in the present context, is a broad term for selecting products that are made from wood boards so the value of products produced is maximized for a given production requirement. Wood contains random defects which prohibit the production of some products. The normal optimization strategy is to mathematically change the value of under/over-produced products directly. The strategy proposed and developed here is to instead separate optimization into two steps: 1) determine all possible product solutions for a board; and 2) select the single best solution that satisfies value and production goals. Maximum utilization of clear wood is achieved because the solution is “frozen” before mathematically changing the value of products. Recovering long-lengths of clear wood is achieved because various length-based valuation strategies may be implemented as post-solution processes. Separating the product selection process from the solution generation process is shown by this work (simulation) to maximize value recovery.

ADAPTIVE COLOR CORRELATION OF KNOTS IN WOOD IMAGES
AND WEIGHTED-VALUE PRODUCT SELECTION METHODS
IN A MACHINE VISION SYSTEM

by

JOHN ROBERT GOULDING

A thesis submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE
in
ELECTRICAL ENGINEERING

Portland State University
1996

ABSTRACT

An abstract of the thesis of John Robert Goulding for the Master of Science in Electrical Engineering presented October 25, 1996.

Title: Adaptive Color Correlation of Knots in Wood Images and Weighted-Value Product Selection Methods in A Machine Vision System

The biggest obstacle to robust color image processing of wood is in developing a color model that represents all possible defect colors. When the color model is too general or too specific, defect recognition fails because too many or too few non-defect pixels match the model, respectively. Because a color image of wood contains far more clear and clear-grain colored pixels than grain-knot and knot colored pixels, it is beneficial to first statistically identify and remove the clear and clear-grain colors and to use the accumulated data to simultaneously enhance and normalize the remaining grain-knot and knot colored pixels. This process is here called adaptive color correlation. The normal image processing strategy is to search and test for defect features directly. The strategy proposed and developed here is to instead classify all wood pixels containing non-defect colors first, and then identify defect features. Once non-defect features are removed from an image, the task of finding candidate defects becomes easier and faster.

This improvement is realized in a sigmoid-shaped color correlation implemented as an adaptive look-up table.

As wood has become more expensive relative to manufacturing costs, more efficient methods of maximizing the recovery of clear wood in every board are sought. Optimization, in the present context, is a broad term for selecting products that are made from wood boards so the value of products produced is maximized for a given production requirement. Wood contains random defects which prohibit the production of some products. The normal optimization strategy is to mathematically change the value of under/over-produced products directly. The strategy proposed and developed here is to instead separate optimization into two steps: 1) determine all possible product solutions for a board; and 2) select the single best solution that satisfies value and production goals. Maximum utilization of clear wood is achieved because the solution is “frozen” before mathematically changing the value of products. Recovering long-lengths of clear wood is achieved because various length-based valuation strategies may be implemented as post-solution processes. Separating the product selection process from the solution generation process is shown by this work (simulation) to maximize value recovery.

ACKNOWLEDGEMENTS

The investigation and development of this thesis was carried out as a business research activity for Techné Systems, Inc., of Albany, Oregon. Although the immediate business application of this Machine Vision System is no longer pressing, it is fortunately broad enough in scope to have general scientific significance. Implementation of this technology continues in a line-scan based crosscut saw project.

The author is sincerely grateful to Dr. George G. Lendaris, a trusted mentor for the past nine years. His many excellent lectures on neural network technology and other discussions of cognitive processes led this author to abandon the conventional absolute color model image processing strategy and peruse a more brain-like adaptive strategy. His teachings also inspired this author's intellectual curiosity to reveal many otherwise obscure features of the problem. Finally, it was Dr. Lendaris' encouragement that prompted the use of this project as a thesis subject.

Thanks also goes to Mark Perkins who taught this author about grading and cutting wood through an intensive two-week training program as a crosscut and rip saw operator, and who, after every day of working in a wood manufacturing plant, asked this author if he "still had all 10 fingers." Acknowledgment must also be made to the friendly discussions and criticisms always encouraged in the Rip & Cut department by Andy Martisak as well as from his staff and the ripsaw operators. In addition, various round-table system development and evaluation discussions with engineering and plant

managers, lumber quality assurance experts, quality control statisticians, electrical and mechanical engineers, lumber buyers, and plant supervisors brought continual help and support.

Many other problems were solved by combining the ideas of the author with those of various people, notably, Cary S. Kiest, Joseph J. LaChapelle, Leonard West, and Stan K. Meyers. The work of this thesis covered such a wide range of problems and required such a long time to complete that it is difficult to place proper credit for the contributions involved, but this is a condition commonly encountered in modern scientific research projects. One can only take the end results as indications that some cooperation is always productive.

This thesis is dedicated to Anna Ballo, the author's mother.

TABLE OF CONTENTS

	PAGE
THESIS APPROVAL.....	ii
ABSTRACT.....	iii
ACKNOWLEDGEMENTS	v
LIST OF TABLES	xi
LIST OF FIGURES.....	xii
CHAPTER	
I PROBLEM CONTEXT	1
Operational Setting and System Overview	2
Preview of Research & Development Objectives.....	3
Wood Product Generation and Wood Grading	5
Sawmill Operations	
Wood Defects	
Grading Products	
Machine Vision System Overview	15
Underlying Technological Contexts	22
Image Processing Subsystem	
Product Optimization Subsystem	
Problem Statement	24

II	IMAGE PROCESSING SUBSYSTEM.....	26
	Background	26
	Prior Image Processing Art	
	Color Image Processing Work	
	Author's Contribution to Solving the Present Color	
	Image Processing Tasks	
	Structured Lighting	
	Basic Defect Properties	
	System Topology	34
	Image Processing	35
	Board Scanning	
	Board Edge Measurement	
	Edge Wane Measurement	
	Color Correlation	
	Defect Location	
	Edge Cut Determination	
	Image Processor Calibration	
III	COLOR CORRELATION METHOD.....	41
	Color Classification	42
	Wood Color Sets	
	Classifying RGB Colors	
	Grayscale Classification Method	
	Sigmoid-Filter Procedure.....	48
	Histogram Analysis	
	Non-Linear Filters	
	Filter Algorithms	
	Testing Classification.....	60

IV	RIP SOLUTION OPTIMIZATION	64
	Background	65
	System Topology	66
	Rip & Cut Operation CIM Organization	
	Product Optimization	71
	Computer-Based Optimization Materials Requirements Planning Board Value Optimizer	
V	SELECTING PRODUCT	78
	Weighted-Value Selection	81
	Time-Series Analysis Product Weights	
	Cumulative Probability Distribution Functions	85
	Rip Selection Method	89
	System Validation	92
VI	CONCLUSION	95
	Author's Contributions	96
	Image Processor Subsystem Summary	99
	Product Value Scheduler Summary	97
	Production Process Seen as Pull-Type Operation Product Recovery in Secondary Manufacturing Operations	

Future Work	103
Probabilities For Optimization	
Adaptive Clipper Circuit	
Raster Shape Correlation	
Grain Angle Determination	
Product Probability Scheduling	
REFERENCES	117
BIBLIOGRAPHY	121
APPENDICES	
A IMAGE PROCESSOR COMPUTER C CODE	123
B EXAMPLES OF COLOR CORRELATED WOOD IMAGES....	138
C BOARD VALUE OPTIMIZER COMPUTER C CODE.....	157

LIST OF TABLES

TABLE		PAGE
I	Defects Missed By Top Cameras For Pine Wood	62
II	Defects Missed By Bottom Cameras For Pine Wood.....	62
III	Area Bounded By Top Cameras For Pine Wood	63
IV	Area Bounded By Bottom Cameras For Pine Wood	63
V	Effects Of Using One Weight Versus Three Weight Classes To Modify The Production Of Rip-Widths	84
VI	Results Of Test 2 Using Three Weights To Modify The Product Values Of The 1-7/8" Rip-Width.....	84
VII	Results Of Test 1 Using A Single Weight To Modify The Product Values Of The 1-7/8" Rip-Width.....	85
VIII	Defect Bounding Results From Observing 100 Processed Images Of Ponderosa Pine Wood Boards	92

LIST OF FIGURES

FIGURE		PAGE
1.	Machine Vision System board scanner, operator's console, and computer monitor shown processing wood at a door and window manufacturing plant in Southern Oregon.....	1
2.	Cross section of a log showing the location and grade of boards produced by the 4-face cutting method.....	7
3.	Seven different rip solutions generated from 6 fixed rip-widths a rip saw operator may choose when ripping a board.....	8
4.	Three ways product (non-hatched areas) may be placed into a board face.....	10
5.	Profile of a shaped wood product showing the area divided into nine sections that may allow different types of defects	13
6.	Block diagram of the Machine Vision System showing product and information flow among associated processes of a wood products manufacturing plant.....	16
7.	Machine Vision System computer architecture showing communication links and messages	18

8. Machine Vision System computer data processing algorithm
showing results as rounded boxes and division across
computer platforms20
9. Data processing blocks of the Image Processing subsystem wherein
the data is processed from left to right23
10. Structured lighting used to determine 4-point board profile
dimensions31
11. Light over a workpiece is not uniform for standard flat reflectors as
brightness decreases by the square of the distance from the
bulb32
12. Light over a workpiece is nearly uniform for parabolic shaped
reflectors as the light is structured into parallel rays.....33
13. The first image processing step is to determine the board profile by
measuring the outside edges36
14. The second image processing step is to determine wane defects
using contrast measurements obtained from the first step37
15. The third step in image processing is to determine the inside and
outside edge cuts38
16. The fourth image processing step is to enhance the board image by
the adaptive color correlation method.....39

17.	The fifth image processing step uses pattern recognition to bound knot defect features with rectangular boxes.....	40
18.	Black & White image of a typical ponderosa pine wood board.....	42
19.	Color correlation conceptualized as transforming the universe of all possible wood colors from having 4 categories to having only 2 categories	44
20.	Histogram and cumulative histogram of the red color channel of an image of ponderosa pine grown in Western Oregon.....	49
21.	Histogram and cumulative histogram of the red color channel of an image of ponderosa pine grown in Southern Oregon.....	49
22.	The mathematical “sigmoid” function	52
23.	Image enhancement using the adaptive color correlation filter	52
24.	Red channel image of a typical ponderosa pine wood board.....	54
25.	Image produced by sigmoid-filter procedure of the red channel image of Figure 24	54
26.	Image produced by summing and normalizing the resultant images produced by sigmoid-filtering the red, green, and blue color channel images of Figure 18	55
27.	Block diagram of the adaptive color correlation process using one sigmoid filter per color channel	56

28.	Bit-Wise OR operation combining 5-bit pixel data with 3-bit LUT selector value	57
29.	Family of 9 sigmoid functions stored in an 8-bit LUT	57
30.	Flowchart of the adaptive color correlation algorithms implementing a grid of neighboring histograms and LUTs ...	58
31.	Flowchart of the shape recognition process using the enhanced images produced by adaptive color correlation.....	58
32.	Flowchart of the knot feature identification process.....	59
33.	A rip & cut sawmill operation where the rip-saw is controlled by a machine vision system	67
34.	The machine vision system organization for computer integrated manufacturing	69
35.	Two ways rip-widths “fill-out” a board face.....	71
36.	Family of fixed-length products comprising rip-width “A”	72
37.	Family of fixed-length products comprising rip-width “B”	72
38.	Two ways product (non-hatched areas) may be placed into the board faces of Figure 35.....	73
39.	Occurrence of the 1-7/8 inch rip-width in ponderosa pine of Common grade for a continuous production period of 212 boards.....	78

40.	The manufacturing operation illustrated as a closed-loop system	81
41.	Possible process input and output feedback used to control the ripsaw operation in a dynamic manufacturing environment..	82
42.	Hypothetical occurrence of the 3-1/2" rip-width in the sequential production of rip solutions	87
43.	Cumulative probability density function for the occurrence of the 3-1/2" rip-width shown in Figure 42.....	87
44.	Hypothetical occurrence of the 13/16" rip-width in the sequential production of rip solutions	88
45.	Cumulative probability density function for the occurrence of the 13/16" rip-width shown in Figure 44.....	88
46.	A flowchart of algorithms used to process a board model to determine alternative rip solutions, select a single rip solution to satisfy value and production goals, and monitor plant sensors to implement closed-loop process control.....	91
47.	Potential hardware implementation of the adaptive color correlation algorithm.....	106
48.	Four ways a board may be cut up when a knot is uncertain.....	113
49.	Probability tree for choosing a rip solution given uncertain information about the inclusion of a defect	114

50. Image of solid red knots in Arizona White Pine wood shown as
red, green, and blue color channel images, respectively139
51. Color correlated image of Figure 50 show as red, green, and blue
color channel images, respectively139
52. Black & white images of the color images presented in Figure 50
and Figure 51, respectively139
53. Image of core defects in Arizona White Pine wood shown as red,
green, and blue color channel images, respectively140
54. Color correlated image of Figure 53 show as red, green, and blue
color channel images, respectively140
55. Black & white images of the color images presented in Figure 53
and Figure 54, respectively140
56. Image of spiked ring knots in Oregon Ponderosa Pine wood
shown as red, green, and blue color channel images,
respectively141
57. Color correlated image of Figure 56 show as red, green, and
blue color channel images, respectively.....141
58. Black & white images of the color images presented in Figure 56
and Figure 57, respectively141

59. Image of core defects in Oregon Ponderosa Pine wood shown as
red, green, and blue color channel images, respectively142
60. Color correlated image of Figure 59 show as red, green, and blue
color channel images, respectively142
61. Black & white images of the color images presented in Figure 59
and Figure 60, respectively142
62. Image of not solid ring knot in Oregon Ponderosa Pine wood
shown as red, green, and blue color channel images,
respectively143
63. Color correlated image of Figure 62 show as red, green, and blue
color channel images, respectively143
64. Black & white images of the color images presented in Figure 62
and Figure 63, respectively143
65. Image of knot with high-angle grain in Oregon Ponderosa Pine
wood shown as red, green, and blue color channel images,
respectively144
66. Color correlated image of Figure 65 show as red, green, and blue
color channel images, respectively144
67. Black & white images of the color images presented in Figure 65
and Figure 66, respectively144

68. Image of bird's-eye knots with knot in Oregon Ponderosa Pine
wood shown as red, green, and blue color channel images,
respectively145
69. Color correlated image of Figure 68 show as red, green, and blue
color channel images, respectively145
70. Black & white images of the color images presented in Figure 68
and Figure 69, respectively145
71. Image of not solid bark-edge knot in Oregon Ponderosa Pine
wood shown as red, green, and blue color channel images,
respectively146
72. Color correlated image of Figure 71 show as red, green, and blue
color channel images, respectively146
73. Black & white images of the color images presented in Figure 71
and Figure 72, respectively146
74. Image of knot in heartwood and sapwood in Ponderosa Pine
wood shown as red, green, and blue color channel images,
respectively147
75. Color correlated image of Figure 74 show as red, green, and blue
color channel images, respectively147

76. Black & white images of the color images presented in Figure 74
and Figure 75, respectively147
77. Image of core defects in rapid-growth Pine wood shown as red,
green, and blue color channel images, respectively148
78. Color correlated image of Figure 77 show as red, green, and blue
color channel images, respectively148
79. Black & white images of the color images presented in Figure 77
and Figure 78, respectively148
80. Image of solid red knot in weathered and dirty Pine wood shown
as red, green, and blue color channel images, respectively....149
81. Color correlated image of Figure 80 show as red, green, and blue
color channel images, respectively149
82. Black & white images of the color images presented in Figure 80
and Figure 81, respectively149
83. Image of red ring knot with blue stain in Pine wood shown as
red, green, and blue color channel images, respectively150
84. Color correlated image of Figure 83 show as red, green, and blue
color channel images, respectively150
85. Black & white images of the color images presented in Figure 83
and Figure 84, respectively150

86. Image of high-angle spike knot in Pine wood shown as red, green,
and blue color channel images, respectively151
87. Color correlated image of Figure 86 show as red, green, and blue
color channel images, respectively151
88. Black & white images of the color images presented in Figure 86
and Figure 87, respectively151
89. Image of knot with growth rings in Pine wood shown as red,
green, and blue color channel images, respectively152
90. Color correlated image of Figure 89 show as red, green, and blue
color channel images, respectively152
91. Black & white images of the color images presented in Figure 89
and Figure 90, respectively152
92. Image of long spike knot, pitch pocket, and pith in Pine wood
shown as red, green, and blue color channel images,
respectively153
93. Color correlated image of Figure 92 show as red, green, and blue
color channel images, respectively153
94. Black & white images of the color images presented in Figure 92
and Figure 93, respectively153

95. Image of low-angle spike knot in Pine wood shown as red, green,
and blue color channel images, respectively154
96. Color correlated image of Figure 95 show as red, green, and blue
color channel images, respectively154
97. Black & white images of the color images presented in Figure 95
and Figure 96, respectively154
98. Image of ring knot with sap stain in Pine wood shown as red,
green, and blue color channel images, respectively155
99. Color correlated image of Figure 98 show as red, green, and blue
color channel images, respectively155
100. Black & white images of the color images presented in Figure 98
and Figure 99, respectively155
101. Image of knot with sap stain in Pine wood shown as red, green,
and blue color channel images, respectively156
102. Color correlated image of Figure 101 show as red, green, and blue
color channel images, respectively156
103. Black & white images of the color images presented in Figure 101
and Figure 102, respectively156

CHAPTER I

PROBLEM CONTEXT

This thesis discusses several improvements made to a prototype Machine Vision System designed to measure and inspect lumber to automate a rip saw operation at a door and window manufacturing plant in Southern Oregon. The installed system is shown in Figure 1. The system scans boards, processes color images, optimizes value recovery for production, and controls the rip saw networks at an average rate of 17 boards per minute.

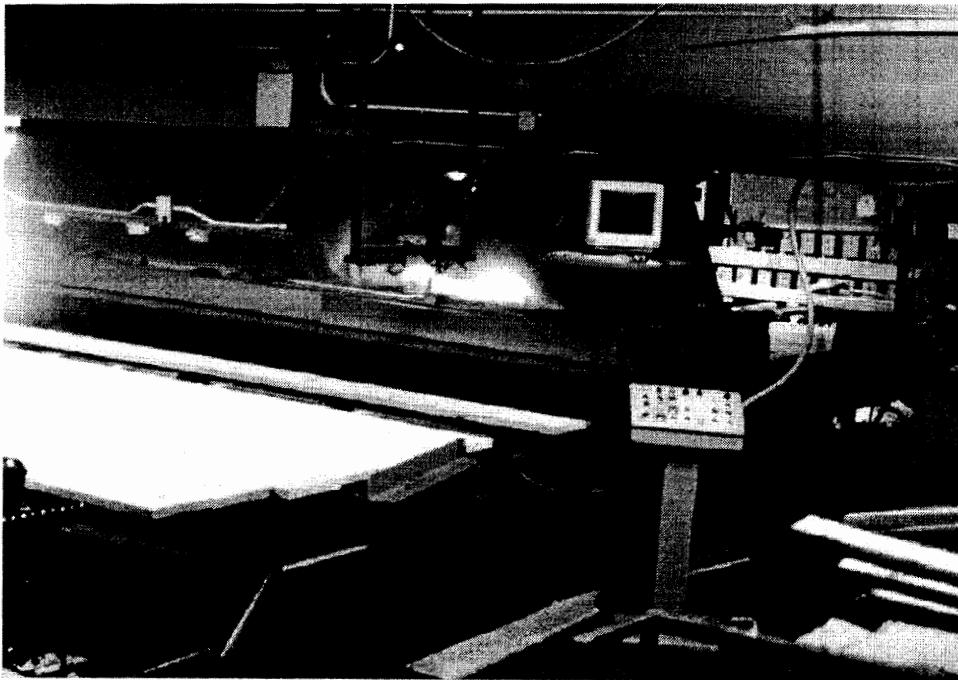


Figure 1. Machine Vision System board scanner, operator's console, and computer monitor shown processing wood at a door and window manufacturing plant in Southern Oregon.

OPERATIONAL SETTING AND SYSTEM OVERVIEW

The Machine Vision System consists of 5 specialized sub-systems: 1) Image Processor; 2) Product Value Scheduler; 3) Board Value Optimizer; 4) Master Process Controller; and 5) Programmable Logic Controller (PLC). The Image Processor uses a total of 16 color video cameras to acquire images from the top and bottom sides of boards measuring up to 24 inches wide by 16 feet long. Sixteen computers are used in parallel to process the images to determine board length and width and the location of wane, knots, and other defects. This computer architecture allows the Image Processor to measure features on each board with an accuracy of 0.050 inches and to process most boards up to 16 inches wide in under 4 seconds. Wider boards and boards with lots of defects require more processing time. The goal of the Image Processor is to reduce nearly 16MB of color bit-mapped video images to a simple coordinate-based board model text file.

Following the Image Processor, the Board Value Optimizer uses the board model data to calculate the optimum combination of rip widths to yield the maximum value recovery (highest dollar value) of constituent products. The goal of the Board Value Optimizer is to decide how to set the rip saw networks, called “rip solution,” to rip the board end-to-end into smaller widths. The Product Value Scheduler works in concert with the Board Value Optimizer to ensure that the rip solution fulfills dynamic production and inventory requirements. The rip solution is then sent to the PLC for real-time control of the rip saw and associated mechanical feeders.

PREVIEW OF RESEARCH & DEVELOPMENT OBJECTIVES

The purpose of this thesis is to discuss the research, development, and implementation of two improvements, one related to computer image processing, and the other related to optimization algorithms used for wood grading and product manufacturing processes. The improvement in computer image processing is based on modifying the image processing strategy for detecting defects in the wood. The normal strategy is to search and test for defects directly; the strategy proposed and developed here is to instead look for and classify all wood pixels containing non-defect colors first, and then apply model-based pattern recognition algorithms to the remaining unclassified pixels in the image to identify defect features. It turns out that once non-defect features are removed from an image, the task of finding defects becomes easier and faster. This improvement is realized in an adaptive color correlation algorithm developed to classify non-defect colors in pine wood images.

The biggest obstacle to consistently accurate model-based pattern matching is in developing a defect feature color that represents all possible defect colors. If a camera's color balance changes due to variations in lighting, component aging, or temperature changes, then the color of the wood is seen to change and the stored defect feature color fails to match. On examining individual pixels of a wood image, the overwhelming majority of pixels belong to non-defect features. Sampling all the pixels of a wood image yields a statistically normal representation of the color of non-defect features.

This observation leads to abandoning the usual color-model methodology and adopting a new two-step method for each image: first, determine the color of non-defect features; and second, filter out non-defect feature pixels while enhancing defect feature pixels. Not so surprisingly, the color of the remaining enhanced defect feature pixels does not vary widely under camera color balance changes or for different species of pine wood. Overall, the resulting Image Processor detects 94% of all knot defects with only 3% to 7% false detection due to grain or other miscellaneous features such as grease, burn, or boot marks.

The optimization-related improvement consists of a product value scheduler to dynamically balance the production of products generated from the rip saw for secondary manufacturing operations. The principal secondary manufacturing operation is the crosscut saw. The Machine Vision System controls the rip saw networks so that any product may be produced. However, rip solutions must maximize the recovered value of the constituent products. Product value scheduling dynamically changes the value of individual products to effect a change in the rip solutions. As the value of a product is reduced or increased, the likelihood that the product will appear in the rip solution is reduced or increased, respectively. Overall, the Board Value Optimizer and Product Value Scheduler are capable of balancing the ripsaw production rate to the crosscut saw consumption rate in under 8 minutes of continuous operation — the time it takes to generate time-averaged statistics from plant process sensors.

The remainder of this chapter provides a background to wood grading and product generation, introduces the system hardware and software architecture, and details the

scope of the problems leading to the work for this thesis. The objective of the project is to cut *clear* wood from lumber for the manufacture of door and window frames with minimum waste and maximum value recovery. *Clear* wood has no *knot* defects. Defects such as knot holes, the edges of the board, and other void-type defects are easy to detect using oblique lighting to create dark contrasting shadows. Knot defects are harder to detect because both color and shape vary from board to board.

WOOD PRODUCT GENERATION AND WOOD GRADING

When lumber was inexpensive and old growth trees were plentiful, nobody bothered to optimize wood product manufacturing processes. Prior to 1970, the cost of raw materials was about 25% of the total operating cost [1] and sawmills were oriented toward high production volume. When raw materials were relatively inexpensive, the costs associated with inefficient use of raw materials was overshadowed by the costs associated with labor, manufacturing, equipment, and storage. Today, the cost of raw materials is in excess of 70% of the total operating costs [1] and sawmills are oriented toward high quality products.

The Machine Vision System project was begun as part of an overall effort to improve the dollar value recovery of the sawmill operation. Other value recovery projects are underway to reduce the blade thickness, or kerf, of saws, and to find new uses for wood scraps previously treated as waste. In general, the wood products industry is undergoing fundamental change by introducing new and more efficient equipment, manufacturing processes, and management practices to stay profitable.

The purpose of a sawmill is to cut and process logs into clear wood products for manufacturing items such as doors and windows. Clear wood is defined by the American grading standards as free of structural and cosmetic defects [2]. While some types of waste wood can be processed into usable product, the main objective of the sawmill is to cut out solid defects or knots that will cause structural weakness or chip-out during secondary moulding and shaping manufacturing processes. This section provides a background to the sawmill and wood cutting practices as well as an overview of how this project was begun.

Sawmill Operations

A simplified description of the process of cutting a tree into usable products follows. A tree is cut into log segments and stripped of bark. Logs lengths vary from 6 feet to 28 feet. The log is then cut along its length into boards or planks of uniform thickness but various widths, illustrated in Figure 2. Board widths vary from about 8 inches to 24 inches. The boards are sorted by grade, stacked into lifts*, and cured to harden the sap. (Wood grades are explained in a following subsection of this chapter.)

Each board is then surfaced on the top and bottom (S2S), and the board is ripped* into a fixed widths. The rip saw operator selects any combination of fixed rip widths that utilizes the full width of the board, illustrated in Figure 3. Many rip saws are capable of

* A “lift” contains about 250 boards and is a standard unit of trade; however, all lifts are bought and sold based on the grade of lumber which represents the expected board feet of usable clear wood.

sawing up to 8 different widths from a single board. Defects are then crosscut[†] out of each ripped board, and any remaining clear wood is cut into fixed product lengths. The smallest usable clear length is always greater than the rip width. The cut product lengths are then put on pallets and used in other wood product manufacturing operations.

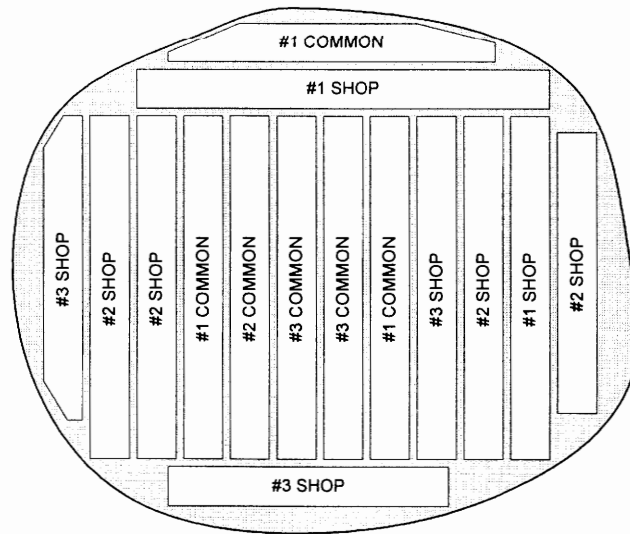


Figure 2. Cross section of a log showing the location and grade of boards produced by the 4-face cutting method.

Historically, only the crosscut saw operator has direct knowledge of how the desired product lengths are fit between defects to make clear wood products. The rip saw operator and the log Sawyer operate under the optimization strategy of maximizing the longest and widest clear wood areas. The rip saw operator, for example, will place the widest rip width in the location of the board that yields the longest clear section. He then

* The term “ripping” refers to the sawing practice of cutting wood with the grain, or transversally end-to-end.

[†] The term “crosscutting” refers to the sawing practice of cutting wood against the grain, or normal to the length.

fills out the remaining board width with other rip widths using the longest and widest strategy. This strategy is adequate; however, waste occurs when a clear section is either greater than or less than a fixed-length product. To reduce waste and recover more value, knowledge of product lengths required in subsequent manufacturing operations must be introduced before the board is ripped.

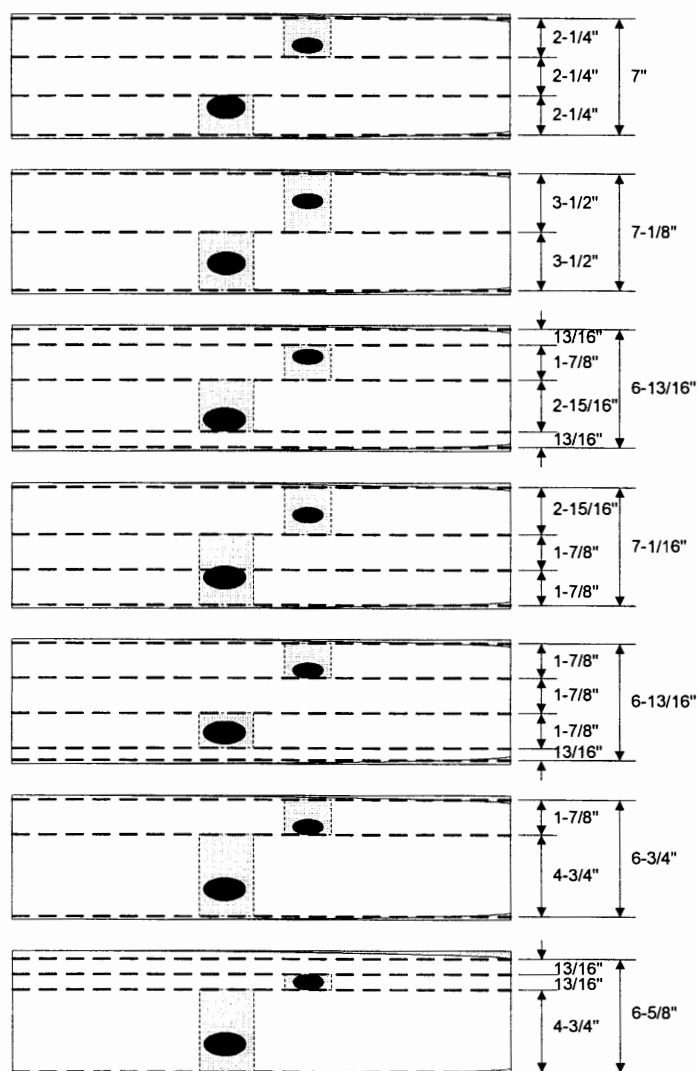


Figure 3. Seven different rip solutions generated from 6 fixed rip-widths a rip saw operator may choose when ripping a board; note that hatched areas represent waste to be crosscut out by the crosscut saw operators.

The method of sawing lumber is similar to solving a jigsaw puzzle. The lumber puzzle is played with rectangular blocks that are placed between knots, and the blocks must be arranged in columns (or rips) of the same width (called “rip-width”). To complicate matters, each block is worth a certain value and the goal of the puzzle is to find the right pattern of blocks that yield the highest total value.

If wood were mostly clear (sparse placement of defects), a very simple optimization strategy could control the production of products. However given the goal of maximum value recovery, not every rip width is economically produced from any one board. As a rule of thumb, the likelihood of producing long product lengths from a board decreases as the width of the rip increases. That is, every board will produce long product lengths for the 13/16” rip width, most every board will produce long product lengths for the 1-7/8” rip width, some boards will produce long product lengths for the 3-1/2” rip width, and few boards will produce long product lengths for the 4-3/4” rip width.

Figure 4 illustrates three different ways a board may be cut into product. Hashed areas represent waste. The top solution is worth \$22.50 and does not utilize wood in the upper right hand corner. The middle solution is worth \$23.00 and does not utilize wood in the lower left hand corner. The bottom solution is worth \$26.50 and fully utilizes the wood. However, this solution would require an unconventional sawing sequence of crosscutting first, then ripping, then crosscutting again. The work described herein focuses on the more conventional procedure of ripping first then crosscutting.

Studies of sawmill operations point out that human operators are unable to achieve optimum solutions because of the complexity of product placement over the

entire length of the board face [1][3][4]. In general, when longer length products are worth more than shorter length products, ripping first then crosscutting maximizes the production of longer length pieces. However, a study by Michigan State University suggests that the crosscutting, then ripping, then crosscutting method is expected to increase yield by 23% [5].

The purpose for implementing a machine vision system is to improve existing ripping then crosscutting facilities, and provide continuous gains in value recovery. The Michigan State University study suggests that retrofitting ripping first then crosscutting type operations will improve yield by 3% [5]. Moreover, a machine vision system is retrofit to an existing operation to yield consistent end-product quality.

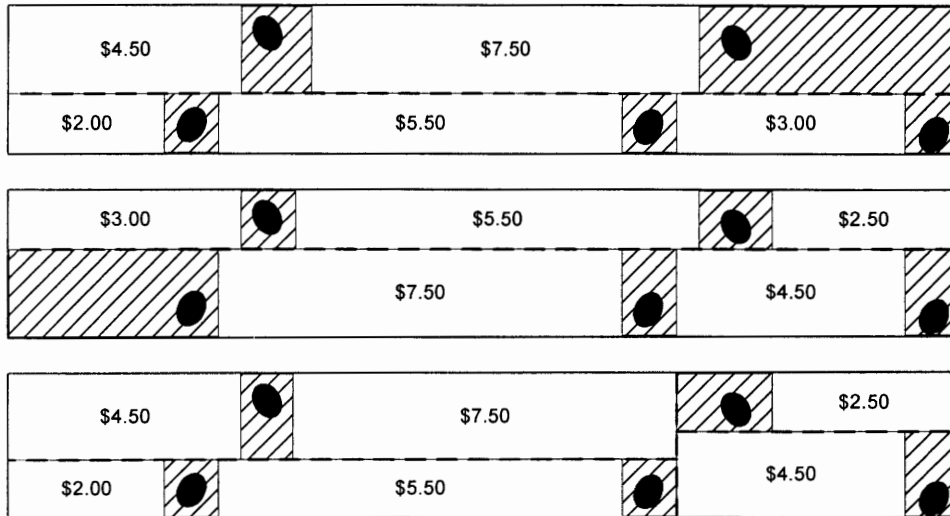


Figure 4. Three ways product (non-hatched areas) may be placed into a board face; note that the bottom solution would require an unconventional sawing sequence of crosscutting, then ripping, then crosscutting again.

Wood Defects

Sawmills must maintain large amounts of inventory in part because the production of wood products is random and cannot be predicted well. No two trees are exactly alike. Therefore the location of defects in wood is random, and no two boards will yield exactly the same products. Other traditional justifications include fixed plant capacity, increasing returns of scale, market uncertainty (stochastic variability), seasonality (predictable variability), and the complexity of dynamic lot sizing procedures. Product value optimization is the balance between maximizing recovered value (the bottom line), minimizing inventory holding, and maximizing manufacturing capacity for a given production rate.

Defects are not desired in the end products. Wood defects are knots, rot, pitch pockets, voids, surface checks and splits, and blue-stain. Void defects include knot holes and wane or bark edge. Warped wood is not defective and is typically usable in short-length wide rip width products. Rip and crosscut saw operators find such defects using their visual and tactile senses.

To improve the speed and recover more value, various machine vision systems have been developed that use cameras and computers to automatically find defects and optimize product in the board [6]. The most effective of these systems create a color bit mapped image of the board and then test the pixels of the image against a color model [7]. Pixels that closely match the model are considered defect pixels. Areas of contiguous defect pixels constitute wood defects. These systems, however, lack consistency and accuracy when presented with wide variations in wood color.

Pine softwood, for example, comprises 14 species of wood: Eastern White, Jack, Loblolly, Lodgepole, Longleaf, Pitch, Pond, Ponderosa, Red, Shortleaf, Slash, Sugar, Virginia, and Western White. Each species is different in color. For instance, Eastern White Pine is pasty-white in appearance with light reddish colored knot defects, Ponderosa Pine is rich yellow in appearance with dark brown colored knot defects, while Red Pine is reddish in appearance with green-brown colored knot defects. Many of the aforementioned systems cannot consistently distinguish heavy grain in wood from a knot. While both are darker than clear wood, only the knot is a defect.

Grading Products

After a rip or crosscut saw operator locates defects in a board face, the operator must grade the clear wood using standardized rules. The purpose of grading wood is to set a standard for commercial trade. American grading standards are formulated by the National Bureau of Standards of the Department of Commerce. Softwood lumber is classified into three main categories as determined by appearance and mechanical strength: 1) Yard Lumber; 2) Factory & Shop Lumber; and 3) Structural Timber.

Yard Lumber. Yard lumber has nine grade types. Grades A & B are called “select lumber.” Grades C & D are called “finish lumber.” Grades 1-5 are called “common lumber.” As grades progress from A to 5, the allowed defects increase. For example, grade A is “practically free from defects,” No. 1 allows “sound and tight knots,” and No. 5 “must hold together under ordinary handling.”

Factory and Shop Lumber. This lumber is graded for use in products such as furniture, window and door frames, and so on. It is visually graded for the amount of wood in a given face that is free or clear of defects in specific areas along the length of the product. The majority of Factory and Shop Lumber grading rules are developed by manufactures for specific secondary manufacturing operations involving molding or shaping equipment. An example of the profile of a shaped wood product and the type of defects allowed is shown in Figure 5.

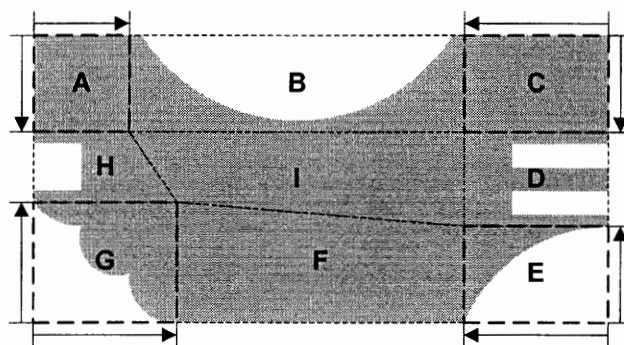


Figure 5. Profile of a shaped wood product showing the area divided into nine sections that may allow different types of defects.

For manufactured products, grading involves dividing the 4 surfaces or faces into 8 dimensioned sections which run the length of the product. Referring to Figure 5, surface grading areas are dimensioned off the corners A, C, E, and G. Each area may contain various types of defect features defined by how the dimensioned sections are used. For example, area F must be clear or free of cosmetic defects such as knots, pitch pockets, and blue stain because it is visible in the finished product. Areas E and G may contain wane or bark edges (defects) because the lack of wood makes no difference in the appearance of the finished product after shaping. Because knots cause wood “tear-out”

during shaping, areas H and D must be clear wood. Areas A, B, C, and I may contain any single-sided structural, molding, or cosmetic defect features because this part of the product is hidden from view.

Grade comprises both quality and end-use, and defects must be classified by both size and appearance. A defect feature is a gauge for defect type, size, and appearance. For example, three small red or solid knot defects may visually appear as one defect feature when located close together. The same three red knot defects may not make a defect feature when located far apart. Most grading rules are highly subjective in nature.

Structural Timber. This lumber is graded by size and use. Beams and stringers, joists and planks, posts and timbers, and pilings comprise the major classifications. Of particular interest are the Premium and Architectural Appearance for B&BTR, C, Select Structural, Selected (grain tight), No. 1, No 2, and Insulation grades. For example, “C” grade permits the following defects for a 9” wide face: no more than three sound and tight 1” knots for each 12’ of length. For a 10” wide face: no more than three 1-1/4” knots for each 12’ of length.

Structural timber is also graded by allowable stress loading. Stresses are internal forces on wood fibers caused by external loading. Maximum allowable loads apply only to the finished product, however. When stresses exceed the elastic limit, wood fibers begin to tear apart in failure. This is observed as load-induced shake and is called “fiber stress at the proportional limit.” Mechanical tests determine this grading by measuring how much load is required to deflect a beam to the proportional limit. The “F” grading, e.g., 1450f, 1700f, etc., from such a test is the maximum allowable fiber stress.

MACHINE VISION SYSTEM OVERVIEW

The Machine Vision System is a network of four personal computers, sixteen vision processors, and one programmable logic controller (PLC). The system is designed to process video images to automate a rip saw operation in a wood products manufacturing plant. The sixteen vision processors and one of the personal computers are used to process the video images. One personal computer is a graphical user interface (GUI). One personal computer is dedicated to determining a rip solution. The fourth personal computer manages the process and communicates with the real-time PLC controlling the rip saw (NAC). This section presents the operating environment, computer architecture, and overall computer program requirements of the system.

The system operating environment is illustrated in Figure 6. This block diagram depicts a number of interrelated processes acting upon a common production line. The operating procedure is as follows:

- 1) Configure production goals and product parameters;
- 2) Place raw materials into operation;
- 3) PLC operates drop feeders to singulate* a board;
- 4) Image top and bottom of board into video memory;
- 5a) PLC operates feeders to singulate another board;
- 5b) Process the scanned board images for profile and defect coordinates;
- 6a) Return to step 4) and continue;
- 6b) Determine rip solution by maximum product value optimization;
- 7) Send rip solution to PLC to set rip saw networks;
- 8) PLC operates feeders to run first board through rip saw;
- 9) PLC sends each rip-width produced to crosscut operation;

* The term "singulate" refers to automated materials handling processes where a single board is separated from a group using a series of feed gates, pin stops, jump wheels, sweep arms, and drop-table mechanisms along a continuous chain conveyor.

- 10) Saw operators crosscut products from clear wood and inventory;
- 11) PLC monitors production rates of crosscut saw operation;
- 12) Product value parameters are reconfigured to meet production goals and balance rip-width production rate to crosscut saw production rate.

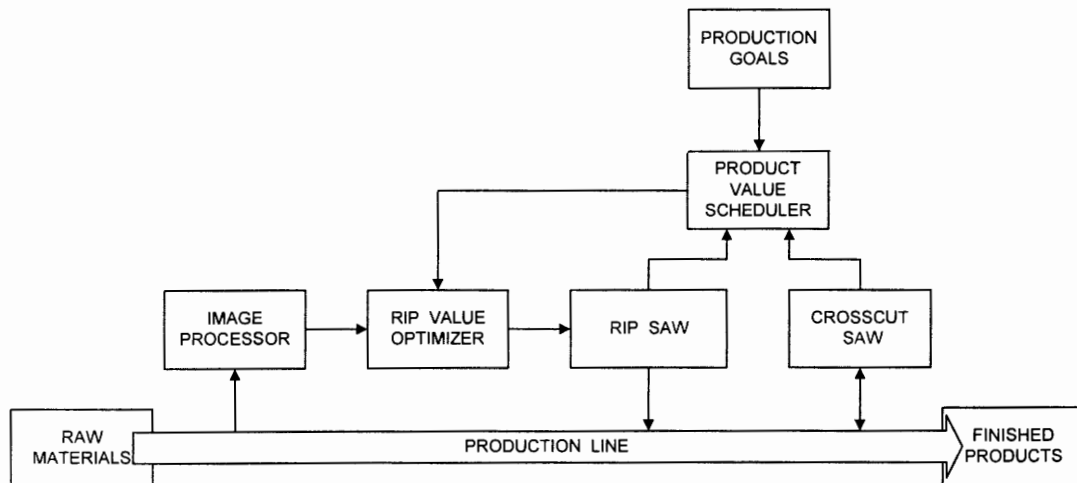


Figure 6. Block diagram of the Machine Vision System showing product and information flow among associated processes of a wood products manufacturing plant.

For the Production Line of Figure 6, lifts of graded kiln-dried boards are taken from inventory and arranged side-to-side transversally by an automated jack hoist and personnel operating a conveyor break-down table. The break-down table uses drop-feeders, also called pin stops, to singulate a single board through the production line. The singulated boards pass through the Image Processor where they are visually inspected. A stacked-blade saw rips each board based on a rip solution generated by the Rip Value Optimizer from information supplied by the Image Processor. The ripped wood is then conveyed to the crosscut saws where operators visually inspect the wood and cut the wood into specific length products based on a production order. The cut products are then stacked and placed in inventory.

As an independent supervisory process, the Product Value Scheduler of Figure 6 monitors the production goals, production rate of the rip saw, and consumption rate of the crosscut saws. An internal model of short and long-term production rates is derived from the actual rates to filter out random spikes and drops. These spikes and drops occur when boards jam and stack on top of each other or when boards jam in the equipment.

If the production rate is greater than the rate of consumption, the value of the over-produced product is (mathematically) reduced incrementally to zero. If the rate of consumption is greater than the rate of production, the value of under-produced product is increased incrementally to market value. This is the method of production control.

Figure 7 illustrates the Machine Vision System computer architecture. This diagram depicts four personal computers, communications network, and attached co-processing hardware, i.e., PLC and Acumen 900 vision processors. The operating procedure is as follows:

- 1) User configures system via GUI computer (Product Value Scheduler);
- 2) Ripsaw operator puts the system into "auto" mode via PLC console;
- 3) PLC sequences a board into the scanner and signals the MPC computer;
- 4) Vision processor acquires top and bottom images of the board;
- 5) Vision processor measures board profile every 2 inches;
- 6) Vision processor uses pattern recognition to find knot defects;
- 7) Vision host computer assembles profiles and defects into board model;
- 8a) Return to step 3) and continue;
- 8b) Board model is sent to optimization computer;
- 9) Optimization computer processes board model into rip solution;
- 10) MPC sends the rip solution to PLC and GUI computer for logging;
- 11) PLC sends crosscut saw operation production rates to GUI computer;
- 12) Product value is reconfigured to balance ripsaw production rates with crosscut saw production rates and system production goals.

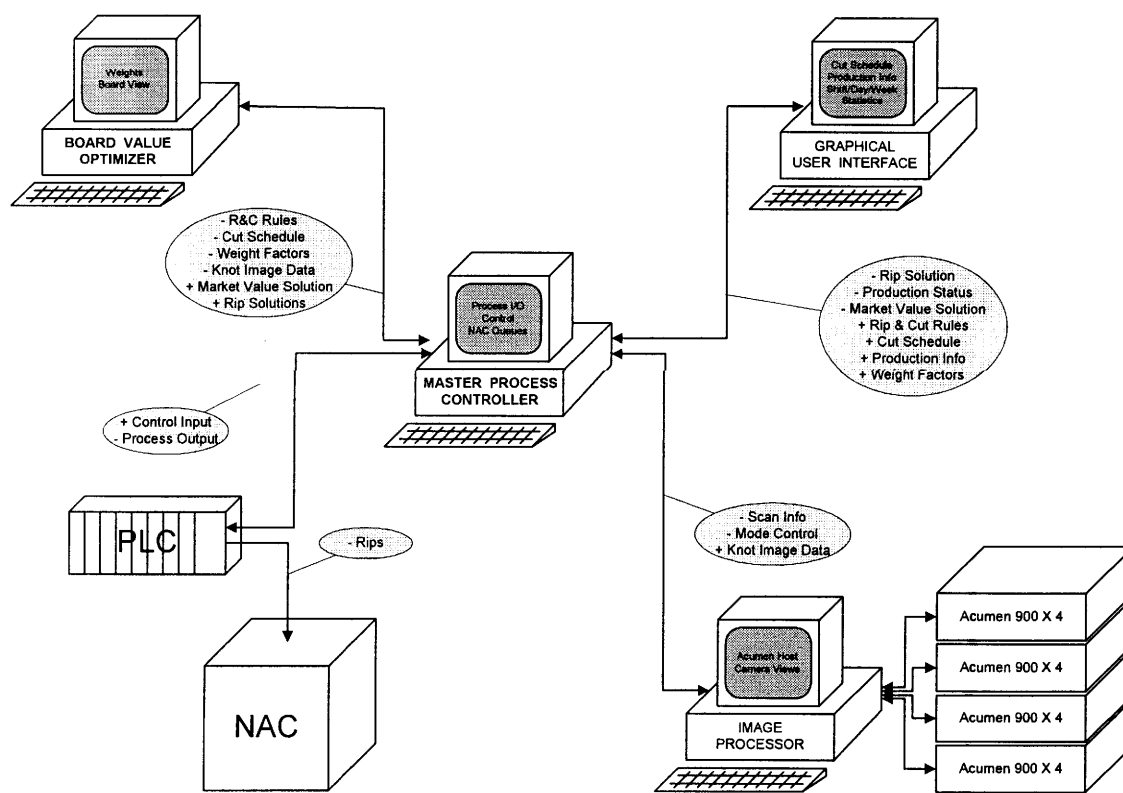


Figure 7. Machine Vision System computer architecture showing communication links and messages.

The Image Processor personal computer, lower right portion of Figure 7, acts as a host processor for 16 Acumen 900 vision processors. Four Acumen 900 processors reside in a single VME chassis. Each Acumen 900 processor contains a video interface for cameras and monitors, video memory, a general purpose AT&T DSP32C CPU, and a Rockwell correlator chip which performs the shape-based pattern matching.

Each of the Acumen 900 processors analyzes a 2 foot long section of the board and executes the same algorithms. This parallelism is required for the system to process a board image in under 4 seconds, i.e., real-time processing. Note that each processor works asynchronously in multiple instruction multiple data (MIMD) fashion.

The Master Process Controller (MPC) computer, in the center of Figure 7, sequences all processes and communicates with the PLC. The main purpose of the MPC is to level processing resources. The most important aspect of real-time processing is dedicating available processing resources to critical-path tasks. All processing tasks related to producing a rip solution are on the critical path. Updating the video screens, reconfiguring product parameters, monitoring production rates, and so on are all non-critical background processing tasks. A second layer of real-time processing is provided by the PLC. PLC processing tasks include emergency stop, feeder actions, photoeye sensors, rip saw control (NAC) and the like.

The Board Value Optimizer (BVO) and Product Value Scheduler (PVS) work in concert to generate a rip solution. The PVS utilizes a GUI interface to display production information for the shift, day, and week. The PVS also maintains the cut schedule containing the current product configuration. The BVO relies on several sources of information to generate the rip solution. This information includes the dimensions of products to produce, present market value of products, types of defects that may be allowed in products (called “grade”), production goals for the present run of products, production to date, and production rate. The BVO tries all combinations of products that can be produced (called “rip solutions”) and selects the highest valued rip solution.

Figure 8 illustrates the Machine Vision System data processing algorithm. This flowchart depicts three independent processes comprising Image Processor tasks, Rip Value Optimizer tasks, and Rip Production Scheduler tasks. The flowchart tasks are implemented as follows:

- 1) Two board images are scanned and placed in video memory;
- 2) Profile information is calculated from the images, directly;
- 3a) The images are enhanced to normalize the knot color;
- 3b) Shape-based pattern recognition is applied to the enhanced image;
- 4) A board model is sent across the network to the BVO computer;
- 5) All possible permutations of rip-widths are applied to the board model;
- 6) Product is applied in each permutation, highest value solution is saved;
- 7) For each permutation, the product values are multiplied by weights;
- 8) The highest value solution is sent to the rip saw;
- 9) Sensors monitor the plant for rip saw and crosscut saw production rates;
- 10) Product weights are generated from difference in production rates.

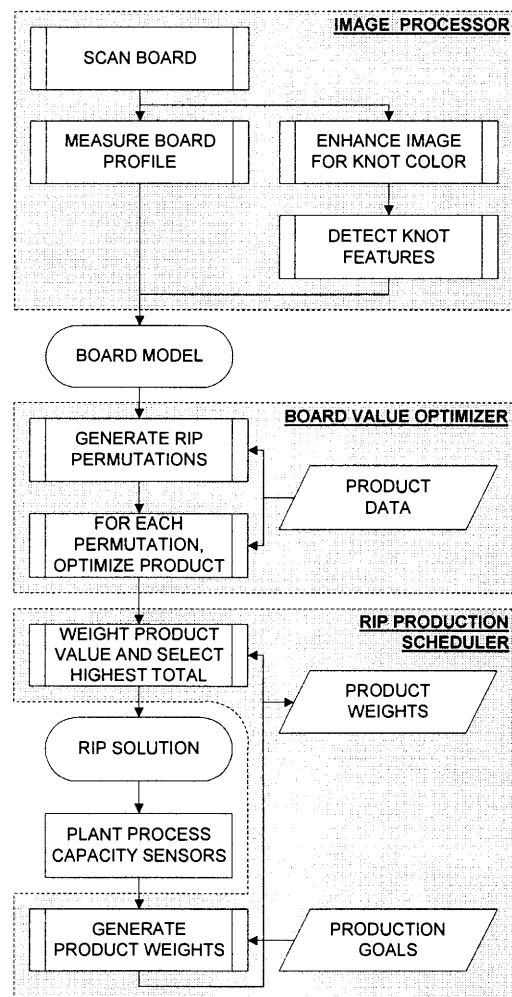


Figure 8. Machine Vision System computer data processing algorithm showing results as rounded boxes and division across computer platforms (grey shaded areas).

This Image Processor data processing tasks, flowcharted in Figure 8, are decomposed into four major programming algorithms. Codework to scan and acquire board images into video memory forms the task sequencing infrastructure. The program is blocked from processing until a scan is requested from the MPC computer. Once a scan is initiated, the computers devote all resources to processing the data as quickly as possible. Since processing time is a function of the number of defects in the board and is thus random, this bucket-brigade sequencing methodology is very effective.

The purpose of the Image Processor algorithms is to reduce raw video data to a simple text file, called “Board Model,” containing the board profile measurements and defect coordinates. While measuring the board profile is accomplished from the raw video data, identifying defect knot features requires enhancing the images for knot color.

The Board Model is sent to the Board Value Optimizer which determines all possible rip solutions supported by the board model. Rip solutions are generated by permutations of the product rip-widths across the width of the usable board face. In an intermediate step for each rip-width, products that can be crosscut out of the rip-widths are determined. These processing tasks rely on product dimension and market value data from the customer.

One rip solution is then chosen by the Rip Production Scheduler and is sent to the rip saw networks to rip the board. The rip solution yields the highest total value of products given the present production capacity and goals. The selection algorithms multiply the market value of products by weights which range in value from 1.0 to 0.0.

The weights are generated as plant feedback to balance the rip saw production rate with the crosscut saw production rate.

In summary, the machine vision system controls a rip saw operation by visually scanning boards and monitoring the downstream plant processes. Image enhancement and pattern recognition are discussed in Chapters II and III. Product optimization and weighted-value selection are discussed in Chapters IV and V.

UNDERLYING TECHNOLOGICAL CONTEXTS

This thesis discusses two improvements, one in the field of image processing and one in the field of production-based product optimization. For image processing, a color correlation method is developed and shown to outperform present model-based color recognition algorithms. For production-based product optimization, a weighted-value selection method is developed and shown to outperform present weighted-value optimization algorithms. This section discusses technical issues underlying the problem-solving method in each field.

Image Processing Subsystem

Figure 9 illustrates major technology components of the Image Processing subsystem. In general, image processing and optimization problem solving efforts involve signals processing. Within the signal processing context, data is collected by the area cameras (left-most block in Figure 9) and is modified and processed in various ways until it exits the rip optimization procedure (right-most block in Figure 9).

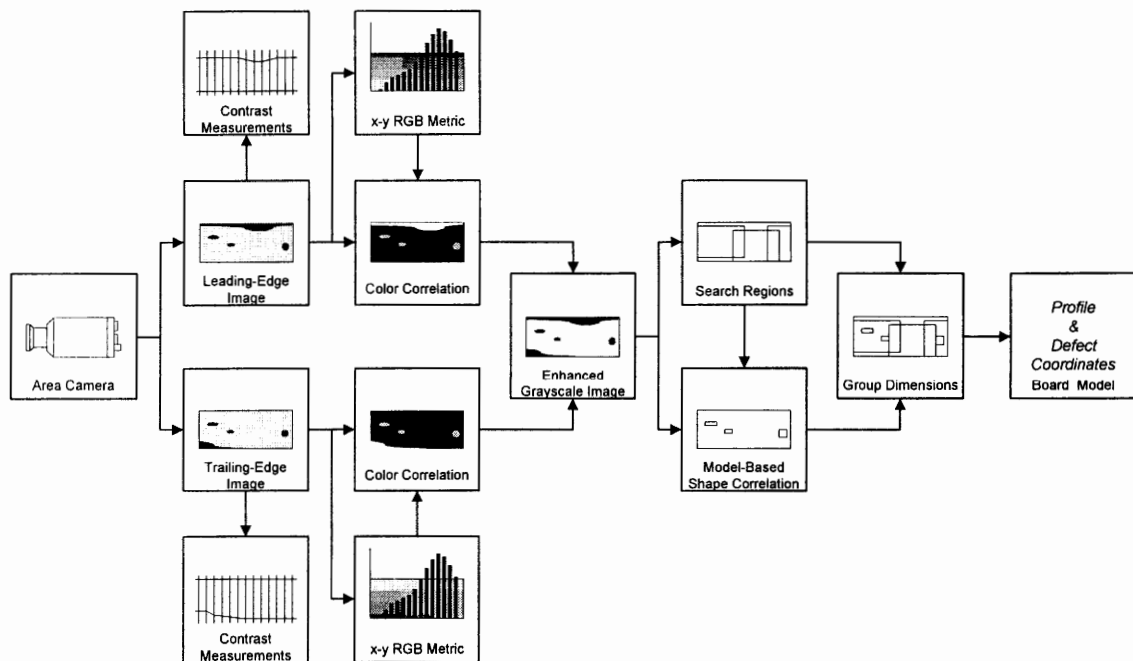


Figure 9. Data processing blocks of the Image Processing subsystem wherein the data is processed from left to right.

Image Processing subsystems comprise cameras, computers, and software algorithms. Cameras use optics and analog video signal processing circuitry. In this work, the camera is modified to effect a form of data compression called “clipping.” Referring again to the data processing blocks of Figure 9, images are stored as bit maps in video memory and require high-level and assembly language DSP (computer) programming to perform contrast measurements and other image processing tasks.

Color correlation relies on statistical measures to develop metrics for signal filtering that produce enhanced color images. Pattern recognition is a 2-dimensional type of signal processing. A shape model is developed as a raster pattern of individual pixels. Pixels from the shape model are compared with image pixels of interest in raster fashion

using correlation mathematics. If the majority of model pixels correlate with image pixels, the image is said to match the shape model. Separating the context of color correlation from the task of pattern recognition benefits the context of shape correlation.

Product Optimization Subsystem

Board value optimization is the process of sequentially trying all possible solutions of a problem and choosing the highest valued solution. Generating possible solutions involves using feedback from the process in closed-loop control fashion. Aside from general mathematical techniques, rip optimization uses signal processing techniques to effect overall system control. Separating the context of signal processing from the task of rip optimization benefits the context of product optimization. In summary, this section discussed the overall context of signal processing and the following chapters develop material from this context.

PROBLEM STATEMENT

This section summarizes Chapter I in terms of a problem statement. This work solves two basic problems: 1) increase the robustness of automatically detecting knot defect features in the Image Processing subsystem; and 2) balance the rip saw production rate to the crosscut saw production rate and maximize the market value of products produced in the Product Optimization subsystem. This work describes the observations that led to the development and implementation of, and the results obtained from using the design solutions in a specific application as a Machine Vision System installed in a sawmill plant in Southern Oregon.

Pattern recognition using average-color models fails when the color of knot features in wood does not correlate to the model. The color of the feature may change because: 1) the color of wood changes across species of trees and for different geographical locations of growing; and 2) the camera used to image the wood changes the color of the wood. Solving this problem required developing a method of enhancing and normalizing the observed color of a wood image so that the color of knot defect features are well separated from the color of non-defect wood.

Balancing the rip saw production to the crosscut saw production fails when a single rip-width-based weight, used in the optimization value equations, is applied to all products produced by that rip width. Because the market value of product does not change with respect to the manufacturing operation, in such equations decreasing the value of over-produced product shifts the maximum value optimization process to produce non-optimum rip solutions. Solving this problem required developing a secondary rip solution selection process that does not effect the maximum value optimization process used to balance production.

Chapter II and Chapter IV introduce the Image Processor and Optimization paradigms and describe the above problems in detail. Chapters III and Chapter V detail the solutions implemented to solve the above problems, respectively. Chapter VI summarizes the work and results. Chapter VI also presents recommendations and future work tasks developed from this work. Following Chapter VI are a series of appendices containing the computer code implementing selected algorithms presented in Chapter III and Chapter IV.

CHAPTER II

IMAGE PROCESSING SUBSYSTEM

The Image Processing subsystem scans and processes pine boards up to 24 inch wide by 16 feet long at an average rate of 17 boards per minute. The Image Processor performs five primary functions. These functions include board scanning, board edge measurement, edge wane measurement, defect location, and edge cut determination. Once these functions are accomplished, the Image Processor passes data along to the Optimizer to produce rip solutions that control the rip saw. In addition, the Image Processor performs a calibration function for defect coordinate accuracy maintenance purposes. These functions are described in more detail in later sections of this chapter.

BACKGROUND

To build an Image Processor, one must plug cameras and computers together and write software that performs mathematical algorithms on individual and groups of pixels. The purpose of Image Processor software is to reduce (lots of) raw pixel image data into a simplified and perhaps higher-level representation (like defect coordinates). Simple image processing techniques use structured lighting to create crisp black & white contrast edges to measure features. Contrast-based measurements determine board width dimensions or profile, bark edge or wane edges, and void-type defects. This section

provides introductory material for the more advanced techniques of image processing to be developed in Chapter III.

Prior Image Processing Art

The work in this thesis builds upon previous developments in Image Processing technology such as defect identification, color space definition, and pattern recognition. A proposal for analyzing wood using automated methods to detect defects in wood dates as far back as 1964 when electronic x-ray fluoroscopy was commercially investigated for detecting rot in wood [8]. Later, a television-like network of filters and amplifiers to process the signal from an x-ray sensitive device to yield a binary indicator of the presence of solid defects (knots) to make lumber grading and sawing decisions was proposed [9].

Beginning in the late-1970s, optical* image processing methods were proposed to determine the presence of surface defects in wood by differentiating tone†. Because the reflective properties of bound wood (knots) differ from that of sapwood, the first commercial optical scanning systems to detect knots used lasers or other bright light sources and intensity-based broad-spectrum photo detectors [10][11][12]. However, such black & white optical image processing systems achieved only mediocre success because

* The term “optical” refers to reflective methods of surface analysis typically involving the visual spectrum, infra-red, and ultra-violet wavelengths, and does not include transmissive methods of density analysis typically involving microwave and x-ray wavelengths.

† The term “tone” refers to a tint or shade of color and to the degree of brightness of a texture in a color pattern recognition processes.

many wood defects must be identified by differentiation based on surface color [3][4][13][14]. When a Machine Vision System misses a defect, poor cut decisions are made and production re-work costs increase. Even a moderate increase in production re-work is sufficient cause for a manufacturing plant to abandon new automated systems. Additional background material in image processing are provided in the bibliography.

Color Image Processing Work

The present project began with software capable of finding knot defects in wood using pattern matching algorithms called shape correlation. In shape correlation, a model shape is compared to candidate pixels in a bit-mapped image. Correlation is a measure of how well the gray-scale values of each color channel of the candidate pixel matches the gray-scale value of each color channel of the model pixel. An exact match yields a correlation value of 1.0 and no match yields zero correlation value.

Correlating a model to an image is represented as the normalized sum of the individually correlated pixels over a target area in the image. If a possible candidate feature has a high correlation to the model shape, then the area of the image is labeled a *knot* defect. In color vision processing, color image correlation treats the red, green, and blue color channels as independent gray-scale images. The color correlation result is a normalized sum correlation value over a target area in the image that matches both a specific color and a shape feature.

Color-Based shape correlation by itself works well only under ideal conditions because it relies on pre-defining the average color properties and shape of the feature of

interest. For pine wood, *Knot* defect color “information” is mostly carried in the *red* and *blue* color channels. *Clear* non-defect “information” is mostly carried in the *red* and *green* color channels. Because the color of pine wood varies widely with species and growing location, it is not possible to describe all possible color models of knot defect color without also describing some clear non-defect colors.

When the color model is too general, pattern recognition fails because too many non-defect pixels match the model. When the color model is too specific, pattern recognition fails because too few defect pixels match the model. Typically, lifts (about 250 boards) of different color (different source or grade) lumber are purchased and used in random order. Failure on any single lift of lumber is unacceptable.

Furthermore, installing a video camera in a lumber mill subjects it to a less than ideal environment. The biggest obstacle to consistently accurate shape correlation is color balance drift of the cameras. If a camera’s color balance changes due to temperature variations, component aging, or contamination by sawdust, then the color of the scanned wood is seen to change and model-based color correlation fails.

The best available color model was measured to fail after only ± 2.5 points of drift in hue out of 256. The goal for the present Image Processing project was to determine a method to increase the robustness of pattern recognition for knot features and to overcome the effects of camera color balance drift.

Long ago, Aristotle defined visual perception as determining “what is where.” More recently (1970s), David Marr of MIT defined visual perception as recognizing feature attributes such as shape, size, color, and texture. Marr’s work proposed that

pattern recognition be accomplished using independent specialized models. Research in pattern recognition has shown the limitations of the model based approach. Specifically, the models themselves are intractable. That is, robust pattern recognition requires several different shape models, since an object's features can vary in many subtle ways.

Author's Contribution to Solving the Present Color Image Processing Tasks

By applying creativity principles to this problem, the author reasoned that human vision systems typically first look at the entire board in a general way for what is *not defect* (clear wood) before performing the specific task of defining a possible defect (knot wood). This method identifies an area of interest that the human vision system then analyzes in greater detail (possibly applying shape analysis) to find knot defects. This observation suggests a new two-step image processing method: first, use color correlation to identify non-defect *clear*; and second, use shape correlation to identify *knot* defects within the regions identified as *not clear* (or what remains).

This method overcomes moderate color balance drift because it relies on the camera's color images to define what is *clear* wood, and clear wood comprises the majority of pixels in the wood image. This may sound like a simple insight, but to date all the methods presented in the literature have focused on identifying defect features and not on finding non-defect features. In preview, Chapter III develops a method of finding statistics about the color of non-defect wood, filtering-out non-defect color pixels and enhancing knot defect color features using sigmoid transforms.

Structured Lighting

Structured lighting is used to determine the board profile (outside edge dimensions and bark or wane edges), void defects such as knot holes, splits, shake, and certain types of surface-texture defects such as planer skip and saw abortions [15] [16][17][18]. As boards travel into the Image Processor scanner, see Figure 1, they are stopped with an internal fence while the lights are activated and the video cameras acquire images. Figure 10 illustrates the concept of structured lighting whereby a low lighting angle from one side produces dark shadows on the opposing wane or bark edge.

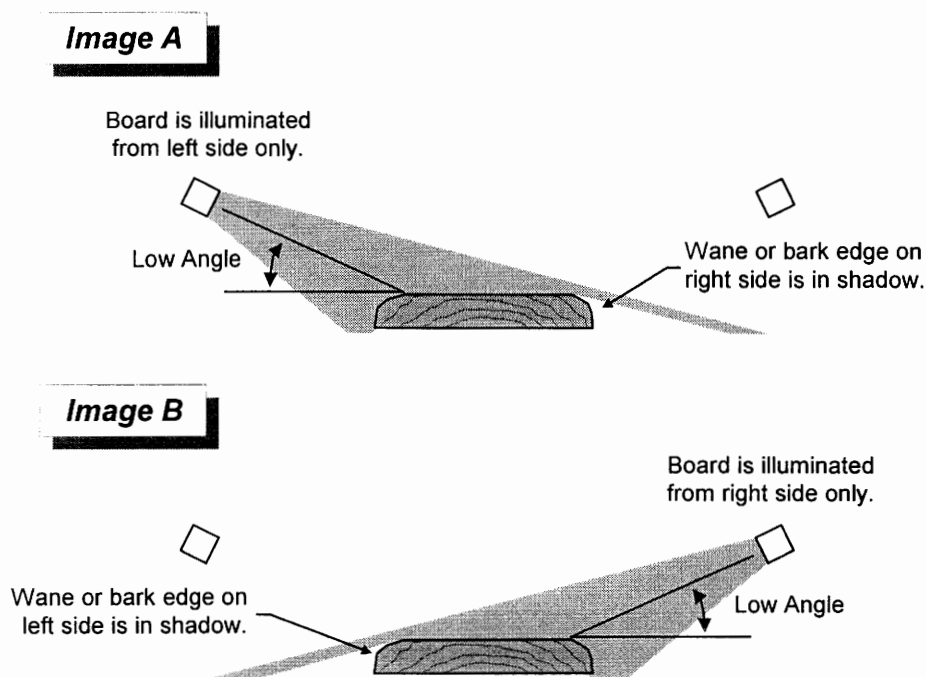


Figure 10. Structured lighting used to determine 4-point board profile dimensions.

To determine the profile of the board, each camera captures two separate images of the board, and the entire board is scanned by a total of 16 color video cameras. Eight

cameras image the top face of the board and eight cameras image the bottom face in overlapping 2 foot long segments of 640 by 520 pixels. Once all 32 images are acquired, the fence is lowered to release the board from within the scanner. Each pair of left- and right-lit images are then sent to one of 16 image processors for analysis.

The most important step to improve the color rendering of a workpiece is to ensure sufficient and uniform illumination. Lenses and reflectors are used to structure the light and eliminate the bottom shadows cast by the support chain raceways. As pictured in Figure 11, the standard flat reflector produces a gradient of light across the board. That is, the board surface closest to the bulb is very bright and the surface farthest away from the bulb is very dim.

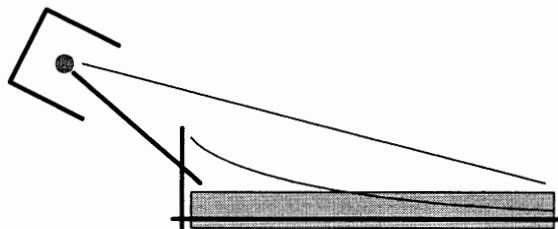


Figure 11. Light over a workpiece is not uniform for standard flat reflectors as brightness decreases by the square of the distance from the bulb.

Figure 12 shows an improved design using a parabolic reflector to deliver uniform light across the board. Having uniform lighting allows both the leading and trailing lit images to contribute to the color correlation process.

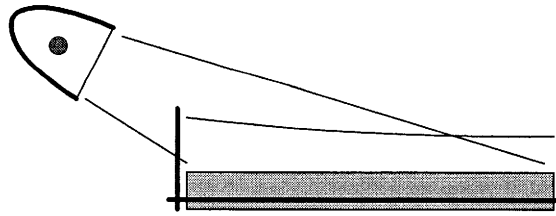


Figure 12. Light over a workpiece is nearly uniform for parabolic shaped reflectors as the light is structured into parallel rays.

Unfortunately, parabolic reflectors are expensive. A perforated metal *gradient filter* achieves the same effect and is a cost-effective solution to producing uniform (but dimmer) lighting. On investigating this idea, a gradient filter was produced using a laser printer. Unfortunately, this design projected lines onto the board surface. A conclusion drawn from this work is: use very small randomly spaced “objects” to construct the gradient filter. This conclusion led to using aluminum paint – to reflect heat – sprayed onto a rolled-up transparency film (preferably Kapton film). The painted-on gradient filter yields near-uniform light meter readings of 8.0, 7.9, 7.7, and 7.4 over the 24” width board face with a 300W bulb.

Basic Defect Properties

How do humans find defects? Squint your eyes and look at a wood board. You will observe that knots are darker in intensity than grain and grain is darker in intensity than clear wood. Humans say that knot wood is darker than clear wood. More specifically, knot wood is darker in contrast than clear wood. Humans use color to find defects; other mechanisms are used to identify suspect colors as knot defects.

Finding defects is, moreover, a global process. If the viewpoint is too local, as in a pixel-by-pixel search, identifying knot colored wood must rely on comparisons with a color model. If the viewpoint is global, wood contains enough clear pixels to constitute a Normal population to develop statistics on contrast mechanics.

By observing the images of several dozen sample boards, a basic 3-step heuristic is developed for identifying color defect properties. This heuristic works regardless of hue drift* of up to 5 points to find knot defect colors sampled in sapwood. Defect features such as splits, voids, knots, rot, blue stain, and so on must be identified using pattern recognition. The heuristic was developed given 5-bits per red, green, and blue color channel and is described as follows:

- 1) IF blue is brighter than average+1 std. dev., THEN wood is clear;
- 2) IF red is darker than average-1 std.dev., THEN wood is knot;
- 3) ELSE wood is grain.

SYSTEM TOPOLOGY

The placement of the Image Processor scanner is immediately upstream of the current manual shadow line station. The shadow line station is where rip saw operators enter manual solutions to control the rip saw networks. Boards will be singulated at the breakdown table and conveyed sequentially into the board scanner. The breakdown table and board scanner are shown in Figure 1. The board scanner frame is approximately

* Camera lens f-stop adjustment is necessary to maintain average image intensity.

19 feet long, 6 feet wide, and 6 feet tall. The breakdown table and conveyor chain height are a minimum of 32 inches from the floor.

A set of pinstops inside the board scanner frame stops each board underneath the halogen lights and cameras for 2.3 seconds while a total of 32 images are collected. The board is then immediately sequenced to the shadow line station and a second board is stopped inside the board scanner. During this sequencing action, the board images are processed by the Acumen 900 Image Processing computers and the Board Value Optimizer personal computer. After approximately 3.5 seconds from the image acquisition, the rip solution of the first board is displayed to the operator. If necessary, the operator has the option to change the rip solution sent to the rip saw networks at the manual shadow line station. Board processing continues automatically in this fashion until stopped by the rip saw operator.

IMAGE PROCESSING

The Image Processor performs five primary functions. These include board scanning, board edge measurement, edge waste measurement, defect location, and edge cut determination. Once these functions have been completed, the Image Processor passes board data along to the Master Processor for rip solution optimization. In addition, the Image Processor performs a calibration function to maintain accuracy. The following sections describe each of these functions in more detail.

Board Scanning

Each board is scanned by a total of sixteen color video cameras, eight on the top half and eight on the bottom half of the Board Scanner. Each camera captures two separate images of the board so that the board width and wane edge can be measured. The board is sequentially illuminated from the leading and trailing sides of the width. The lights are positioned at a low angle to cast heavy shadows on wane edges. Figure 10 illustrated the concept of oblique lighting. The scanner frame is painted flat black so that the board face is the only “bright” object in the field of view.

Board Edge Measurement

When all the camera images are in video memory, the first processing task is to locate the maximum board edges to build a wire-frame profile of the board. The board outline is overlaid on the actual video image of the board as shown in Figure 13. Note that to achieve maximum processing throughput, such display tasks occur after all image processing tasks are completed and the board model is sent to the Board Value Optimizer.

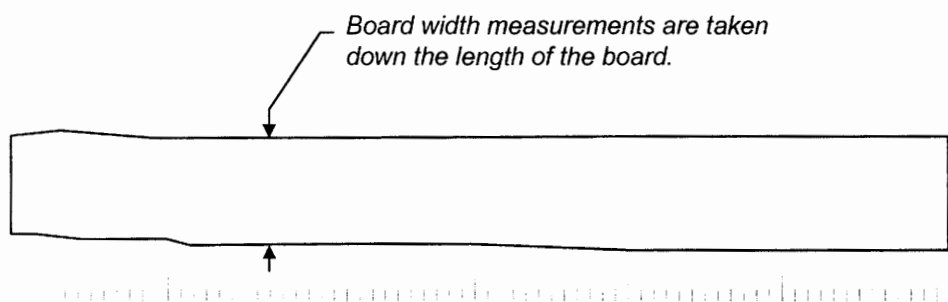


Figure 13. The first image processing step is to determine the board profile by measuring the outside edges.

Width measurements are taken on one inch increments using the contrast between the bright board face and the dark scanner frame interior from the leading and trailing-lit images. The intersection of light and dark pixels marks the edge of the board. The interface pixels are mapped to exact x & y-axis Cartesian coordinates. The maximum board edge coordinates are saved in a data table for later use.

Edge Wane Measurement

The next step of image processing is to identify edge wane or bark edge. This operation is performed in the same way as determining the maximum board edges. However, wane is a defect, and wane edge measurements differing more than $\frac{3}{8}$ inches from the maximum edge are saved in a separate list of defects. The wane edge defects are overlaid on the actual video image of the board as shown in Figure 14.

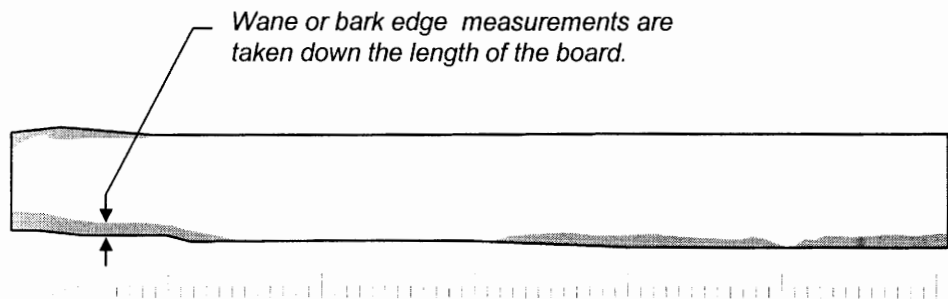


Figure 14. The second image processing step is to determine wane defects using contrast measurements obtained from the first step.

Edge Cut Determination

The final vision processing task is to determine the inside and outside edge cuts. An edge cut (called “hogging”) uses a large router to chip off wane edges. The circular

saw blades of a rip saw are not designed to clean up wane edges because the wood is not thick enough and causes the blades to twist. The minimum edge cut is the distance from the maximum edge of the board, at which point the saw blade cuts the full thickness of the board over most of the board length.

Edge cuts are determined from profile and defect measurements. The edge cuts determine the usable board width, and thus constrains the Board Value Optimizer's rip solutions. The edge cuts are overlaid on the enhanced board image as dashed "clean-up lines" as shown in Figure 15. Edge cuts are added to the list of profile measurements.

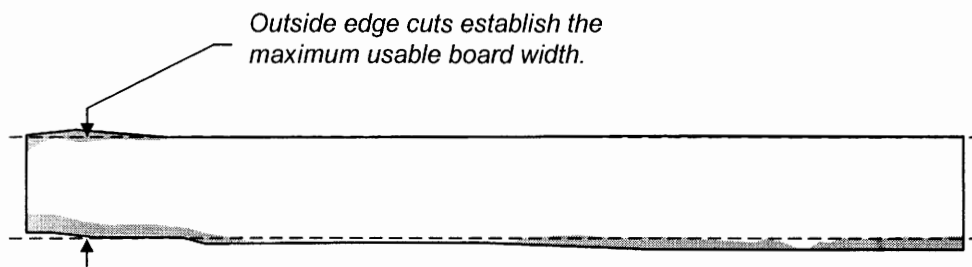


Figure 15. The third step in image processing is to determine the inside and outside edge cuts.

Color Correlation

The third image processing step is to enhance and normalize the color images of the board. This step converts the red, green, and blue pixels of both leading-lit color images into a single grayscale converted image. The enhanced image replaces the actual video image of the board as shown in Figure 16.

This operation uses dynamically adaptive color correlation algorithms developed by the author (see Chapter III) to determine *clear* and *clear-grain* color pixels for

identifying *non-defect clear wood*, thereby separating out the regions with a higher likelihood of *grain-knot* and *knot* color pixels. The color correlation algorithms determine clear-grain and clear color for each board image and are thus insensitive to camera drift and average color model limitations.

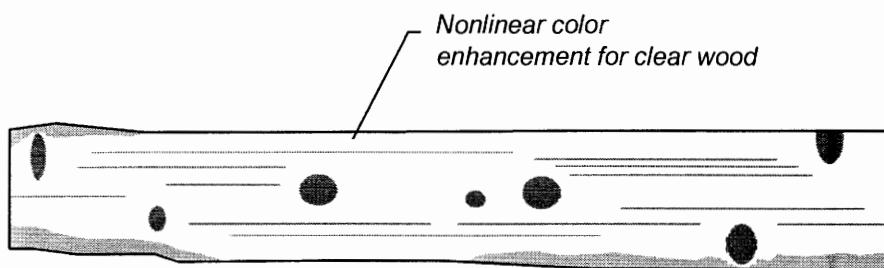


Figure 16. The fourth image processing step is to enhance the board image by the adaptive color correlation method.

Defect Location

The next image processing step is to identify and bound the minimum and maximum coordinates of knots and other similar defect features of the *not-clear* areas. Identification is performed by pattern matching a shape model on the enhanced image. The minimum and maximum x- and y-axis coordinated, called a bounding box, are determined by the geometry of the shape model. The bounding box is added to the list of defects, and the boxes are overlaid on the enhanced board image as shown in Figure 17. Together, the profile measurements and defect list comprise the board model. The board model is sent to the Board Value Optimizer to determine the rip solution.

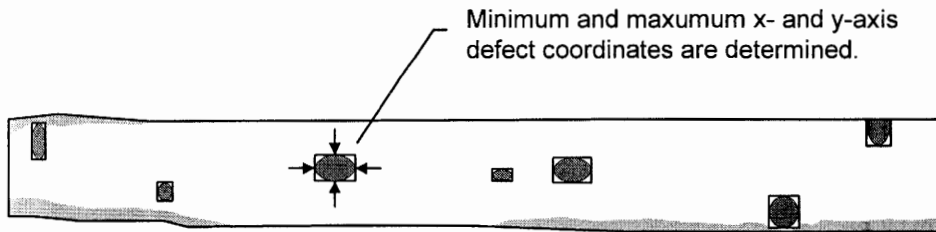


Figure 17. The fifth image processing step uses pattern recognition to bound knot defect features with rectangular boxes.

Image Processor Calibration

Recovering the most value from the rip operation requires the Image Processor to perform with consistent accuracy. Proper image alignment is required to overcome effects of camera lens barrel distortion, mechanical stresses, temperature changes, and other factors for pixel to Cartesian coordinate mapping. The Image Processor is calibrated (on a regular basis) using a reference board. The board is of known width and length and contains artificial defects at known coordinates. When the board images are processed, the board model is calibrated to the reference board. Subsequent processed images conform to the calibration.

CHAPTER III

COLOR CORRELATION METHOD

The techniques of finding knot defect features in wood using computer-based image processing entails two distinct operations. First, groups of neighboring pixels are detected based on color properties. An individual pixel is a candidate for inclusion if its color is close to the color of knot defects features. This operation is called color classification. Second, the groups of neighboring pixels are analyzed to determine if the gross shape resembles knot defect features. This operation is called pattern recognition.

This chapter develops the author's adaptive color correlation method used to classify individual pixels of a color image of wood into non-defect clear and clear-grain colored wood and defect grain-knot and knot colored wood. A section on color correlation introduces three classifications of wood color and presents the fundamental principles of wood color classification. A section on histogram modes presents a method for finding classification statistics in wood images. A section on filter design and implementation describes how to classify knot features in wood images. In the final section, the results of implementing the adaptive color correlation method in the Machine Vision System at a wood products manufacturing plant are presented.

COLOR CLASSIFICATION

Wood color is generally classified into three classes of “clear,” “grain,” and “knot” color. These three classes arise naturally from the properties of exogen wood fibers. In exogen softwood, wood grows in layers on the outside of the tree. Two distinct layers of springwood and summerwood are grown each season. Summerwood comprises the well known darker “growth rings” seen in tree cross-sections. Summerwood is denser than springwood and comprises *grain* wood colors. Springwood is less dense than summerwood and comprises *clear* wood colors.

Summerwood and springwood are further divided into two long-term growth regions called sapwood and heartwood. Sapwood is less dense than heartwood and overall lighter in color. Both clear and grain wood colors are consistent within each region with respect to each other. The heartwood region is moreover resin stained. Figure 18 shows an example of heartwood in the darker center region and sapwood in the outside regions of a typical Ponderosa Pine wood board grown in Western Oregon.



Figure 18. Black & White image of a typical ponderosa pine wood board.

Knots exist throughout summerwood and springwood. A knot is part of a branch that once carried resin from the sapwood to the leaves and back again. Around knots, the wood is cross grained, i.e., the summerwood and springwood grow around the branch. Branches contain both springwood and summerwood and are sometimes stained by resin into very dark colors. In general, branches are denser than the springwood and sapwood of the tree.

The color of knot wood is darker than the color of grain wood, and grain wood is darker than the color of clear wood. Interestingly, the hue of knot, grain, and clear wood can be very similar. Most important, the contrast relationships between knot, grain, and clear wood is the same for all species of pine exogen sapwood trees.

Wood Color Sets

The universe of all possible wood color is usually considered to comprise four sets: clear, grain, knot, and everything else (e.g., rot, pitch, blue stain fungus, and so on). The left block of Figure 19 conceptualizes this set paradigm. Clear wood is bright in intensity, and knots are dark in intensity. The majority of wood pixels in an image belong to the clear set. The minority of wood pixels in an image belong to the knot set. A third set called grain overlaps the clear and knot sets as well as containing certain intermediate intensities. The remaining intensities are miscellaneous colors found in nature and manufacturing processes (e.g., burn, grease, paint, and so on).

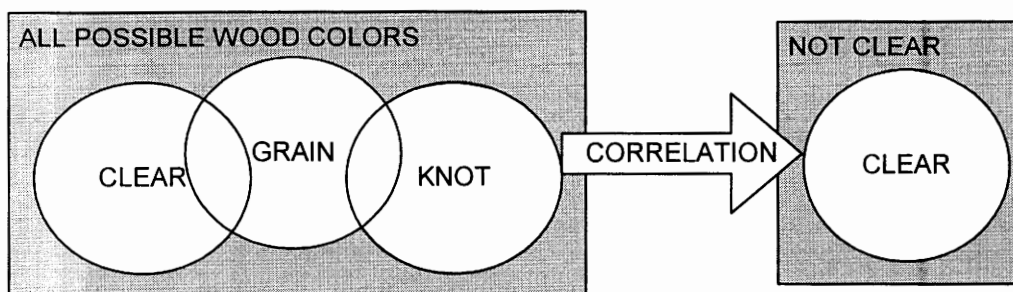


Figure 19. Color correlation conceptualized as transforming the universe of all possible wood colors from having 4 categories to having only 2 categories.

Classifying RGB Colors

Color correlation image processing methods use the distance from a known or model color (fixed point or region) in 3-dimension color space (red, green, and blue vectors) to classify the likelihood that a test color is represented by the model color. If the distance is below a threshold, the test color is classified. Because the color properties of wood change from tree to tree and color detecting equipment introduces color-change anomalies such as random noise and color balance drift, prior research has focused on increasing the robustness of color model classification methods.

Research performed to increase the classification robustness through color space transforms such as RGB-to-HSI (hue, saturation, and intensity) has shown no improvement [19]. Research performed to increase the classification robustness through “extended-dimension color” spaces (i.e., adding reflectance measurements to produce a 4-dimensional classification space) has shown only marginal improvements for the additional data gathering and manipulation complexity [20]. Research performed to

increase the classification robustness by changing the shape of the known color model (i.e., cone shaped regions and multiple neighboring “potato-shaped” color models) has shown some improvement when the color change is predictable [21].

Research performed to increase the classification robustness through an analysis of neighboring pixels has shown significant improvement. Color space models that compare a test pixel’s color to neighboring test pixel’s colors are capable of classifying *darker* not-clear wood colors from *brighter* clear wood colors [22][23]. The terms “brighter” and “darker” refer to a pixel’s color intensity with respect to neighboring pixels. If the distance from the test pixel’s color to the average neighboring test pixel’s colors is greater than a threshold, the test color is classified.

Experiments performed by this author indicate a significant advantage to performing neighbor-based classification using color information as opposed to black & white intensity information. Further, the aforementioned relationships between clear, grain, and knot colors of wood are exemplified in each of the three primary color channels. From this new paradigm, four general rules are developed for each of the color channels to classify an unknown wood color into either the *clear* or *not clear* sets based on the color of neighboring pixels, as pictured in the right of Figure 19 and as follows:

- 1) IF *blue* is *bright*, THEN wood is *clear*;
- 2) IF *red* is *dark*, THEN wood is *not clear*;
- 3) IF *red+green+blue* is *bright*, THEN wood is *clear*;
- 4) IF *red+green+blue* is *dark*, THEN wood is *not clear*.

For the above color correlation rules, the terms *bright* and *dark* refer to the average value of neighboring pixels. For an individual color channel, a “universe of

discourse” encompasses all possible values of intensity, e.g., 0 to 31 for 5-bit pixels.

Pixels classified as *not clear* are candidate pixels for defect features and are later processed using shape correlation algorithms. Although pixel classification is binary, the shape correlation algorithms operate over a large group of pixels and thus reduce or smooth miss-classification errors.

Grayscale Classification Method

Using the average color value of neighbor pixels and the aforementioned classification rules overcomes the limitations of using a fixed color model. However, binary IF-THEN classification is a poor representation of how well a test pixel is classified with respect to its neighbors. Binary image classification degrades the overall pattern recognition process because shape correlation is a mathematical (grayscale) equation. A better classification strategy is to describe the classification result in terms of the probability that a test pixel belongs to a specific set. One approach is to describe the aforementioned classification rules using certain kinds of mathematical equations.

When an IF-THEN rule is described via such equations, the resultant classification is a numerical value from zero to 1.0, and classified pixels may have dual (partial) membership in both *not clear* and *clear* sets, respectively. Subsequent shape correlation compares the classified pixel to a model pixel representing full membership in a specific set. This two-step process yields a second (grayscale) value representing the membership or likelihood a group of test pixels are classified to the *not clear* color and to a model defect feature shape. When this group value is above a threshold, the area is

labeled a defect feature. The minimum and maximum coordinates of the defect feature are determined from the model defect feature shape. The model defect feature shape is confidential to the manufacturing plant and is not included in this thesis.

A piece-wise non-linear color correlation equation is developed from concepts used in Fuzzy Logic mathematics. Fuzzy Logic mathematics combine IF-THEN rules with linear equations ($y = mx + b$) to produce non-linear results. Experiments performed by this author indicate that non-linear color classification strategies are shown to outperform linear color classification strategies for wood images. (See bibliography for more background material on Fuzzy Logic mathematics.) Four general rules are developed for each of the color channels to classify an unknown wood color into either the *clear* or *not clear* sets, as pictured in the right of Figure 19 and as follows:

- 0) $P_C = 0$
- 1) IF $T_B > A_B$ THEN $P_C += \frac{1}{4} (T_B - A_B) / (SAT - A_B)$
- 2) IF $T_R < A_R$ THEN $P_C += \frac{1}{4} (A_R - T_R) / (A_R)$
- 3) IF $T_{RGB} > A_{RGB}$ THEN $P_C += \frac{1}{4} (T_{RGB} - A_{RGB}) / (SAT - A_{RGB})$
- 4) IF $T_{RGB} < A_{RGB}$ THEN $P_C += \frac{1}{4} (A_{RGB} - T_{RGB}) / (A_{RGB})$

Where T_B is the intensity of the test pixel's blue channel, A_B is the average intensity of neighbor pixels, SAT is the saturation or maximum pixel value, and P_C is the likelihood that the test pixel belongs to either the not clear ($P_C = 0.0$) or clear ($P_C = 1.0$) sets. The subscripts X_R and X_{RGB} correspond to the red channel and the sum of the red, green, and blue channels, respectively. On evaluating each equation, the P_C likelihood is summed and carried to the next equation, and the total resultant value is used for pixel classification.

The number of neighbor pixels to sample and method of sampling neighbor pixels depends greatly on the lighting method used. Experiments performed by this author indicate that both local and global neighbor averages must be used. Because a single board may contain both sapwood and heartwood, local neighbor pixels must contribute strongly to the average color. Halogen lights provide very little blue color, for example; so, many pixels must be sampled to accurately define the blue color neighbor average.

SIGMOID-FILTER PROCEDURE

In practice, the average color of wood is a poor metric for classifying test pixels into either *clear* or *not clear* sets. Because of camera white-balance drift, variation in lighting, and the natural variation in the color of wood, a method must be used that adapts the color correlation algorithms to re-learn wood color characteristics for every board image. This section covers the development and implementation of sigmoid shaped color channel filters to classify individual pixels in wood images.

Histogram Analysis

After a board is imaged into video memory, a histogram is generated for each color channel. The histograms are then analyzed for three peaks and assigned to the categories: 1) knot; 2) grain; and 3) clear. Figure 20 and Figure 21 are histograms of the red color channel for two ponderosa pine wood samples grown in different climates. Note that each histogram contains 31 bins corresponding to 5-bit color channel resolution of the Acumen 900 Image Processor.

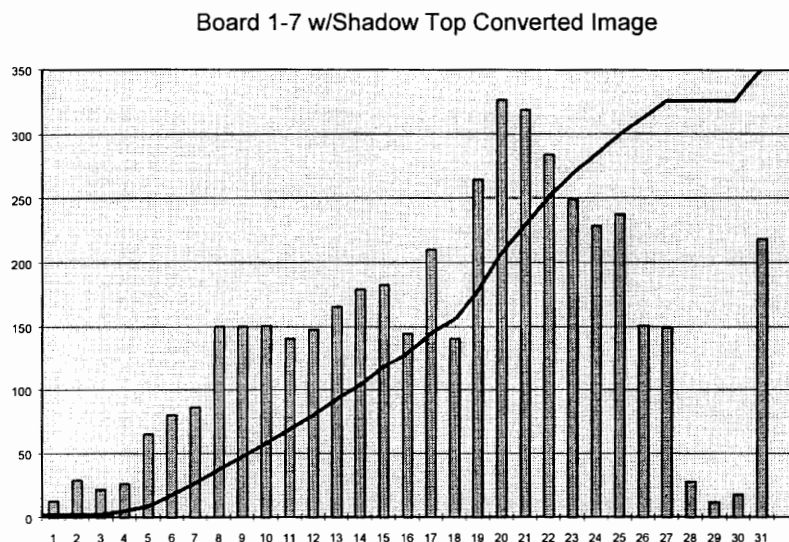


Figure 20. Histogram and cumulative histogram of the red color channel of an image of ponderosa pine grown in Western Oregon.

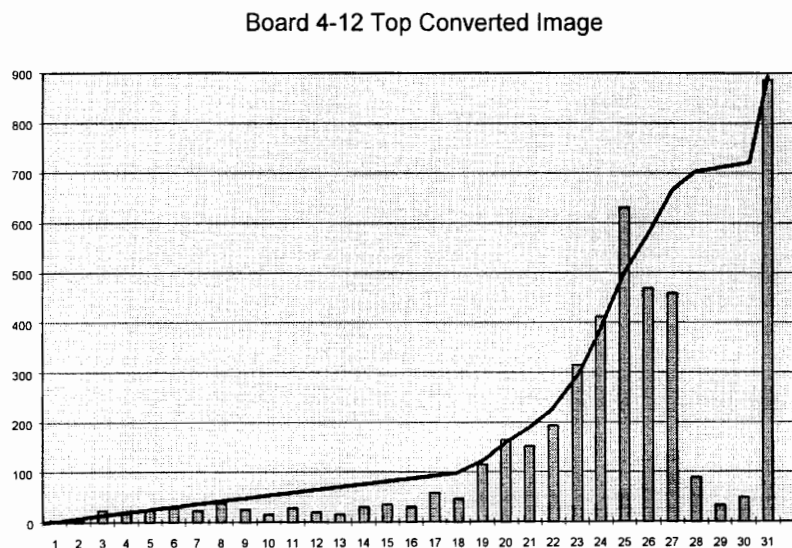


Figure 21. Histogram and cumulative histogram of the red color channel of an image of ponderosa pine grown in Southern Oregon.

By intent, the red color channel exhibits saturation, and histogram bin 31 counts actual pixel intensities greater than 31 (5-bits). This saturation effect happens because the

image intensity bandwidth is greater than the camera output bandwidth in the red channel. Charge-coupled device (CCD) cameras used to image wood are not as sensitive to light in the green and blue color channels as they are in the red color channel. Further, halogen lamps produce less blue light than red and green light. So, the camera gain was increased to somewhat normalize the information content contained in each color channel. This saturation effect mimics an analog clipping circuit and is later used as a pre-processing step in the sigmoid shaped filters.

Analyzing the histogram and cumulative histogram of Figure 20, from a rich yellow colored board having moderate grain and 3 knots, three peaks are present at about 9, 20, and 31. These peaks are assigned to knot, grain, and clear wood colors for this board. Each peak describes an almost Normal bell-shaped mode. Overlap should be visualized between the knot and grain modes and the grain and knot modes.

Analyzing the histogram and cumulative histogram of Figure 21, from a light yellow colored board having only grain and no knots, three peaks are present at about 8, 25, and 31. These peaks are assigned to knot, grain, and clear wood colors for this board. Because the board contains few knot colored pixels, confusion can arise between the grain and knot peaks.

Non-Linear Filters

Non-Linear filters are introduced here to allow classifying colors using look-up tables or LUTs. A LUT stores the input-to-output pairs of pre-calculated mathematical

equations, such as a non-linear filter, in memory. In computer operations, a LUT is a very fast and efficient operation.

Underlying all histograms of pine wood images is the observation that a Normal-shaped distribution of grain colored pixels is identifiable. Because grain color also contains clear and knot colors, it is not appropriate to classify grain into just one of the clear or knot sets. Rather, grain color is better used as a natural boundary between the knot and clear sets.

The reason for development of the non-linear filters is as follows. Because clear wood colors are generally bright in intensity, i.e., *above average* intensity, pixels of intensities above that of the average grain intensity will be assigned (mapped) to values approaching saturation (31 in 5-bit resolution). Because knot wood colors are generally dark in intensity, i.e., *below average* intensity, pixels of intensities less than the average grain intensity will be mapped to values approaching zero. The intensities which are approximately average will correspond to the transition region. These considerations yield a nonlinear filter function as shown in Figure 22. The mathematical “sigmoid” function is used here to fill this role.

Figure 23 illustrates the process developed for classifying pixels of a wood image. The upper middle x-y plot illustrates a sigmoid filter. The lower axis represents the actual measured intensities and the vertical axis represents the “assigned” values for each pixel. Using the sigmoid filter, pixel values less than the midpoint of the sigmoid plot are mapped to lower values corresponding to knot defect features. Pixel values greater than the midpoint of the sigmoid plot are mapped to higher values corresponding to non-defect

features. Pixel values near the midpoint remain unchanged and may either contribute to or negate neighboring pixels tested in the shape correlation process.

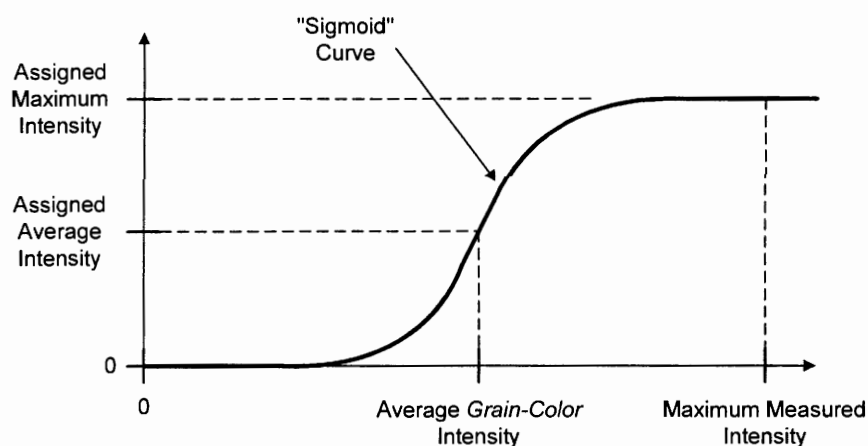


Figure 22. The mathematical "sigmoid" function.

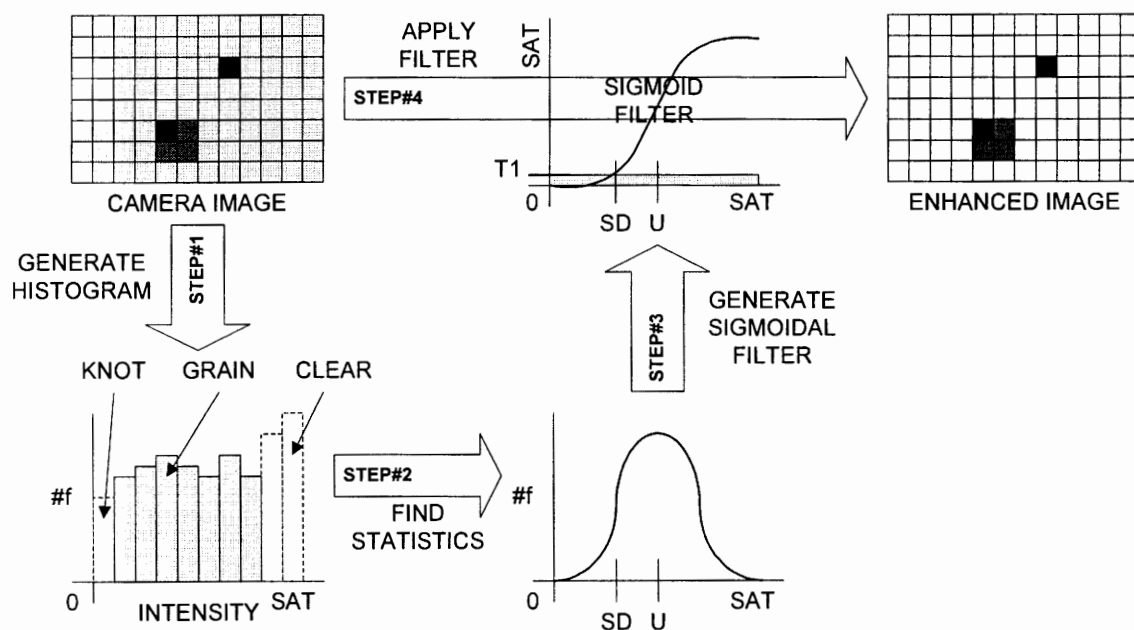


Figure 23. Image enhancement using the adaptive color correlation filter.

Image enhancement begins, as diagrammed in Figure 23, by developing a histogram for each color of the camera image of the identified wood workpiece. Because bright intensity pixels with values of 31 (saturation), 30, 29, etc. (for 5-bit resolution) directly correspond to clear colored wood, these upper-bins of the histogram are not included in generating statistics for grain colored wood. Similarly, dark pixels with values of 0, 1, 2, etc. directly correspond to knot colored wood or other not clear defect colors and are also not included in generating statistics for grain colored wood. The mean and standard deviation are calculated for the remaining histogram data.

Because the sample size used to generate the histogram is large (as many as 245,760 samples), the mean and standard deviation are used directly for the sigmoid function. The equation for the sigmoid function, where μ is the mean, σ is the standard deviation, and (sat) is the normalizing saturation value is as follows:

$$f(x) = \frac{SAT}{1 + e^{-\left[\frac{x-\mu}{2\sigma^2}\right]}}$$

Step 4 of Figure 22 concludes the image enhancement by applying the sigmoid filter to each pixel in the image. The sigmoid filter is the equivalent of a cumulative probability distribution of the normal “bell-shaped” distribution. The sigmoid filter is centered about the mean and normalized to saturation, and the slope decreases as the standard deviation increases. Given Normal population statistics, the threshold value, T1, is selected as one standard deviation below the mean to contain fewer than 16% of candidate knot-grain pixels for subsequent image processing tasks.

Figure 24 shows the red color channel image of the ponderosa pine board shown in Figure 18. Figure 25 illustrates the result of sigmoid-filtering the red channel image of Figure 24 using the above 4-step procedure. The enhanced dark pixels are parts of knot defect features and grain located in heartwood regions. Figure 26 is the result of summing and normalizing the sigmoid-filtered red, green, and blue channel images of Figure 18 using the above 4-step procedure.



Figure 24. Red channel image of a typical ponderosa pine wood board.



Figure 25. Image produced by sigmoid-filtering procedure of the red channel image of Figure 24.

Because of variations in lighting and general variations in wood colors, it is desirable to sample and apply the color correlation filter to small areas of wood. In practice, developing histograms along the x and y axis of a wood image and using moving average principles provides the best overall mix of local and global information.

Appendix A contains C code that color correlates a 24 inch wide by 28 inch long image (520 by 640 pixels) using a grid of 64, 0.4 inch histograms in the width direction by 80, 0.4 inch histograms in the length direction. Appendix B presents eighteen images of wood defect samples that were color correlated using the algorithms of Appendix A.



Figure 26. Image produced by summing and normalizing the resultant images produced by sigmoid-filtering the red, green, and blue channel images of Figure 18.

Filter Algorithms

Figure 27 illustrates a block diagram of the afore-described adaptive color correlation process. The process begins from the left with the video memory containing a wood image. As pixels flow from memory, the individual red, green, and blue colors are

stripped from the pixel. The histogram and statistics are calculated to enhance the knot feature defects. Both the statistics and pixel color value are used in generating (actually selecting from memory) the sigmoid filter. The results from each filter are combined into a single enhanced image.

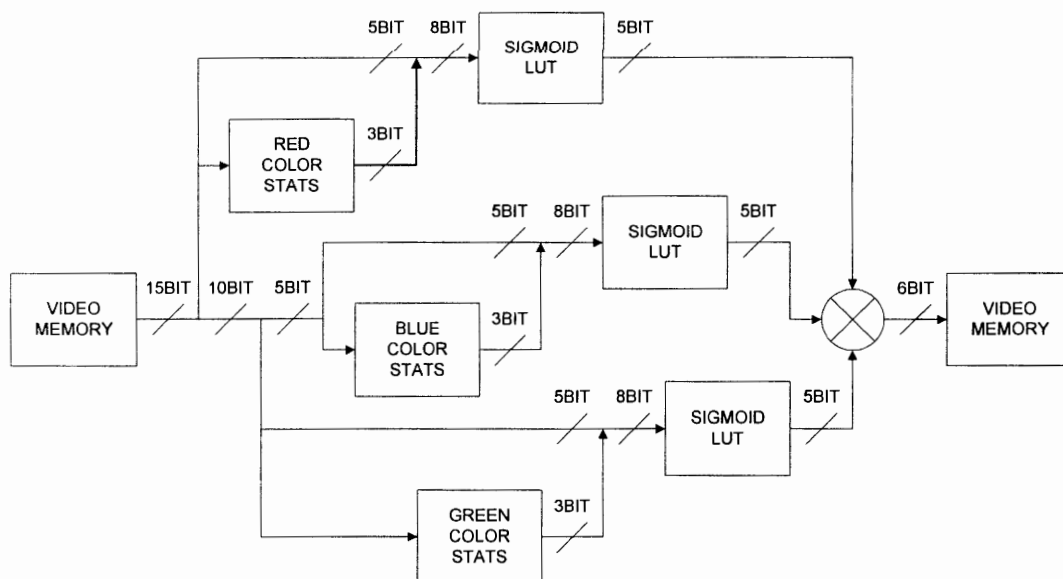


Figure 27. Block diagram of the adaptive color correlation process using one sigmoid filter per color channel.

Note that the sigmoid filters are pre-calculated and stored in look-up tables (LUTs). A LUT is used to decrease processing time of often-repeated complex mathematical calculations. Given a 5-bit image and a 3-bit LUT selector, an 8-bit LUT can map nine (3-bit) different generalized sigmoid functions. Figure 28 shows how the 3-bit LUT selector is combined with the 5-bit pixel data using a logical OR operation. Figure 29 graphically shows a family of nine sigmoid functions used in Figure 27 and the aforementioned 4-step procedure. Sigmoid #1 is selected (000XXXXX) when the average grain-color is dark, and sigmoid #9 is selected (111XXXXX) when the average

grain color is bright. The slope of the sigmoid function, determined by the standard deviation, is based on the average standard deviation for several sample boards. This “averaged” approach was found to be satisfactory in practice given the 5-bit resolution of the color image data.

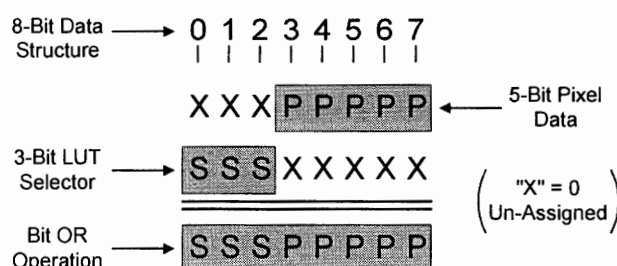


Figure 28. Bit-Wise OR operation combining 5-bit pixel data with 3-bit LUT selector value.

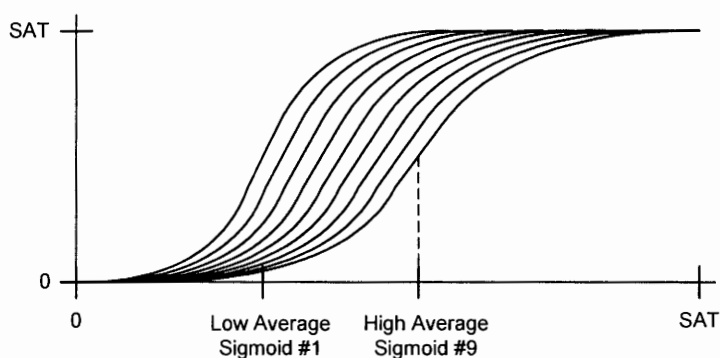


Figure 29. Family of 9 sigmoid functions stored in an 8-bit LUT.

Figure 30 illustrates two main loops used to implement the moving average principle of image enhancement. Figure 31 and Figure 32 illustrate several loops used in pattern recognition algorithms. Figure 31 begins the pattern recognition process with pixels from the enhanced image. Figure 32 details the shape correlation process.

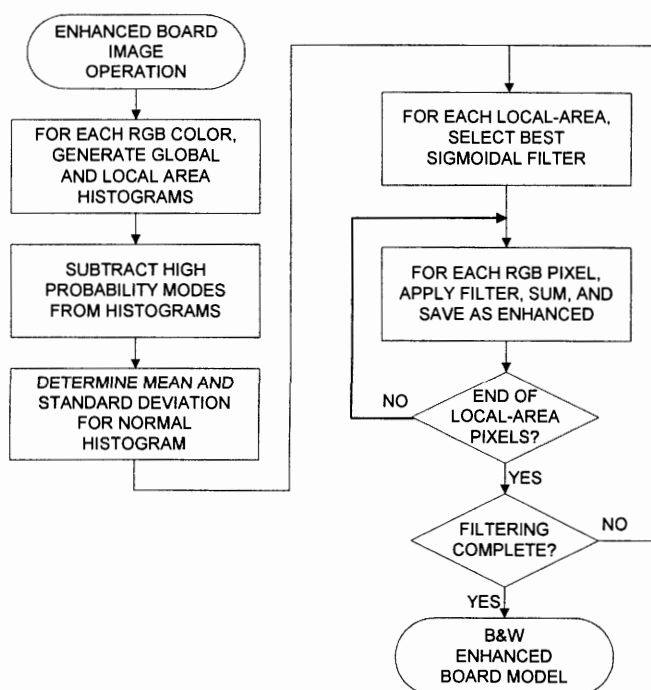


Figure 30. Flowchart of the adaptive color correlation algorithms implementing a grid of neighboring histograms and LUTs.

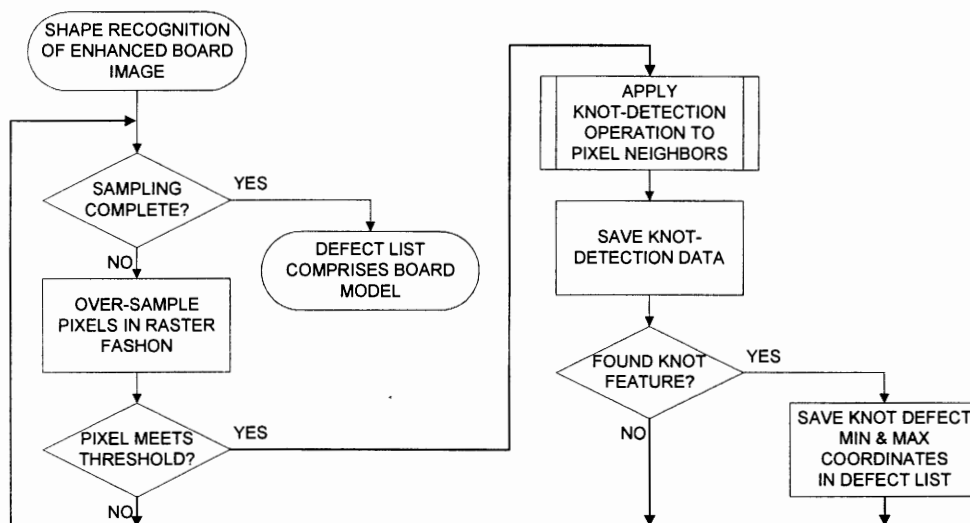


Figure 31. Flowchart of the shape recognition process using the enhanced images produced by adaptive color correlation; the “knot-detection operation” is detailed in Figure 32.

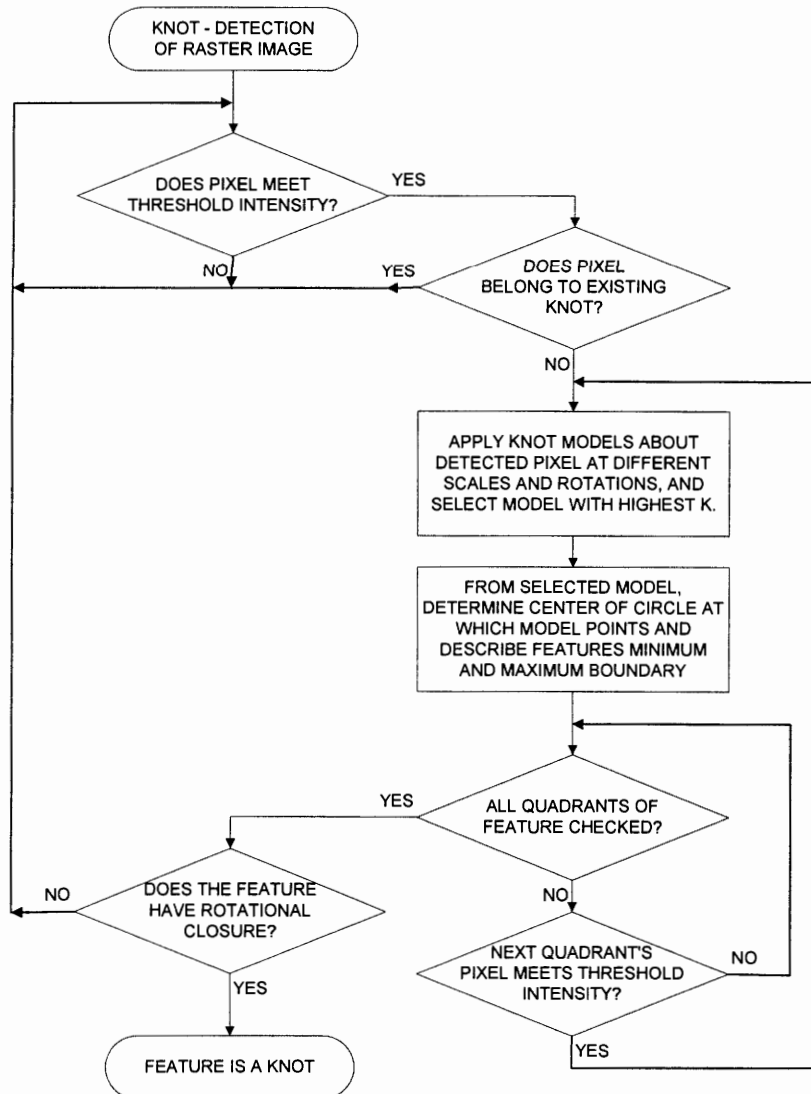


Figure 32. Flowchart of the knot feature identification process; this flowchart details the “knot-detection operation” block of Figure 31.

The flowcharts of Figure 31 and Figure 32 are included for interest. A complete discussion of shape-based pattern recognition is beyond the scope of this thesis. However, it is important to note that the aforementioned adaptive color correlation method significantly improves processing throughput. Shape-based pattern recognition is

an area-based process that samples the image in nine pixel (3 x 3) blocks for very simple patterns such as edges, splits, and corners. The Acumen 900 image processing hardware is designed to sample the image in 4096 pixel (64 x 64) blocks to identify very complex features such as knots and other non-symmetric features. Because wood defect features are random shapes, the shape model used to identify defect features must be translated in the x and y-axis, rotated, and scaled in size as it is applied to the image. Taken together, translating, rotating, and scaling a shape model requires a huge amount of computation. So, reducing the number of defect feature candidates to be analyzed makes it possible to apply multiple translations, rotations, and model sizes to each defect feature candidate and still achieve the desired (improved) throughput.

TESTING CLASSIFICATION

In a 94 board test of the Machine Vision System, the Image Processor subsystem was observed to yield 96.38% correct bounding of the visible top knots (852 out of 884). Given a 3% estimated loss of bottom knots due to chain races, the Image Processor yielded 94.88% bounding for both top and bottom sides. In addition, the Image Processor falsely bounded 77 closed grain features. Forty-six of the 77 falsely bounded features occurred at the board edges (top cameras 1 and 8) which are subject to poor lighting conditions due to enclosure constraints. The Image Processor also labeled 8 inch and shorter patches of medium to dark blue stain a defect feature.

In optimizing the adaptive color correlation algorithms, a trade-off for improved accuracy was made for a substantial reduction in speed. From the test data and visual

observations of the Image Processor video screen, the 5-bit resolution (32 steps) of the Acumen 900 vision processor is just adequate. A 6-bit resolution (64 steps) would be much better.

A global orientation of false grain was observed along the core of some boards. It should be possible to edit shape/position-based low probability-of-model-matches. However, the rip saw operators noted that the system was producing good, tight core rips heavy in knots. One operator commented that the system consistently produced 1-7/8" core rips where he would normally stick with 2-1/4" or wider rips. Using the narrower 1-7/8" rip width is better because wider rips can possibly be taken elsewhere. This also suggests that some false-grain errors are "good" because such defects reduce the value of the core area and cause the core rip to be taken.

The Image Processor edge cleanup was established by trial-and-error methods following the goal of maximizing the usable board width. The results of processing an entire lift of 2-shop did not produce edge cleanup errors nor did the operators reject any rip solutions for edge cleanup reasons.

Several instances of miss-labeling heavy grain knot defect features are noted in Table I and Table II. At this time, such miss-labeling is thought to be resolvable through additional rules in the pattern recognition algorithms. If this is not the case, the knot feature threshold will be adjusted for tighter correlation values. Table III and Table IV show that pixels are robustly classified as knot defect features as the area is correctly bounded or over-bounded to include some cross-grain.

TABLE I
DEFECTS MISSED BY TOP CAMERAS FOR PINE WOOD

	Board 1 %Miss	Board 2 %Miss	Board 3 %Miss	Board 4 %Miss	Board 5 %Miss	Board 6 %Miss	Total %Miss
Camera 1	0	0	0	0	0	0	0
Camera 2	0	0	0	0	0	0	0
Camera 3	0	0	0	15% [*]	0	0	2.5%
Camera 4	0	0	0	0	0	15%	2.5%
Camera 5	0	0	0	0	0	33%	5.5%
Camera 6	0	0	0	0	0	0	0
Camera 7	0	0	0	0	0	0	0
Camera 8	0	0	0	100% [†]	0	0	0
Total	0	0	0	2.1%	0	6%	1.3%

TABLE II
DEFECTS MISSED BY BOTTOM CAMERAS FOR PINE WOOD

	Board 1 %Miss	Board 2 %Miss	Board 3 %Miss	Board 4 %Miss	Board 5 %Miss	Board 6 %Miss	Total %Miss
Camera 1	20%	N/A [‡]	N/A [*]	50%	N/A [*]	N/A [*]	23.3%
Camera 3	N/A [*]	N/A [*]	15%	N/A [*]	N/A [*]	0	7.5%
Camera 5	N/A [*]	N/A [*]	0	N/A [*]	0	N/A [*]	0
Camera 7	0	0	0	0	0	0	0
Total	10%	0	5%	25%	0	0	7.1%

^{*} Missed Pitch Pocket.

[†] No Leading or Trailing Edge Found.

[‡] Defect covered by chain raceway and/or shadow.

TABLE III
AREA BOUNDED BY TOP CAMERAS FOR PINE WOOD

	Board 1 % Area	Board 2 % Area	Board 3 % Area	Board 4 % Area	Board 5 % Area	Board 6 % Area	Total % Area
Camera 1	100%	200%*	120%	100%	100%	100%	120%
Camera 2	95%	100%	110%	105%	100%	100%	102%
Camera 3	100%	130%	120%	85%	100%	200%	123%
Camera 4	105%	100%	130%	100%	100%	85%	103%
Camera 5	100%	400%†	110%	230%	120%	67%	171%
Camera 6	100%	100%	120%	150%	100%	200%	128%
Camera 7	100%	200%	110%	100%	100%	110%	118%
Camera 8	100%	200%	100%	N/A‡	100%	100%	117%
Total	100%	179%	115%	121%	103%	120%	123%

TABLE IV
AREA BOUNDED BY BOTTOM CAMERAS FOR PINE WOOD

	Board 1 % Area	Board 2 % Area	Board 3 % Area	Board 4 % Area	Board 5 % Area	Board 6 % Area	Total % Area
Camera 1	80%	N/A§	N/A*	50%	100%	N/A*	76.7%
Camera 3	N/A*	N/A*	85%	N/A*	N/A*	100%	92.5%
Camera 5	N/A*	N/A*	100%	N/A*	100%	N/A*	100%
Camera 7	100%	100%	200%	100%	95%	100%	116%
Total	90%	100%	128%	75%	98.3%	100%	97.6%

* Grouping error.

† Heavy grain labeled knot.

‡ No defects found.

§ No defects found.

CHAPTER IV

RIP SOLUTION OPTIMIZATION

When the cost of wood was cheap relative to other manufacturing costs, nobody worried about optimizing the way wood boards were cut into products. Any wood board was used for whatever wood product was desired regardless of minimizing waste or maximizing value recovery. As wood has become more expensive relative to manufacturing costs (which have also increased), more efficient methods of maximizing the recovery of clear wood in every wood board are sought. Because every uncut wood board has a certain dollar value based on its grade, maximizing the recovery of clear wood as it is cut up also minimizes waste and maximizes recovered value (or profit).

A study by Michigan State University suggests that experienced rip saw operators do a relatively good job — 3% less yield than “pencil cutting” — at determining rip solutions for a rip-then-crosscut sawmill operation [5]. However, rip saw operators are people, too. People have “bad days,” people are sometimes sick, and for whatever other reason, people do not always produce the same quality of work day in and day out. The purpose of installing a Machine Vision System to automate the rip saw operation in a sawmill is to improve the consistency and quality of optimal and near-optimal solutions.

This chapter introduces rip solution optimization methods and the overall manufacturing requirements for a Machine Vision System used to automate a rip saw in a

rip and cut sawmill operation. A rip solution is a selection of rip-widths which a rip saw operator enters into a rip saw networks controller, and the rip solution is used to set the spacing of blades in a movable-blade rip saw. Typically, rip saws have between 5 and 8 movable blades that produce from 1 to 10 rip-width products when ripping a single board that may be up to 26 inches wide and 28 feet long.

BACKGROUND

Generally speaking, there are three different optimization approaches to determining a rip solution. The first optimization approach is to select rip-widths that produce the maximum amount of clear wood in each rip-width, and the widest rip-widths are maximized first. The crosscut saw operator later determines the actual products cut from the rip-widths. The second optimization approach is to place actual cut products into the rip-widths, determine the actual value of the rip solution, and select the highest value solution. The third optimization approach is a modification of the first two approaches where certain rip-widths are selectively de-emphasized by (mathematical) value reduction based on work-in-progress (WIP) inventory.

Most manufacturing operations utilize the third optimization approach of manufacturing for WIP inventory. Because the crosscut saws follow the rip saw, the simplest method to control rip-width manufacturing is to limit the number of crosscut saws assigned to specific rip-widths. When a rip-width is over-produced, a materials flow bottleneck shuts down the rip saw. The rip saw operator thus maintains a constant level of WIP inventory in queues feeding the crosscut saws.

Machines now keep track of the WIP inventory and the production goal to some degree. A production goal might be to produce a certain amount of product in a limited amount of time. This information is provided to the crosscut saw operator to influence the cuts made to the next board.

The drawback to optimizing a rip solution for maximum clear wood is that a rip-width may be cut into any number of different length products. Typically clear areas of wood occur in random lengths. A decision must be made about what specific cuts to make to the board to provide the desired production goals. If the decision is by a human operator, the end products produced by the crosscut saw operator may not maximize value recovery, given the production goals.

SYSTEM TOPOLOGY

A sawmill consists of more than just a rip saw and a crosscut saw. A sawmill includes raw materials, people, machines, process control sensors, communication networks, production reports, and product inventory. This section describes a typical rip & cut manufacturing environment and the associated computer integrated manufacturing (CIM) system topology.

Rip & Cut Operation

Figure 33 depicts the plan view of a typical rip & cut sawmill operation. Raw materials, starting from the lower left corner, are conveyed through several manufacturing process and end up as product inventory on the right hand side. This manufacturing

operation depicts a single rip saw producing six different rip-widths, i.e., $13/16$, $1-7/8$, $2-1/4$, and so on, to feed six crosscut saws producing fixed length products for inventory.

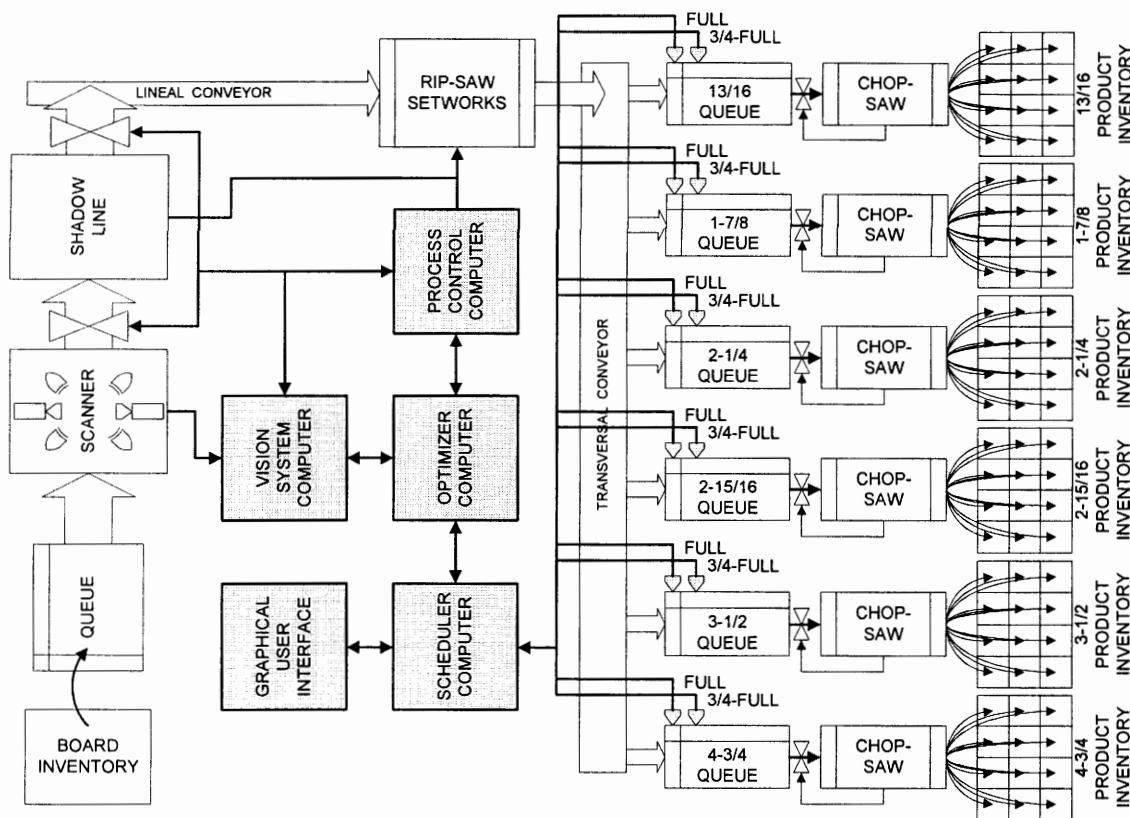


Figure 33. A rip & cut sawmill operation where the rip saw is controlled by a Machine Vision System (shaded and in bold).

The manufacturing begins with wood boards taken from inventory and arranged side-by-side on a transversal conveyor. The conveyor feeds the Image Processor system scanner which images the board and determines the profile measurements and defect locations. A scanned board is passed to the shadow line where it waits for a rip solution from the optimization computer. When a solution is obtained, the board is transferred to a second conveyor where it is transported linearly through the rip saw to produce the rip-widths of the rip solution. The individual rip-widths are then conveyed to one of six

different manned crosscut saws. Each crosscut saw cuts only one specific rip-width. The crosscut saws cut product out of the ripped boards, and the product is placed in inventory.

Before rip-widths enter the crosscut saw, they sit in a first-in first-out queue. Two sensors monitor the status of the queue. If the queue is full, the rip-saw must be stopped and the queue must be allowed to empty a little. The 3/4-full sensor is present to allow early detection of impending production stoppage.

Both the 3/4-full queue sensors and the full queue sensors are used as feedback to the scheduler computer. The scheduler computer also contains a graphical user interface (GUI) whereby production managers can place products on or off the production schedule. If a product is on the schedule, the optimizer computer will use it to determine the value of the rip solution. Again, the rip solution with the highest market value is used to cut the board.

If the scheduler computer detects a 3/4-full sensor tripped, the scheduler decreases the value of the products associated with that particular rip-width. As the value of the rip-width's products decreases, the frequency of the rip-width appearing in the solution decreases. As the 3/4-full sensor is un-blocked, the value of the rip-width's products is allowed to increase back to market value and that rip-width's production increases. At some point, a production equilibrium is reached where all rip-width queues are at or near the 3/4-full level and products are optimally valued for the crosscut saw production rates.

CIM Organization

The rip & cut manufacturing operation is organized as an independent self-controlled process, and the Machine Vision System is integrated as an “island of automation.” The Machine Vision System forms a stand-alone computer integrated manufacturing (CIM) environment. The CIM model for the Machine Vision System, illustrated in Figure 34, is an organization of computers linked together by a high-bandwidth local area network or LAN.

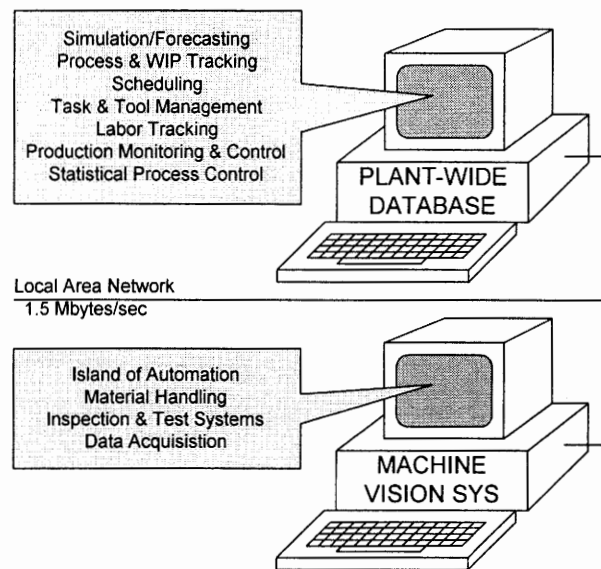


Figure 34. The Machine Vision System organization for computer integrated manufacturing.

In an automated plant, all operations are controlled by a central mainframe computer that communicates instructions to “islands of automation” such as the Machine Vision System. The primary job of the mainframe computer is to determine the master

production schedule. The Machine Vision System uses the graphical user interface (GUI) to input products and display time-critical production information.

In this implementation, processing tasks are logically split into demand-driven material handling and data acquisition activities and higher-level but not-time-critical tasks such as simulation, tracking, and SPC activities. The latter activities are performed by other computers located throughout the plant-wide LAN network. Separating such functions allows for simple and easy to use ergonomic control schemes at the Machine Vision System.

The Master Process Controller of the Machine Vision System (see Figure 7) manages all time-critical processes of the Rip & Cut island of automation. A programmable logic controller (PLC) is dedicated to direct machine control over power devices, user input/output, Image Processor system hardware, and other real-time devices. The Image Processor system and optimization hardware communicate results to the Master Process Controller which in turn sends control instructions to the PLC and updates the central database. The Master Processor Controller also receives optimization parameters from the GUI and communicates them to the Image Processor System and Board Value Optimizer.

The main operator interface to the Master Process Controller is through the PLC control panel. The goal of this approach is to simplify the operating environment and maintain real-time operations at a minimum cost of hardware. The GUI maintains the optimization parameters and collects statistics regarding the production of WIP inventory.

PRODUCT OPTIMIZATION

Optimization, in the present context, is a broad term for selecting cut products that are made from a wood board so that value of cut products is maximized for a given production requirement. Optimization involves three basic steps: 1) determining possible rip-widths that “fill-out” a board, e.g., $2\frac{1}{4}+2\frac{1}{4}+2\frac{1}{4}$ or $3\frac{1}{2}+3\frac{1}{2}$ or $4\frac{3}{4}+13/16+13/16$ and so on; 2) selecting the highest value permutation of the rip-widths, e.g., $4\frac{3}{4}+13/16+13/16$ or $13/16+4\frac{3}{4}+13/16$ or $13/16+13/16+4\frac{3}{4}$; and 3) selecting the single best solution that satisfies value and production goals.

Figure 35 through Figure 38 illustrate how a rip solution is determined. Figure 35 illustrates two different rip solutions that “fill-out” a board width (see also Figure 3). The dashed line locates the position of the rip saw cut along the length of the board. Each rip solutions is generated from a permutation of the two rip-widths “A” and “B,” i.e., A-B and B-A.

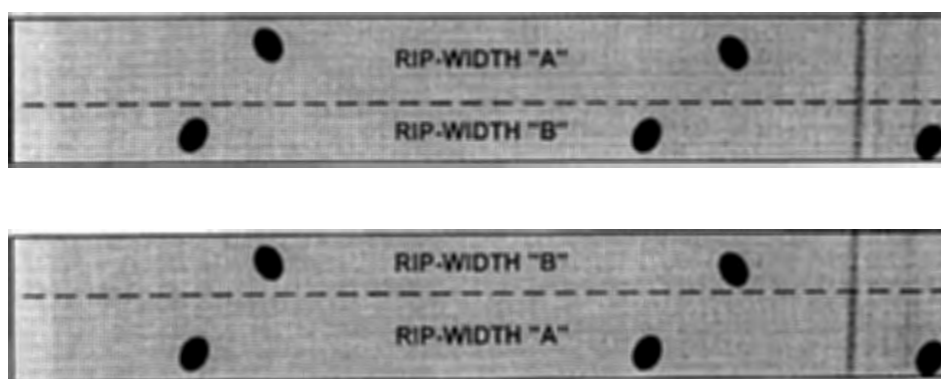


Figure 35. Two ways rip-widths “fill-out” a board width.

Figure 36 and Figure 37 illustrate two families of fixed-length products (called a cut order) comprising the rip-widths. After ripping the board, rip-width “A” may be crosscut into any of 4 different-length products, and rip-width “B” may be crosscut into any of 5 different-length products. The goal of crosscutting the rip-width is to produce the highest value of product. So, the crosscut operator will select the appropriate combination of fixed-length products that best maximizes the value of clear wood.

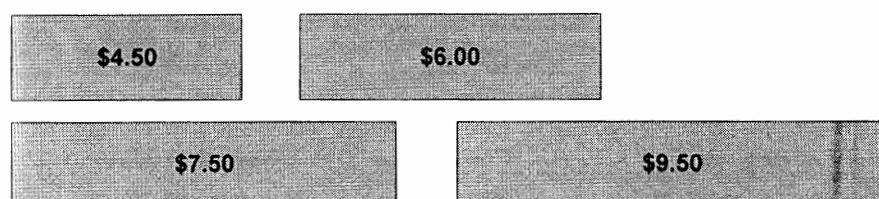


Figure 36. Family of fixed-length products comprising rip-width “A.”

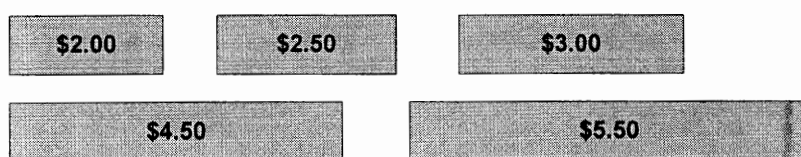


Figure 37. Family of fixed-length products comprising rip-width “B.”

Figure 38 illustrates how the fixed-length products may be placed into the two permutations of rip-widths of Figure 35. The total value of products generated by ripping the board using the first A-B rip-width permutation is \$22.50. The total value of products generated by ripping the board using the second B-A rip-width permutation is \$23.00. To recover maximum value, the rip solution should be the second B-A rip-width permutation. This section discusses the methodology of product optimization and proposes computer-based wood product optimization for sawmills

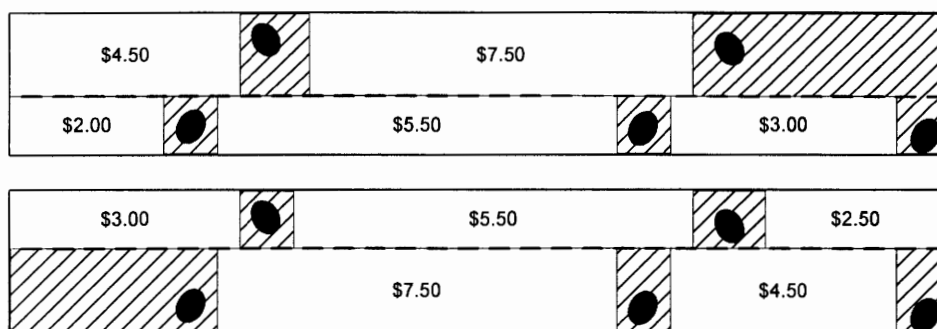


Figure 38. Two ways product (non-hatched areas) may be placed into the board faces of Figure 35.

Computer-Based Optimization

Because it is not possible for a human rip operator to physically measure the clear lengths of wood available in a board and optimize the rip solution for specific product lengths, some form of computer-based optimization must be employed to recover maximum value given specific end products. Computers can perform very complex optimization tasks requiring mathematical precision and accuracy. A computer programmed to perform rip solution optimization may employ several different methods to place product into clear areas of boards.

The simplest but most computationally expensive of these methods is to try all possible combinations of products in an exhaustive search and select the combination that yields the highest value. Because the problem is decomposed first by ripping and then by crosscutting, the crosscutting solution is easily obtained through greedy or dynamic programming knapsack algorithms [24, 25, 26, 27, 28, 29]. The next method of optimization is to use expert-type rules to reduce the number of possible solutions and then employ the first method [30, 31, 32, 33].

A third method of product placement optimization, called coverage, is derived from game theory [34]. Using this method, the entropy (or value) of a board solution decreases (or increases) as *blocks* of solutions (products in rip-widths) are *fit* in the clear areas of a board. Because the shortcut to the coverage process begins as a function of maximum utilization (longest and widest clear), the starting solution eliminates many non-optimal solutions. Unlike the methods trying all possible solutions, coverage algorithms are gradient-based in nature [35]. Because the entropy of a rip solution converges, the process maximizes the time the best-covered solution is sought.

A fourth method of optimization builds upon the coverage process by adapting the shape of clear regions using probabilistic reasoning rather than fitting known blocks into clear regions. This method is based upon neural network and genetic algorithm technology. It involves an ensemble of inter-connected competitive neurodes designed to “grow” within the clear regions to a solution of maximum use and value. Self-Organizing artificial neural systems are shown to outperform conventional expert and search algorithms in terms of the number of solutions tried per second in sequential processing architectures [36, 37].

Because a board is ripped length-wise and then crosscut width-wise, the neurodes compete in teams along the length of the board and secondarily along the width of the board. The final solution represents a two-dimensional optimization of the recovered value and desired products. Because of the generic n-dimensional nature of neural networks, other optimization parameters, such as the desirability of infrequently produced products, can be added to the main optimization algorithms without additional overhead.

For this work, the first optimization method of trying all possible combinations of product is employed. This optimization method is simple to implement and provides a good framework for understanding and rapidly changing (blackboarding) the optimization methodology. Further, many time-consuming calculations may be pre-calculated and stored in look-up tables (LUTs) to reduce the processing time of exhaustive searches.

It is estimated that some of defects found by the Image Processor will have less than certain correlation with the model defect features. To find the optimum solution, it is important that *fuzzy* defect information be used in the optimization process. Further discussion of probabilistic reasoning and the use of coverage-based optimization methods are presented in Chapter VI.

Material Requirements Planning

It is a matter of historical record that among manufacturing companies that have implemented materials requirements planning (MRP) systems since the 1960s, the most significant results were achieved by those who simultaneously undertook a fundamental overhaul of their organizations. When implemented, MRP systems not only demonstrate their operational superiority, but afford an opportunity to gain new insights into the manufacturing process. Implementing MRP in the Rip & Cut process will provide a revolutionary fundamental *understanding* of the production of WIP inventory from lumber. MRP systems have been proven to reduce WIP inventory and improve delivery service at the same time.

The mainstays of conventional inventory management must be open to question given the potential of *controlling* the production of the Rip & Cut operation. Specifically, the concepts of stock replenishment built around reorder points, the square-root approach to the economic order quantity, the analysis and categorization of inventory by function, and the notion of aggregate inventory management must be repealed in favor of MRP. With MRP in the Rip & Cut process, production schedules and lead times will be accurately calculated.

Board Value Optimizer

MRP systems possess an inherent ability to reevaluate the validity of all open shop-order and purchase-order due dates, and thus keep these due dates current. To determine process times requires first, understanding and second, developing a strategy of predicting material availability. The Board Value Optimizer, as part of the Machine Vision System, provides a mechanism to collect, analyze, and predict the availability of WIP inventory for each grade of board. While processing lumber, the Board Value Optimizer will develop a model for planning material availability.

The Board Value Optimizer adds the dimension of time to inventory status data. As a step beyond the perpetual inventory control concept of *what* and *how much to produce* in the traditional rip & cut sawmill, the Machine Vision System will answer the crucial question of *when will it be produced*. *When* will the WIP inventory be available? The inventory planner needs this information to prevent a stockout or shortage and to set production schedules. The Machine Vision System approaches process control with the

following principles: 1) optimizing use and value for production; and 2) calculation of WIP inventory availability over time.

The Board Value Optimizer will forecast (using conventional autoregressive forecasting techniques) the availability of WIP inventory so maximum utilization of clear wood may be taken on a board-by-board basis. This process is exemplified in expediting long-length products. MRP calculates WIP inventory availability and time-phases the value and utilization parameters for the optimization process. In this fashion, the evolution of an MRP-based optimization system is a logical extension of the present weight-based perpetual inventory control system. The goal of adding the MRP time-phasing dimension is to control the availability of each piece of WIP inventory so maximum value may be obtained from each board as the system follows a master production schedule. However, the MRP concept presupposes that lead times to produce WIP inventory can be determined by evaluating past results.

Because the Board Value Optimizer determines the physical availability of WIP inventory, the Rip & Cut process becomes an order-expediting or “pull” process. Under the present manual order-launching or “push” system, value is achieved at the cost of product availability. Hence, large quantities of safety or buffer WIP inventory must be maintained to fulfill production scarcities of long-length products. The proposed Machine Vision System will reduce the dependence on long-term production averaging (perpetual inventory control) through the discrete utilization of lumber for availability with respect to maximum utilization.

CHAPTER V

SELECTING PRODUCT

Every tree grows differently, so the pattern of defects in a wood board is random and hence the production of rip-widths is random given the goal of maximizing the recovered value. Figure 39 illustrates a time-series view of the Machine Vision System continuously ripping 212 Common-grade boards. The x-axis tracks the board number and the y-axis indicates the production of one or more 1-7/8 inch rip-widths.

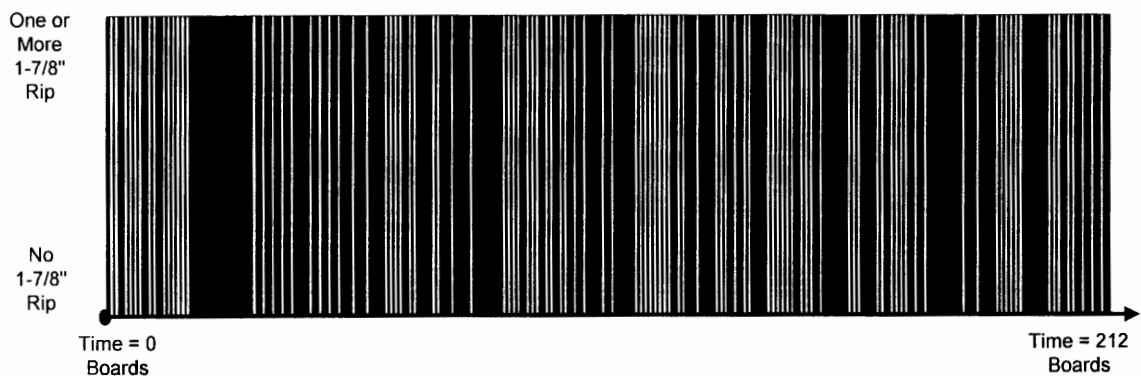


Figure 39. Occurrence of the 1-7/8 inch rip-width in ponderosa pine of Common grade for a continuous production period of 212 boards.

Wide black bands along the x-axis of Figure 39 indicate the Machine Vision System is cutting the 1-7/8 inch rip-width “heavy.” Note that the dollar values of all rip-

width's products are essentially static over the entire production period.* Figure 39 illustrates that the natural occurrence of the 1-7/8 inch rip-width is random. The production of other rip-widths is similar.

When the Machine Vision System cuts “heavy,” the downstream crosscut saw processes are not able to keep up with the volume of rip-widths produced, and rip-saw production is forced to stop. Overproduction of narrow rip-widths are common because narrow rip-widths have fewer defects per lineal foot than wide rip-widths; hence, narrow rip-widths are common in high-value solutions. For the Machine Vision System to function effectively in a manufacturing environment, the production of each rip-width must be controlled over time.

As a general rule of product optimization, decreasing the “value” of a rip-width effectively decreases the probability that the rip-width will appear in the highest valued rip solution of any given board. So, the overproduction of narrow rip-widths may be controlled by mathematically decreasing the value of the rip-width. Conversely, the under-production of wide rip-widths may be controlled by mathematically increasing the value of the rip-width. In practice, however, it is only necessary to decrease the value of narrow rip-widths, since the remaining highest value solutions will contain wide rip-widths. In value-based optimization, mathematically changing the value of a rip-width acts as a proxy for controlling the production of rip-widths.

* In this production test, weights are used to change the values of products for each rip-width; however, the weights are based on long-term production goals and change slowly over a 100 board period.

Using value as a proxy for controlling rip-width production has the effect of changing the goal of maximum value recovery to maximum production-value recovery. However, it would be an error to optimize the crosscut products based on production-valued product. For example, if long-length products of a narrow rip-width are undervalued for production reasons, wider rip-widths containing shorter-length products are taken instead. Because the maximum value optimization method does not prevent the “value loss” of the more desired long-length products, the value of a rip-width must be calculated based on market-valued products.

A board may be ripped using only one rip solution; however, the same board has many possible rip solutions. By choosing an appropriate rip solution, the production of rip-widths may be controlled. Further, changing the value of rip-width products in a rip solution (after product optimization) produces the desired control effect and does not alter the value recovery optimization objective. Therefore, rip-width production may be controlled through a secondary rip solution selection process wherein product optimization is performed at market value, and the rip solution is selected based on production value.

This chapter describes a weighted-value rip solution selection method developed by the author for use in the Machine Vision System (cf. Figure 8, page 20). The methodology and development of the weighted-value selection process is presented first. Testing of the selection algorithms follows, and implementation observations of the Machine Vision System concludes this chapter.

WEIGHTED-VALUE SELECTION

If wood were mostly clear (sparse placement of defects), a very simple selection process could control the production of rips. However, not every rip is economically produced from any one board given the goal of maximum value recovery. The production of rip widths from the rip saw must balance the consumption of rip widths at the crosscut saws for the system to work. This is a pull-type manufacturing operation, and is illustrated in the flow diagram of Figure 40.

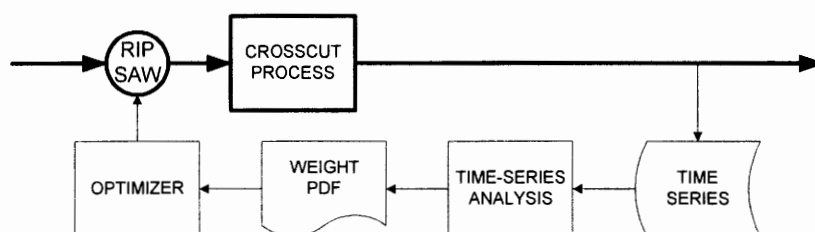


Figure 40. The manufacturing operation illustrated as a closed-loop system.

To control the rip-width production rate for the crosscut saws, the manufacturing operation must operate as a closed-loop process. That is, information about the crosscut saw production rate must be put into the Machine Vision System to “close the loop” of manufacturing. As Figure 41 illustrates, production through the crosscut saw is monitored and processed as a time series to yield production rates. The production rate is then analyzed and used to update weights that are multiplied with product market values in the optimizer. As the weights decrease from 1.0 to 0.0, the value of products decreases to zero, and the probability that the corresponding rip-width will be produced decreases. Other variables could be added to improve system functionality, illustrated in Figure 41.

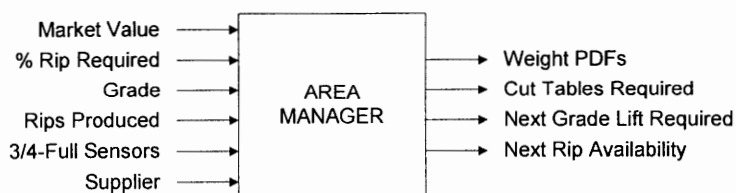


Figure 41. Possible process input and output feedback used to control the rip saw operation in a dynamic manufacturing environment.

Time-Series Analysis

For this work, the only feedback to the Machine Vision System comes from the 3/4-full and full on/off sensors monitoring each crosscut saw. The production rate of the crosscut saw must be determined from a time-series calculation of the sensor data. By monitoring the time the sensor is on and off during a fixed period of time, both short and long-term production rates are calculated.

Both the short and long-term production rates are forecasts of the actual rate. If the forecasted rate is greater than the actual rate, the 3/4-full sensor will turn on. If the forecasted rate is less than the actual rate, the 3/4-full sensor will turn off. The forecasted rate is increased if the sensor is off and decreased if the sensor is on.

Because boards jam and stack on top of each other or jam in the equipment, the 3/4-full sensor sometimes relays false information about the actual production rate. Such random effects are removed through time-based averages. Two such averages are maintained for a 5 minute (short-term) and 10 minute (long-term) period. Such decomposition beliefs are conceptually written as: $\text{actual} = \text{pattern} + \text{randomness}$.

Production is understood from its four basic components: 1) trend; 2) seasonality; 3) cyclical and combined cycles; and 4) randomness. The goal of using time-based

averages is to remove only randomness in the short-term production rate and to remove seasonality, cyclicalities, and randomness in the long-term production rate.

Using two production rates allows the Machine Vision System to react in two modes. The short-term feedback allows the system to respond quickly to situations that may cause production to stop. The long-term feedback allows the system to stabilize and balance production rates. The short-term feedback, however, must affect the weight changes to a lesser degree than the long-term feedback or the system will be unstable.

Product Weights

If the weights used to control production apply equally to all products in a specific rip-width, the maximum utilization of long-lengths of clear wood is not achieved. A simulated optimization run revealed the necessity of having multiple weights over several length classes of end product. Using multiple weights improved the recovered value by $17.8\% \pm 5.8\%$ per board foot, i.e., \$21,537.59 with multiple weights versus \$17,988.10 with a single weight. Table V shows the difference in recovered value between Test 1 where a single weight modifies all products in a rip-width and Test 2 where three weights are used to modify different length classes of products in a rip-width.

The first length class modifies all the products less than 13 inches long, the second length class modifies 13 inch to 72 inch long products, and the third length class modifies products over 72 inches in length. Having multiple weights for different length classes enables the optimizer to selectively choose the rips that have higher intrinsic market value, i.e., longer and wider clear wood. Allowing short-length product weights

to fall to near zero value while the value of long-length product weights remains at market value best maximizes market value recovery.

TABLE V

EFFECTS OF USING ONE WEIGHT VERSUS THREE WEIGHT
CLASSES TO MODIFY THE PRODUCTION OF RIP-WIDTHS

Measurable	Test 1 One Weight Per Rip-Width	Test 2 Three Weights Per Rip-Width	Test Variance
BdFtg IN	21,308.7	20,633.5	+/- 2.01%
BdFtg OUT	21,004.6	20,306.0	+/- 1.78%
Recovered Value	\$17,988.10	\$21,537.59	+/- 0.53%
Value/BdFt	\$0.8442	\$1.0438	+/- 1.49%

Table VI and Table VII detail the results of Test 2 and Test 1, respectively. Each test simulated ripping 1000 boards of All-Grade quality using actual board model data taken from the Machine Vision System. For the tests, the 1-7/8" rip-width production rate was controlled to produce 3500 BdFtg. All other production rates were uncontrolled.

TABLE VI

RESULTS OF TEST 2 USING THREE WEIGHTS TO MODIFY THE
PRODUCTION RATE OF THE 1-7/8" RIP-WIDTH

	13/16" Rip	1-7/8" Rip	2-1/4" Rip	2-15/16" Rip	3-1/2" Rip	4-3/4" Rip	Total \$Value
BdFtg OUT	338.2	3584.1	2292.8	4645.9	2260.6	7184.4	
< 13"	\$0.00	\$23.73	\$16.63	\$35.31	\$35.72	\$142.55	\$253.94
13" - 72"	\$85.09	\$2156.01	\$1130.59	\$2313.35	\$1022.45	\$4131.75	\$10839.24
> 72"	\$557.15	\$2061.76	\$1687.27	\$3020.27	\$687.27	\$2430.70	\$10399.42
Total \$/Rip	\$642.24	\$4241.50	\$2834.49	\$5368.93	\$1745.44	\$6705.00	

TABLE VII

RESULTS OF TEST 1 USING A SINGLE WEIGHT TO MODIFY THE
PRODUCTION RATE OF THE 1-7/8" RIP-WIDTH

	13/16" Rip	1-7/8" Rip	2-1/4" Rip	2-15/16" Rip	3-1/2" Rip	4-3/4" Rip	Total \$Value
BdFtg OUT	28.0	3449.1	4620.5	229.8	6477.1	5452.8	
< 13"	\$0.00	\$30.41	\$21.49	\$31.39	\$108.13	\$92.52	\$283.94
13" - 72"	\$0.00	\$2339.46	\$1444.62	\$1706.97	\$2608.99	\$1620.04	\$9720.08
> 72"	\$14.27	\$1662.16	\$1065.85	\$1593.97	\$2572.88	\$1074.68	\$7983.81
Total \$/Rip	\$14.27	\$4032.03	\$2531.96	\$3332.33	\$5290.00	\$2787.51	

The far right columns of each table sum the recovered value for each of the three product length classes. Test 2 using three weight classes is shown to recover more value in products > 72" in length versus Test 1 using only one weight. Even though long-length products are worth more than shorter-length products, the results of this test indicate that weight-based maximum value optimization does not guarantee production of desired long-length products. Thus, using three or more length-based weight classes to (mathematically) reduce the value of shorter-length products before reducing the value of longer-length products is warranted.

CUMULATIVE PROBABILITY DISTRIBUTION FUNCTIONS

Long and wide clear wood should be taken whenever possible – clear rip-widths can always be cut into smaller lengths and rip-widths – because it occurs less often. From the previous section, using separate weight classes for different product lengths increases the recovered value when manufacturing constrains rip-width production. Further, if a

long and wide clear is present in the wood, it would be unwise to choose a rip solution that does not cut a long and wide clear product in favor of a mix of rip-widths that better match production. Recall that the goal of implementing the Machine Vision System is to improve recovered value. This section describes a method of choosing a desirable market-value rip solution over a weighted-value rip solution.

The frequency with which any product occurs may be measured, and a metric for determining the probability that any product will occur in the next board can be predicted. The frequency of occurrence of a product is an excellent metric for selecting a market-value rip solution over a weighted-value solution. A cumulative probability distribution function (CPDF) is one such metric measuring the occurrence of product.

The CPDF is generated from a histogram of the occurrence of a product over time. Figure 42 and Figure 43 show the hypothetical result of conducting a time-series experiment monitoring the occurrence of the 3-1/2" rip-width in a rip solution during a sequential production scenario. Figure 42 shows the histogram of spacing of the 3-1/2" rip-width where the x-axis counts the number of boards between consecutive 3-1/2" rip-widths. From this chart, the 3-1/2" rip-width most often occurs by the time the fourth board is ripped, and at least one 3-1/2" rip-width will be produced from any five boards.

Figure 43 is a CPDF for the histogram of Figure 42. The CPDF graphically illustrates the cumulative nature of waiting for the next 3-1/2" rip-width to occur during sequential production processes. By the fourth board, the probability of ripping a 3-1/2" rip-width is nearly 100%.

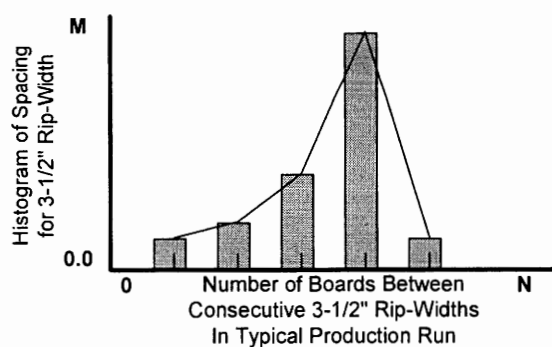


Figure 42. Hypothetical occurrence of the 3-1/2" rip-width in the sequential production of rip solutions.

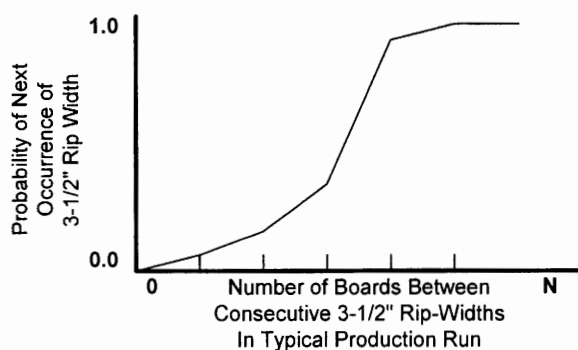


Figure 43. Cumulative probability distribution function for the occurrence of the 3-1/2" rip-width shown in Figure 42.

Figure 44 and Figure 45 show the hypothetical result of conducting a time-series experiment monitoring the occurrence of the 13/16" rip-width in a rip solution during a sequential production scenario. Figure 44 shows the histogram of spacing of the 13/16" rip-width where the x-axis counts the number of boards between consecutive 13/16" rip-widths. From this chart, the 13/16" rip-width most often occurs in every board. That is, at least one 13/16" rip-width will be produced from any single boards.

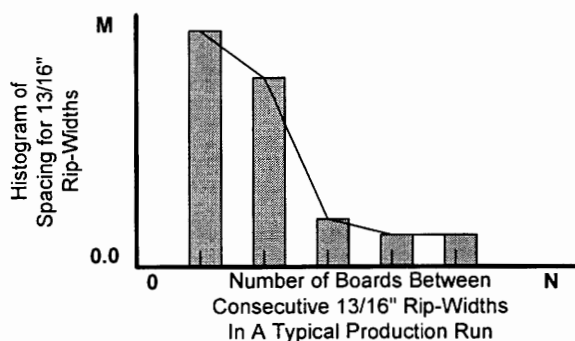


Figure 44. Hypothetical occurrence of the 13/16" rip-width in the sequential production of rip solutions.

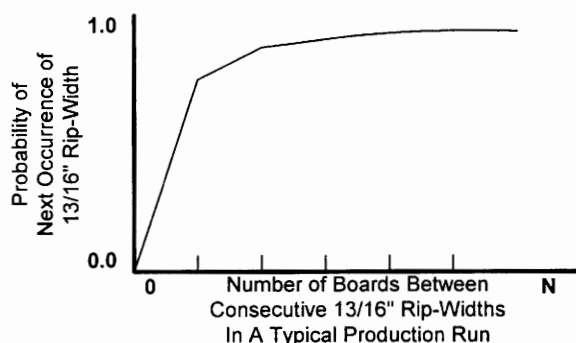


Figure 45. Cumulative probability distribution function for the occurrence of the 13/16" rip-width shown in Figure 44.

Cumulative probability distribution functions can be used to make a comparison between the actual market value and the weighted value in the rip selection process. If the weighted-value solution produces the 13/16" rip-width and no 3-1/2" rip-widths, and the actual market value solution produces the 3-1/2" rip-width and no 13/16" rip-widths, then the actual market value solution should be chosen because the probability of finding a 3-1/2" rip-width in the next board is low.

Because the CPDF is similar to a weight, the CPDF is used as a market-value multiplier to find the weighted-value for the rip solution selection process. In practice, a CPDF is developed for each product, and the frequency with which the product is certain^{*} to occur is used to determine the weight class for production-weights. So a product that occurs frequently is paired with a production-weight that heavily decreases the product's value when the rip-width production rate is greater than the crosscut saw production rate. Conversely, a product that occurs infrequently is paired with a production-weight that has little effect on the product's value when the rip-width production rate is greater than the crosscut saw production rate. In practice, a non-linear equation is developed which maps the CPDF to an infinite number of weight classes.

In summary, CPDFs are powerful statistical descriptors of systems and have found applications in every discipline from control systems to stock market economics to animal sociology. In many contexts, past performance is a good indicator of future events. By studying and analyzing the production of rip-widths and wood products, a mathematical expression is developed to select rip solutions for production and preserve the goal of maximizing value recovery.

RIP SELECTION METHOD

A rip solution is selected by choosing the highest weighted-value rip solution as determined by balancing production and understanding that short clear product is more

^{*} In this work, certainty is the CPDF of $+3\sigma$ or 99.86% for a Normal function.

likely to be found than long clear product. The Product Value Scheduler is a critical component of the Machine Vision System. The Product Value Scheduler calculates the rip-width production rates from the rip saw, calculates the rip-width production rates through the crosscut saws by monitoring the 3/4-full sensors on each crosscut saw queue, and monitors the production of wood products to determine the frequency of occurrence to generate product weights. The product weights are used to select the rip solution used to rip a scanned board. In overview, this is a three-step process: 1) determine how the wood board wants to be ripped based on the frequency of long & wide clear wood products; 2) if a rip-width contains long clear wood, the weight for long-length products is large, e.g., 1.0, to preserve the market value; and 3) if a rip-width does not contain long clear wood, the weights for short-length products are small, e.g., to decrease the frequency of rip-width production.

In general, there is an inverse relationship between a product's weight and the percentage the rip "fills out" a board. This product weighting methodology seems almost too simple to be effective, but from aforementioned simulations, a consistent improvement over the average-weighted method is observed.

The rip selection process is implemented as shown in the flowchart of Figure 46. Implementation occurs in two phases called Optimization and Scheduler. Optimization algorithms determine all possible rip solutions that can be generated from the board model. Scheduler algorithms select one rip solution to rip the board and monitor the resulting action of 3/4-full and full process sensors to adjust the product weights. A listing of the algorithms used are found in Appendix C.

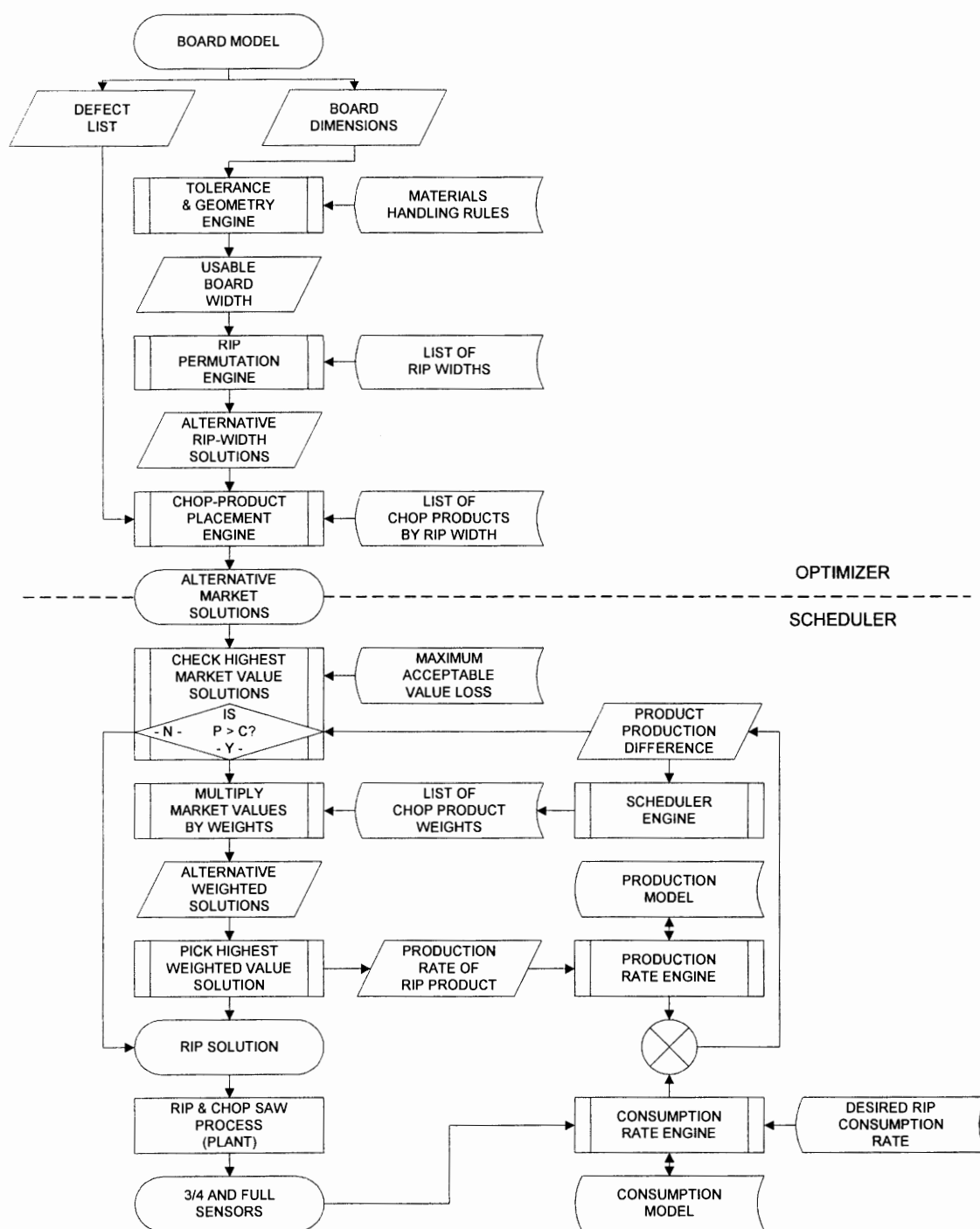


Figure 46. A flowchart of algorithms used to process a board model to determine alternative rip solutions, select a single rip solution to satisfy value and production goals, and monitor plant sensors to implement closed-loop process control.

SYSTEM VALIDATION

On November 11, 1995, an extensive production study was conducted on the Machine Vision System. The purpose of the test was to determine if the weighted value selection method works effectively in a manufacturing plant.

The first measure of overall performance is verification of the Image Processing subsystem to measure the board and find defects. Board measurements are assumed correct as the board “cleans up” through the rip saw and clear areas are present in the rips. Quantifying defect identification is performed by watching the camera display monitor and counting knots greater than 3/8” that are correctly bound, counting missed knots, and counting false grain hits. This test is summarized in Table VIII, and validates the Image Processor performance. Defect identification is broken into two categories of “correctly bound defects” and “missed defects” and the two together sum to the total defects missed by the Image Processor, and “false defects” comprise grain hits and defect boundary extension in the board-width direction.

TABLE VIII

DEFECT BOUNDING RESULTS FROM OBSERVING 100
PROCESSED IMAGES OF PONDEROSA PINE WOOD BOARDS

100 Board Test	Correct	Missed	False
Top Op-Rip Cameras	99.44%	1.32%	2.29%
Bottom Op-Rip Cameras	99.51%	2.08%	3.11%

For the given wood processed during this test, the Image Processing subsystem is finding and correctly bounding 98.12% ($100\% - (0.56\% + 1.32\%)$) of the top-side defects

and 97.43% ($100\% - (0.49\% + 2.08\%)$) of the bottom-side defects. (Does not include knots missed under chain races.) The wood run during this test was 12 foot long by 6 inches to 13 inches wide commons, generally straight with little warp, twist, bow, or crook effects, having shallow wane edge features, typical light yellow appearance with solid well-defined red-green knots, some feathered knots, and some knots smaller than 3/8 inch, and medium grain features.

The second measure of overall performance is verification of the optimization system. Using the Board View screen, the image of the board, defects, and the placement of cut stock boxes with respect to the board and defects was validated. Further, the weighted value method is correctly balancing the flow of rips to the cut tables; no cut table was completely without wood nor was production stopped because of over production of rip-widths.

The third measure of overall performance is operator feedback. Ripsaw operators are expert sawyers and excellent judges of wood product quality. If the Machine Vision System makes a “bad” decision when ripping a board, the ripsaw operator will manually re-rip the board. During the 17-22 March 1996 base test of the Machine Vision System, 8 production shifts reported only 0.37% re-rips (224 out of an estimated 61,200 boards) due to “bad” rip solutions. 43 Boards were also re-ripped due to excessive taper in outside rips; however, the ripper’s solutions did not yield higher recovered value.

In summary, the production-value selection method demonstrated robust process control while preserving valid decisions from the optimizer during operational tests of the Machine Vision System. Note that the results from other value-based tests are

confidential to the manufacturing plant and are not included in this thesis. Generally speaking, the Machine Vision System is capable of outperforming human rip-saw operators in recovering maximum value from a board given dynamic production constraints because the rip-solution selection process is dynamic. Human rip-saw operators are observed to change rip-width production strategies in an on/off fashion. That is, when a rip-width is over-produced, the human rip-saw operator will simply not cut the rip-width for a period of time and then continue as before. The process of selecting a rip-solution which minimizes the production of over-produced rip-widths on an on-going basis is too complex a task for human rip-saw operators to perform at production rates of 15 to 17 boards per minute.

CHAPTER VI

CONCLUSION & SUGGESTED FUTURE WORK

The result of this work leads to the statement: *because a color image of wood contains far more clear and clear-grain colored pixels than grain-knot and knot colored pixels, it is beneficial to first statistically identify and remove the clear and clear-grain color and to use the accumulated data to simultaneously enhance and normalize the remaining grain-knot and knot color pixels. This process is here called adaptive color correlation. Enhanced and normalized knot colored defect features are then robustly recognized using shape-based pattern recognition.*

Adaptive color correlation is shown by this work to possess several characteristics which recommend it in various color image processing applications. Principal applications include identifying defects in a workpiece where the color of the workpiece varies but the color space relationships between non-defect colors and defect colors is present. Adaptive color correlation replaces average-color models and is implemented as a pre-processing step to shape-based pattern recognition.

Separating the weight-based product valuation and selection processes from the product optimization process is shown by this work to preserve market-value solutions which recommends it in various optimization-based inventory driven manufacturing applications. Principal applications include dynamic processes where any number of end

products may be manufactured from the workpiece. The workpiece contains random defects which prohibit the production of some products. Weight-based product valuation and selection for inventory scheduling is implemented as a post-processing step to workpiece optimization.

This chapter summarizes the work of the adaptive color correlation algorithm and the product value scheduling algorithms presented in this thesis. The first section contains conclusions about the Image Processor work, and the second section contains conclusions about the optimization work. This chapter also presents suggested areas and topics for future work. The results of implementing this work in the *OP-RIP II+ Machine Vision System* are located in Chapter III and Chapter V.

AUTHOR'S CONTRIBUTIONS

The author devoted about three years to the Image Processing and Product Optimization problem, with nearly one-third year being field work. The larger part of the project was spent on laboratory work developing, constructing, and generalizing the various software algorithms of the Machine Vision System. Generalization is an important iterative step in software development, and its activities center around finding optimal algorithm embodiments to improve the real-time processing throughput. Generalization tasks took roughly one-half of the total laboratory time. The majority of

the field work was spent on developing various length-based valuation strategies to make the Machine Vision System rip wood the way a human rips wood.

The goal of the image processing portion of the project as presented to the author was simply stated as: "Take the known algorithms and make the system throughput 15 boards per minute." Early in the development process, the author determined that the shape-based model correlation processes would not yield the desired throughput for the large number of candidate defect features required for the "fixed" average-color model approach. Accordingly, the author then developed a new image processing methodology based on an "adaptive" color model. The given image processing strategy was to search and test for defect features directly. The strategy proposed and developed by the author was, instead, to classify all wood pixels containing non-defect colors first, and then identify defect features. The result of this work improved image processing throughput to an average 3.5 seconds per scan from an average 5.0 seconds of the conventional strategy for a given set of 3x3 to 64x64 shape models dynamically applied to each defect feature candidate. The author's additional work to improve the shape-based feature detection and the development of reflective gradient filters to provide uniform lighting are not reported in this thesis.

One of the author's contributions to image processing deemed to be particularly significant in the present context, was the observation that wood contains far more clear and clear-grain colored pixels than grain-knot and knot colored pixels and that such

information can be statistically quantified. The author believes that apart from its application to wood, the model color adaptation methodology and algorithms which have been developed are superior to absolute or fixed model methods used in other image processing paradigms, and that its versatility will recommend it in numerous applications. Because the human image processing system does not use absolute color mechanisms, and because machine image processing systems are required to solve problems that humans can solve, the author's work recommends that many naturally occurring image processing problems would be amenable to similar solutions.

In addition to the above-described work on adapting the color model, the Optimizer was improved by this author's work by virtue of separating the optimization into two steps: 1) determine all possible product solutions for a board; and 2) select the single best solution that satisfies value and production goals. This strategy, along with software memorization and blackboarding techniques, decreased the optimization processing throughput to 0.5 seconds per board model from 8.0 seconds of the previous strategy. The author's additional work to field-develop value-based logic to emulate operator rip permutation strategies, the development of an ergonomic (simplified) operator interface, and the establishment of product and requirements specifications are not included in this thesis.

Another potentially significant contribution of the author to wood products optimization was the observation that "wrong" solutions are typically produced when

using: 1) weighted-value methods during the product placement strategy, and 2) single-weight classifications for all wood products of a single rip-width. The author believes that apart from its application to wood, the optimization and selection methodology and algorithms which have been developed are superior to conventional weighted-value optimization approaches. The author's work recommends that many optimization problems involving random placement of defects in a workpiece would be amenable to similar solutions.

IMAGE PROCESSOR SUBSYSTEM SUMMARY

In summary, improvements to the Image Processor were successful. The adaptive statistics-based color correlation filter enhances defect features in all types of pine wood without pre-defining a defect feature color model. Implementing the color correlation algorithms as a dynamic look-up table (LUT) and porting it to the DSP RAM for fast 0 wait state execution improved the average real-time throughput to 3.5 seconds per scan from 4.5 seconds (a significant improvement). Color correlation accuracy and dynamic adaptation improved using a “clipped” (saturating the CCD cameras) intermediate 15-bit resultant image. Clear, grain, and knot color feature separation improved using curve-fitting techniques given a histogram of the intermediate 15-bit resultant image. A better non-linear sigmoid transform specifically designed to “binarize” clear from knot given

the 5-bit accuracy of the Acumen 900 Image Processor produced crisp, or non-probabilistic, defect feature recognition.

Shape-based pattern matching and subsequent defect bounding and conversion to real-world coordinates was corrected and verified by measuring known defects. False recognition of grain patterns along the core of some boards led to the development of a false-grain-elimination algorithm. This algorithm removes elliptical single-sided defects from the core area of a board, and was adapted to provide detection of board core areas to the optimizer. Adjustments to the edge cleanup algorithm for warped boards were implemented, and although the adjustments were conservative in nature, up to 1/8" more material from the 70% cleanup line may be recovered.

By designing and fabricating a reflective gradient filter, uniform lighting was obtained across the length and width of the board face. Note that the end halogen light bulbs have higher wattage to better illuminate the board ends. White paint is also used to reflect incident light shown on the scanner frame to the board ends.

PRODUCT VALUE SCHEDULER SUMMARY

In summary, improvements to the Optimizer System were successful. The Product Value Scheduler was developed to separate rip and crosscut product optimization and rip solution selection. The task of balancing the rip saw production rates to the crosscut saw production rates was accomplished. Product value weights are updated by monitoring

inventory requirements, the production rate of the rip saw, and the production rate of the crosscut saw.

Production Process Seen as Pull-Type Operation

The Machine Vision System works most efficiently as a “pull” operation. That is, the crosscut saw operators generate a demand for rips from the rip saw. Value (or profit) is lost anytime the maximum-value rip solution is not selected because of production requirements. In simulations, the weighted-value rip solution selection method preserves the recovered value of long product lengths.

In practice, too much emphasis is placed on keeping each crosscut saw operator busy. In practice, wood is “pushed” onto the manned tables without regard to value loss and the long-term production mix. Because different grades of lumber produce different mixes of product, value is lost as the system balances production rates from lift-to-lift. More value is recovered when crosscut saw operators (production capacity) move about to man different crosscut saws, thus better matching the grade of wood in the system.

Because the value of long cut stock is greater than short cut stock, the product selection process was made sensitive to long clear areas in boards. The weighted value of long cut stock typically does not change from the market value. This effect contributes to rip solutions containing long cut stock. Unfortunately, long cut stock of narrow-width rips must be chopped into short cut stock, called finger joint stock, if the board is warped or twisted. Also, long clear areas on the outside of boards drive the highest market-value

selection process to choose rip solutions placing long and wide rip-width cut stock in outside rips. This was commonly observed as a 308-114-308 rip pattern where the 114 rip contains some unusable core material which must be hogged. Bound stress cannot be identified by the Image Processor.

Product Recovery in Secondary Manufacturing Operations

Finally, it was observed that the crosscut saw operators do not recover long-length products well. Most important, the crosscut saw operators need several inches of clear material buffering the long-length products before they attempt to take it. Cutting short-length products is the operator's preferred method of crosscutting if the long cut stock is not an obvious decision. If a product is so long as to extend outside of the visual perception range of an operator, that operator is less likely to take the product. Why should the operator take the time to look for a long-length product when he is rewarded for throughput?

The Optimizer, however, follows the rule of maximizing value and often places high-value cut stock closely between solid defects. Thus, additional "buffer" material must be added to long-length products to ensure that the crosscut saw operators make the same cut decisions as the Optimizer. This material is usable waste; so, the smallest length of buffer material should be no less than the short finger joint products.

FUTURE WORK

Much has been learned from the Machine Vision System. We should take this opportunity to study what can be improved, plan to make improvements to the existing system or the next generation of systems, and implement the improvements wherever cost effective. The following subsections summarize areas of possible improvement.

Probabilities For Optimization

Much of the information used to detect defect features by the Image Processor algorithms is not used in the optimization algorithms. For example, each defect feature has associated with it a probability from 0 to 100% that it best fits the color and shape of a knot defect feature. A board has certain regular properties, such as core and knot helix patterns, that could be used to validate the computed probabilities of defects.

The Image Processor, however, uses a fixed threshold to determine if a defect feature exists or not. If the threshold is too low, too many non-defects will be falsely found. If the threshold is too high, too few defects will be found. Rather than have the Image Processor make a go/no-go “decision” about a defect candidate, the optimization process can be improved by including all likely defect candidates in the Board Model.

Applying probabilistic reasoning to the optimization process will yield a rip solution of the highest probable value. Levels of confidence or certainty of a rip-width solution change dynamically on a board-by-board basis. Using the maximum amount of

information possible in the optimization stage guarantees that the long-term use and value of clear wood is maximized. This topic is further explained in the following sub-section entitled “Product Probability Scheduling.”

Issues of side defects, other surface defects, and warped and twisted boards are best addressed through secondary feedback mechanisms. For example, a closed-loop rip and crosscut production operation using automated chop saws would provide yield feedback to the Product Value Scheduler to update the weights to find the most value in the wood by incoming grade. Additionally, this closed-loop system best couples a materials/inventory requirement schedule with the production capacity of the wood given the goal of value recovery.

Finally, key information about how an individual rip is to be optimized into finished goods for maximum value is thrown away once a board is ripped. The most logical next step to improving the system is to paint marks on a board where it is to be crosscut into products as it enters the rip saw. Such marks would have a dual purpose. The paint marks would aide the crosscut saw operator in making decisions about how to cut for maximum value. The crosscut saw operator’s job becomes one of verifying the system, and because the operator spends less time making complex decisions, the operator’s throughput increases. In an automated plant, the painted marks could be read by one of several existing automated crosscut saws.

Adaptive Clipper Circuit

In the current algorithms, the sigmoid transform is modified over every 0.4"x0.4" area due to computer processing time constraints. This coarse transform leads directly to false grain hits and missed light- or clear-color knots. By performing a pre-scan of the board during the time it is allowed to settle, a pre-calculation of the sigmoid transformations can be calculated on a pixel by pixel basis. The pre-scan board image will be blurry, but blur does not affect the color of the board. The total time to complete a scan is expected to decrease by 1.23 seconds.

It is possible to further reduce the image processing time by moving the dynamic LUTs to hardware. Figure 47 illustrates one such circuit designed to work between the camera and the Image Processor hardware. The circuit contains a matrix multiplier controlled by an independent microprocessor performing the adaptive LUT in gate array circuitry. Such a system will operate as follows:

- 1) the incoming pixels are converted from an analog to digital signal;
- 2) the color correlation process pre-calculates the histogram of each color channel during a pre-scan image;
- 3) the on-board CPU will perform a gaussian curve fit for clear, grain, and knot – this is the framework for learning and compensating for camera drift and dirt accumulation – and a second normalizing gaussian curve fit for clear, grain, and knot is also calculated;
- 4) the CPU will memory map the image to dynamically choose from among 256 different sigmoid transforms on a pixel by pixel basis to convert each color channel image;
- 5) after the input LUT, the red, green, and blue color channels are multiplied together and summed in the matrix multiplier for 24 bit to 8 bit data compression of the red, green, and blue information;
- 6) the summed pixel is normalized by the output LUT;
- 7) the output pixel is converted back to an analog signal.

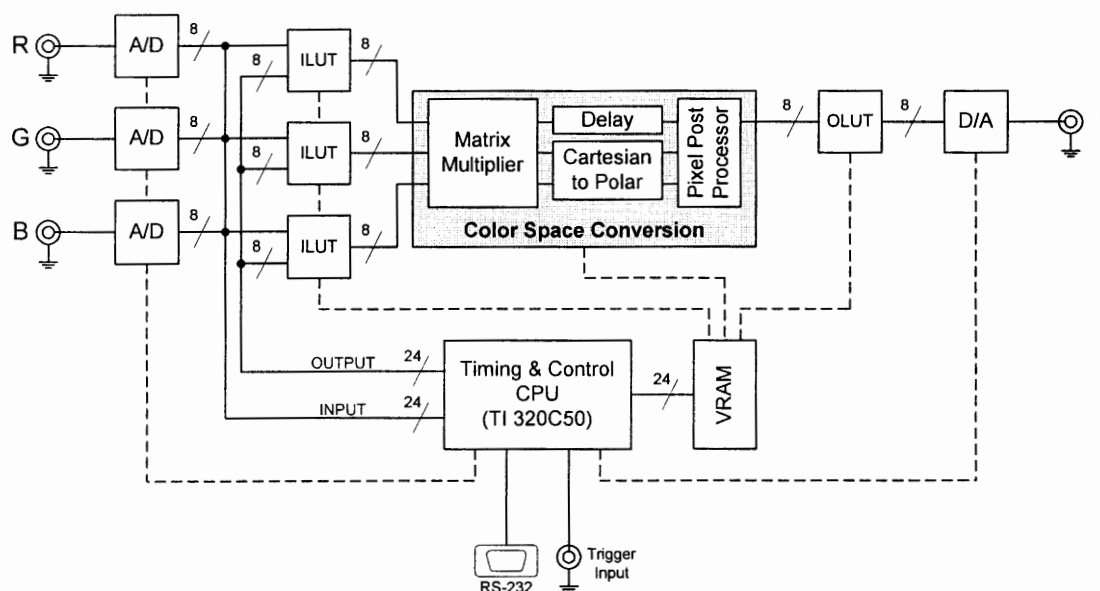


Figure 47. Potential hardware implementation of the adaptive color correlation algorithm.

Raster Shape Correlation

Because the future of the Rockwell correlation chip is uncertain, the shape-based correlation algorithms must be ported to conventional DSP platforms. Porting requires developing new algorithms for DSP processors. Model matching is essential to locating knots in planer skip and heavy grain (noise). Thus, the present area-based knot model must be transformed to a raster-type model that is easily implemented in a DSP memory architecture. In general, the function of an area model is preserved in a line model when additional post-processing algorithms are added.

The next generation of Machine Vision System will remain basically the same. The correlater chip is easily replaced with a DSP performing the same computations. While the correlater chip performs four math functions at once, it operates at only a 25MHz clock cycle. A 100MHz processor performing four sequential math functions will have about the same performance. Additionally, DSP-based image processing hardware such as Coreco's F/64-DSP16 supports an 80Mbyte/sec data transfer rate between boards. Such communication pathways allow the top and bottom cameras to communicate knot location and probability information. By analyzing the shape of both the top and bottom defect features, false grain labeling should be significantly reduced.

Grain Angle Determination

The Image Processor must be improved to correctly bound defects based on grain angle. Such improvements are beyond the capacity of the correlater-based Acumen 900 computer given the 4 second per board processing constraint. Using a 3-Chip color camera with 8-bit color resolution per channel will increase the color-space resolution and allow grain angle and small defect pattern analysis.

A backpropagation neural network was developed to classify grain as either horizontal, mixed, or vertical. By studying the neural network, a simple 2-D adaptive linear combiner filter was developed to perform the same operation. This 2-D filter can be applied over the image to determine grain angle. Grain will be used to validate single

and double-sided defects. Additionally, grain direction will be used to determine the precise boundaries of knots.

Product Probability Scheduling

In the present optimization routine, too much emphasis is placed on having absolute knowledge of defects. Indeed, the very nature of wood yields uncertain information about defect features. By presenting the optimization program with only high probability defects, information is lost at the Image Processor. Rather than have the Image Processor make a go/no-go “decision” about a defect candidate, the Image Processor should pass computed probabilities of defects to the Optimizer. Granted, such cases represent less than 8.5% of the total defects as false grain hits and knots over chain raceways. It is further arguable that a small percentage of side-knots are not visible from the top and bottom views of a board. Nevertheless, it is deemed more important to optimize a solution based on complete knowledge about defects than optimize using incomplete, or “missing,” defect information.

Background. Probabilistic reasoning, neural networks, and fuzzy logic go back to the early 1950s, but the technology was generally ignored until the early 1980s. Since then, both neural networks and fuzzy logic have taken off in the research world, and the more recent 1990s has begun to see applications in everything from fighter plane control systems to fuzzy-logic controlled washing machines. Essentially, reasoning with incomplete or “fuzzy” information involves working with mathematical equations. Back

in the late 1950s, only a hand-full of researchers with PhDs in mathematics could actually make these equations do useful things on analog computers. So when switching networks demonstrated their promise in the early 1960s, researchers abandoned writing complex equations for the more simple Boolean algebra. We know today's switching networks as digital computers or PCs, and Boolean algebra as IF-THEN binary logic (known as ladder logic in PLC terminology). Classifying features as only defects follows this IF-THEN binary paradigm.

The world is actually analog and not binary. That is, there are many shades of gray to physical processes. For example, a binary thinker (somebody who only thinks in black and white) would say that a person is either young or old. A fuzzy thinker would say that there are infants and children and teenagers and young adults and adults and elderly and ancient people. A person can be both a young adult and an adult. (Although possible, a person isn't a child and an adult.) When was the last time a car was produced with only two forward gears, 1st gear and overdrive? It just isn't very efficient.

Three compelling reasons are driving the re-birth of probabilistic reasoning: technology, applications, and new algorithms in the form of neural networks and fuzzy logic. Personal computers are now capable of doing analog calculations with real-time (near analog) speed. Our applications are getting harder and are demanding more complex solutions. New biologically inspired research has produced some very robust and flexible mathematical equations and algorithms. Most important, engineers and

mathematicians used to spend months, if not years, trying to understand the dynamics of physical processes that couldn't be visualized in three dimensions. Just try drawing a simple W-X-Y-Z graph in four dimensions (height, length, width, and depth) and you'll appreciate the complexity of modern aerodynamic controls. Fortunately, both neural networks and fuzzy logic can be used to solve problems having more than three dimensions. Also, because we perceive our world in three dimensions, we tend to apply our three dimensional bias to applications involving lots of independent variables. How many economists keep trying to model the stock market with only a few significant indicators? Their predictions haven't been too accurate, on average.

Probabilistic reasoning uses mathematical equations to map input variables to output variables ($y=mx+b$). If you present such equations with any W, X, and Y values, it will calculate the exact value of Z, for example.

Neural networks differ significantly from fuzzy logic in the way the mathematical equations are written. Just like the neurons in our bodies, a neural network is an ensemble of simple threshold-based signal processors called neurodes. In fact, the mathematical laws governing how a neurode processes information was based on studies of actual biological neurons. Just like our neurons -- although much less complex -- neurodes can learn information. It's the learning process or learning algorithm that makes a network of neurodes unique among mathematical equations.

Fuzzy logic doesn't use a learning algorithm. Instead, fuzzy logic is a tool that represents a problem in little two dimensional (X and Y) chunks that we can understand and visualize. It's the ability to represent lots of information simply that makes fuzzy logic unique among mathematical equations.

A Case for Probabilistic Reasoning. Consider the case where the Image Processor correctly finds a knot on the top side of a board AND does not find it on the bottom side because it is covered by a chain raceway. Four chain raceways support the lumber and hide about 3% of the knots from the bottom cameras.

Suppose that knots greater than 1.500" are *double-sided defects*. Knots less than 1.500" may or may not go all the way through a board. So if the Image Processor fails to find one side of a knot AND the knot is less than 1.500", the knot is classified as a *single-sided defect*. The goal is to maximize the value of the lumber. When a board is sawed for single-sided defects, the resulting value of the cut lumber is less than if it had been sawed given a double-sided defects because a false cut decision was made. Miss-classifying defects equates to lost profits.

Because rip saw operators don't measure each knot, it's not just the size of the knot that determines if it's a single- or double-sided defect. In fact, rip saw operators rarely turn a board over to see if a knot is double-sided. Shape or closure of the knot and the contrast between the knot, grain, and the clear wood contribute to determining the probability of each defect class. Recall from fuzzy logic that a knot may have partial

membership in every class. For example, the knot in this case might have 54% membership in *double-sided defects*, 45% in *single-sided defects*, and 1% in the *no defect* class. The *no defect class* is added so the sum of all probabilities is 100%.

The fuzzy logic methodology rejects “crisp” decisions. Optimization processes make a crisp decision each time it classifies a defect using only length. In a hypothetical case, a given knot is 1.450” long. Crisp decisions allow no flexibility for how close the knot is to the crisp 1.500” double-sided defect decision case. What if the knot is 1.499” long? The knot is still a *single-sided defect*. Perhaps 90% of all double-sided knots are greater than 1.500” in length. If we can classify an additional 3% by including shape and contrast information, then we can increase the value of lumber by that 3% as well.

Traditional logic requires multiple levels of IF-THEN rules to fuzzify or soften a crisp decision. For example: IF *length* < 1.500” AND IF *closure* > 85% AND IF *contrast* > 50% THEN *double sided*. Traditional logic gets very complicated very fast. Probabilistic reasoning, like fuzzy logic, takes an n-dimensional approach using multiple mathematical variables to make decisions. Probabilities are calculated by the Image Processor for size, closure, and contrast. The union of these probabilities would then make the crisp decision. In this fashion, an unlimited number of variables (e.g., length/width ratio, board curvature, saw tolerance, misalignment factor, etc.) can contribute to the outcome of a rip solution.

Figure 48 illustrates four different ways the a board may be sawed and the value of each board segment. The two solutions on the left are valid for a double-sided defect, $P=54\%$. The two solutions on the right are valid for a single-sided defect, $P=45\%$. The value of each rip solution is calculated by summing the segments.

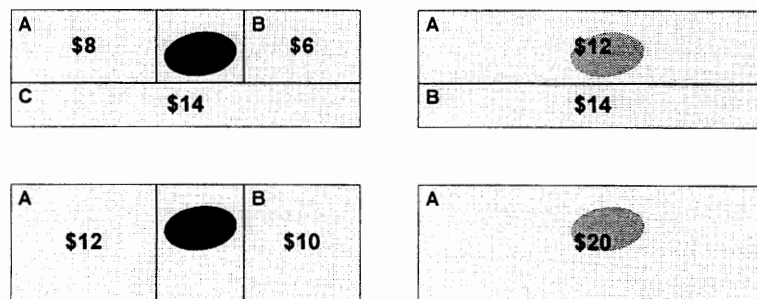


Figure 48. Four ways a board may be cut up when a knot is uncertain.

Figure 49 illustrates a decision tree for the rip solutions of Figure 48. Node 0 is the start, nodes 3-7 are the results. The value of nodes 3-7 are calculated by multiplying the sum of the board segment values with the predecessor probabilities. Each node represents a decision probability, P . The value of node 3 is $(P_1)(P_3)(\sum \$)$ or \$6.30. The value of choosing double-sided defects, node 1, is the sum of the possible outcomes, nodes 3 and 4. Starting at node 0, node 1 is greater than nodes 2 and 7; so the knot is classified as a *double-sided defect*. From node 1, node 3 is greater than node 4; so node 3 is the rip solution. The process of traveling from node 0 to node 1 to node 3 is called probabilistic reasoning.

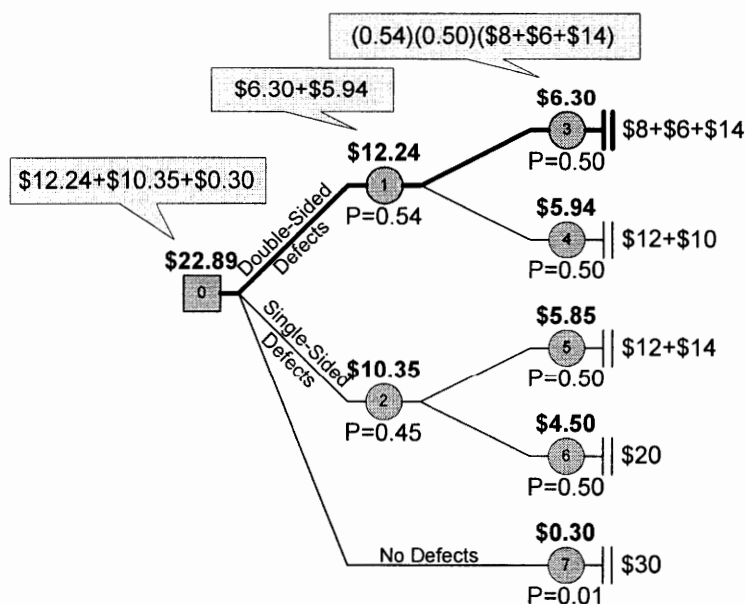


Figure 49. Probability tree for choosing a rip solution given uncertain information about the inclusion of a defect.

A decision tree is a good method to visualize a decision process with less than 10 variables. Fortunately, several fuzzy logic expert-like mathematical techniques exist to eliminate many of the possible decision branches. Probabilistic reasoning is a very fast and robust way to determine a solution given incomplete and uncertain information.

Production Feedback. The Product Value Scheduler will also benefit from additional information. MRP systems possess an inherent ability to reevaluate the validity of all open shop-order and purchase-order due dates, and thus keep these due dates up to date. To determine process times requires understanding and developing a strategy of predicting material availability. The Board Value Optimizer provides a mechanism to collect, analyze, and predict the availability of WIP inventory for each

grade of board. While processing lumber, the Board Value Optimizer will develop a model for planning material availability.

The Board Value Optimizer adds the dimension of time to inventory status data. As a step beyond the perpetual inventory control concept of *what* and *how much to produce* in the present system, the Machine Vision System will answer the crucial question of *when will it be produced*. *When* will the WIP inventory be available? The inventory planner needs this information to prevent a stockout or shortage and to set production schedules. The Machine Vision System approaches process control with the following two principles: 1) optimizing use and value for production; and 2) calculation of WIP inventory availability over time.

The Board Value Optimizer will forecast the availability of WIP inventory so maximum utilization of clear may be taken on a board-by-board basis. This process is exemplified in the slicer cut paradigm. MRP calculates WIP inventory availability and time-phases the value and utilization parameters for the optimization process. In this way, the evolution of an MRP-based optimization system is a logical extension of the present weight-based perpetual inventory control system. The goal of adding the MRP time-phasing dimension is to control the availability of each piece of WIP inventory so maximum value may be obtained from each board as the system follows a master production schedule. The MRP concept presupposes that lead times to produce WIP inventory can be determined by evaluating past results.

An evaluation of WIP inventory produced from lumber leads to the thesis that WIP inventory will be produced in desired quantities given an optimization method based solely on value. Manipulating the value of individual WIP inventory through weights will produce the desired production mix, on average. However, this thesis does not allow dynamic flexibility in terms of production utilization. The strategy for cutting lumber must be to maximize the utilization of clear for production – otherwise, too much material is lost as usable waste (sawdust).

MRP and coverage theory add to the maximum value strategy by minimizing usable waste generated from each board relative to the time-phased demands of production. Following this method, it makes sense to produce long-length products whenever possible because the next available long-length product may not be realized in the short term. In the n-dimensional optimization space, maximum *value* is a complex function of both product availability and WIP inventory value.

Because the Board Value Optimizer determines the physical availability of WIP inventory, the Rip & Cut process becomes an order expediting process. Under the present manual system, value is achieved at the cost of availability. Hence, large quantities of safety or buffer WIP inventory must be maintained to fulfill lumpy-availability of long-length products. The proposed Machine Vision System will reduce the dependence on perpetual inventory control (long-term averaging) through the discrete utilization of lumber for availability with respect to maximum utilization.

REFERENCES

NUMBER

1. Wengert, Gene. "Wood Technology Seminar." Pacific Power address. Albany, OR. 16 December 1993.
2. Western Wood Products Association. Standard Grading Rules for Western Lumber. 1991.
3. Penman, David, Olof Olsson, and Chris Bowman. "Automatic Inspection of Reconstituted Wood Panels for Surface Defects." Paper. Industrial Research Ltd. Auckland, New Zealand. 1995.
4. Sorensen, Jean. "Color-Grain Scanner Yields Higher-Value Products." Forrest Industries. (September, 1990): 41-2.
5. Michigan State University. n.p. n.d., qtd. in Wengert, Gene. "Wood Technology Seminar." Pacific Power address. Albany, OR. 16 December 1993.
6. Bruner, Charles C., et. al. "Enhancing Color-Image Data for Wood-Surface Feature Identification." Proceedings. 5th International Conference on Scanning Technology & Process Control For the Wood Products Industry. Atlanta, GA. 25-7 October, 1993.
7. Bruner, Charles C., et. al. "An Evaluation of Color Spaces for Detecting Defects in Douglas-Fir Veneer." Industrial Metrology. 2 (1992): 169-84.
8. Miller, D. G. "Detection of Rot In Wood by Electronic X-Ray Fluoroscopy." British Columbia Lumberman. (October, 1964)
9. Szymani and McDonald. "Defect Detection in Lumber — State of the Art." Forest Products Journal. (November, 1981)
10. Conners, et. al. "Code Identifying and Locating Surface Defects in Wood — Part of An Automated Lumber Processing System." IEEE Transactions on Pattern Analysis and Machine Intelligence. Vol. TAMI-5. No. 6. (November, 1983)

11. Conners. "A Prototype Software System Locating and Identifying Surface Defects in Wood." Proceedings. Vol. 1. Seventh International Conference on Pattern Recognition. Montreal, Canada. July 30-Aug 2, 1984.
12. Maristany, Alberto G., et. al. "Classifying Wood-Surface Features Using Dichromatic Reflection." Proceedings. Vol. 1836. SPIE: Optics in Agriculture and Forestry. Boston, MA. (16-7 November 1992): 56-64.
13. Brunner, Charles C., et. al. "Using Color In Machine Vision Systems For Wood Processing." Wood and Fiber Science. 22(4). 1990. 413-28.
14. Arden, Torence J. "Color Sorting of Lumber." United States Patent No. 4,992,949. 12 February, 1991.
15. Sanglert, Benkt. "Producing Signals Denoting Location of Edges of A Finished Surface on A Partially Finished Workpiece." United States Patent No. 3,886,372. 27 May 1975.
16. Dahlström, Claes, et. al. "Method and Device for Optical Scanning of A Series of Transversal Dimensional Values at A Board or Plank." United States Patent No. 3,983,403. 12 June 1975.
17. Maxey, Carl W. "Automatic Wane Detector." United States Patent No. 4,186,310. 29 January 1980.
18. Chasson, Leon H. "Method and Apparatus for Automatically Processing A Workpiece Employing Calibrated Scanning." United States Patent No. 4,188,544. 12 February 1980.
19. Bruner, Charles C., et. al. "An Evaluation of Color Spaces for Detecting Defects in Douglas-Fir Veneer." Industrial Metrology. Vol. 2. New York: Elsevier, 1992. 169-84.
20. Bruner, Charles C., et. al. "Enhancing Color-Image Data for Wood-Surface Feature Identification." Proceedings. Fifth International Conference on Scanning Technology & Process Control For The Wood Products Industry. Atlanta, GA. 25-7 October 1993.

21. Lockett, James F. "Foreign Object Discriminator for Sorting Apparatus." United States Patent No. 4,260,062. 7 April 1981.
22. Maristany, Alberto G., et. al. "Exploiting Local Color information for Defect Detection on Douglas-Fir Veneer." Proceedings. Fourth International Conference on Scanning Technology In the Wood Industry. Burlingame, CA. 28-9 October 1991.
23. McConnell, Robert K. Jr., Ronald A. Messa, and Henry H. Blau, Jr. "Color Machine Vision." Proceedings. Sensors Expo: Boston. Boston, MA. 16-8 May 1995.
24. Giease, P. J. and K. A. McDonald. "OPTYLD—A Multiple Rip-First Computer Program to Maximize Cutting Yields." Forest Products Laboratory Research Paper FPL-412. (1982).
25. Giease, P. J. and J. D. Danielson. "CROMAX—A Crosscut-First Computer Simulation Program to Determine Cutting Yield." Forest Products Laboratory General Technical Report FPL-38. (1983).
26. Gilmore, P. C. and R. F. Gomory. "The theory and Computation of Knapsack Functions." Operations Research. Vol. 14. (1966): 1045-74.
27. Hallock, H. and P. J. Giease. "Cutting Yields from Standard Hardwood Lumber Grades When Gang Ripping." USDA Forest Service Research Paper FPL-370.
28. McDonald, K. A., P. J. Giease, and R. O. Woodfin. "Maximum Cutting Yields for 6/4 Ponderosa Pine Shop Lumber." Forest Products Laboratory Research Paper FPL-437. (1983).
29. Salkin, H. M. and C. A. Dekleyver. "The Knapsack Problem: A Survey." Naval Research Log Quarterly. Vol. 22, No. 7. (1975): 127-44.
30. Azarm, S. et. al. "Heuristic Optimization of Rough-Mill Yield With Production Priorities." ASME Journal of Engineering for Industry. Vol. 113, No. 1. (1991): 108-15.

31. Christofides, N. and C. Whitlock. "An Algorithm for Two Dimensional Cutting Problem." Operations Research. Vol. 25, No. 1. (1977): 30-44.
32. Kolesar, P. J. "A Branch and Bound Algorithm for the Knapsack Problem." Management Science. Vol. 13. (1967): 723-35.
33. Sarin, S. C. "Two-Dimensional Stock Cutting Problems and Solution Methodologies." ASME Journal of Engineering for Industry. Vol. 105, No. 3. (1983): 155-60.
34. Hall, Peter. Introduction to the Theory of Coverage Processes. New York: John Wiley, 1988.
35. Hasdorff, Lawrence. Gradient Optimization and Nonlinear Control. New York: John Wiley, 1976.
36. Kohonen, Teuvo. Self-Organization and Associative Memory. New York: Springer-Verlag, 1989.
37. Kapralski, Adam. Sequential and Paralled Processing in Depth Search Machines. New Jersey: World Scientific, 1994.

BIBLIOGRAPHY

FUZZY LOGIC

Klir, George J. and Tina A. Folger. Fuzzy Sets, Uncertainty, and Information. Englewood Cliffs: Prentice Hall, 1988.

Kosko, Bart. Neural Networks and Fuzzy Systems. Englewood Cliffs: Prentice Hall, 1992.

IMAGE PROCESSING

Banks, Stephen. Signal Processing, Image Processing and Pattern Recognition. New York: Prentice Hall, 1990.

Gonzalez, Rafael C. and Paul Wintz. Digital Image Processing. 2nd. ed. Reading: Addison-Wesley, 1987.

McGarry, John. 900 Series Programmer's Guide. Portland, OR: Acumen, Inc., 1991.

Myler, Harley R. and Arthur R. Weeks. The Pocket Handbook of Image Processing Algorithms In C. Englewood Cliffs: Prentice Hall, 1993.

Russ, John C. The Image Processing Handbook. 2nd. Ed. Ann Arbor: CRC Press, 1995.

OPTIMIZATION

Cormen, Thomas H., Charles E. Leiserson, and Ronald L. Rivest. Introduction to Algorithms. Cambridge: MIT Press, 1992.

Lapin, Lawrence L. Probability and Statistics for Modern Engineering. Monterey: Brooks/Cole, 1983.

Orlicky, Joseph. Materials Requirements Planning. New York: McGraw-Hill, 1975.

Page, G. William and Carl V. Patton. Quick Answers to Quantitative Problems. Boston: Academic, 1991.

Papoulis, Athanasios. Probability, Random Variables, and Stochastic Processes. New York: McGraw-Hill, 1991.

Pfeiffer, Paul E. Concepts of Probability Theory. 2nd. ed. New York: Dover, 1978.

WOOD PRODUCTS

Oberg, Fred R. Heavy Timber Construction. American technical Society, 1963.

APPENDIX A

IMAGE PROCESSOR COMPUTER C CODE

This appendix presents the computer software programming functions used to perform color correlation on the Image Processor. All programming code is written in the C language and is designed to be compiled for and implemented on the AT&T DSP32C computer platform. The first programming function, called "init_tables," builds the sigmoid-shaped color correlation look-up table (LUT). The second and third functions, called "color_rsums" and "color_csums," compute the average grain color of two wood images stored in memory. These two functions are implemented on the Rockwell correlator on the Acumen 900 Image Processor, and were written by John McGarry of Acumen, Inc. The fourth function, called "colcor_images," performs the color correlation on the wood images and stores the resultant as an image in memory.

```
/******  
  
////FUNCTION:  init_tables  
////SCOPE:    SUBSYSTEM  
////PURPOSE:  
//  Initialize tables used in the image conversion routine.  
//  Generate the color splitting LUT and color correlation LUT.  
//  
////INPUTS:  
//  Called from main.  
//  Parameters:  NAME      TYPE      DESC  
//              ----      - - - -  
//              oput      i24       not used  
//              iput      i24       not used  
//  
////ASSUMPTIONS:  
//  Memory is free.  
//
```

```

////OUTPUTS:
//   Return Value:  TYPE  DESC
//                   ----  ----
//                   HAND  0
//
////AUTHOR:
//   John Goulding, 1996.
//

*****/

HAND init_tables(oput,iput)
i24  oput;
i24  iput;
{
    register i24 i, tval, cval, sval;
    register float scale, infpt;

    splt_table=(char *)vmem_alloc(1,0x20000,CHAR_TYPE);
    corr_table=(char *)vmem_alloc(1,0x200,CHAR_TYPE);

    /*****/
    /* COLOR CHANNEL SEPARATION */
    /*****/

    /* This routine makes the color splitting table */
    /* It is a 1:1 mapping to perform a 5-bit-shift */
    /* This is a 17-bit LUT */
    for(i=0; i<0x20000; i++)
    {
        tval = i&0x18000;

        if(tval==0x0)
        {
            /* Mask lowest 5-bits */
            splt_table[i] = i&0x1F; //blue
        }
        else if(tval==0x8000)
        {
            /* Shift upper 5-bits and mask */
            splt_table[i] = (i>>10)&0x1F; //red
        }
        else
        {
            /* Shift middle 5-bits and mask */
            splt_table[i] = (i>>5)&0x1F; //green
        }
    }

    /*****/
    /* SIGMOID CORRELATION FILTER */
    /*****/

    /* Set the inflection point */
    /* Use lower values for end cameras */
    if(vpe_num==1 || vpe_num==8)

```

```

{
    infpt = 5.0;
}
else
{
    infpt = 5.5;
}

/* This routine makes the sigmoidal correlation table */
/* This is an 9-bit LUT */
for (i=0; i<0x200; i++) //0x1FF = 1111 1111
{
    /* The four bit transition value */
    tval = ((i>>4)&0x1E)+1; //0x1E = 11110

    /* The five bit rgb color value */
    cval = i&0x1F; //0x1F = 11111

    /* Shift the sigmoid right (add more leading zeros) */
    if(cval>=1 && tval>=0x8)
    {
        cval--;
    }

    /* Shift the sigmoid right again for the darker end cameras */
    if(cval>=1 && (vpe_num!=1 || vpe_num!=8) && tval>=0x10)
    {
        cval--;
    }

    /* Shift the sigmoid to the right again on max transition */
    if(cval>=1 && tval>=0x18)
    {
        cval--;
    }

    /* Generate the sigmoid shape */
    /* Check for saturation */
    if(cval>=tval)
    {
        corr_table[i] = 0x7; //saturated 1/6 above average color
    }
    /* Construct sigmoid */
    else
    {
        /* The inflection point is linear for greater contrast */
        sval = tval-(i24)((float)tval/infpt);

        /* Below the inflection point */
        if(cval<=sval)
        {
            /* The sigmoid shape is nonlinear cubed */
            scale = 7.0/(float)(tval*sval*sval);
            corr_table[i] = (i24)((float)(cval*cval*cval)*scale)&0x7;
        }
        /* Above the inflection point */
        else
        {

```

```

        /* The sigmoid shape is nonlinear squared */
        scale = infpt/(float)tval;
        scale = (7.0-((7.0*(float)sval)/(float)(tval)))*scale*scale;
        sval = tval - cval;
        corr_table[i] = 0x7-((i24)((float)(sval*sval)*scale)&0x7);
    }
}

return(0);
}

```

```

/*****

```

```

/////FUNCTION: color_rsums
/////SCOPE:      SUBSYSTEM
/////PURPOSE:
//  Compute the average grain color using model.
//  Perform operation row-wise on correlator.
//
/////INPUTS:
//  Called from colcor_images.
//  Parameters:  NAME      TYPE      DESC
//              -----
//              Row0      i24      Starting row
//              Col0      i24      Starting column
//              High      i24      Height of area
//              Wide      i24      Width of area
//              Offset    i24      Offset area
//
/////ASSUMPTIONS:
//  Correlator is free.
//
/////OUTPUTS:
//  Return Value:  NAME      TYPE      DESC
//              -----
//              pColorSum  i24      Resultant
//
/////AUTHOR:
//  John McGeary, Acumen, Inc., 1995.
//

```

```

*****/

```

```

void color_rsums(pColorSum,Row0,Col0,High,Wide,Offset)
register COLORSUM *pColorSum;
register i24      Row0;
register i24      Col0;
register i24      High;
register i24      Wide;
register i24      Offset;
{

```

```

    register i24  i, Row, Sum;
    register float scale=1.0/Wide;

```

```

/*-----

```

```

-- Set the model transform coefficients. | 256*cos 256*sin |
--                                     |-256*sin 256*cos |
-----*/

K00 = 256; K01 = 0;
K10 = 0;   K11 = 256;

/*-----
-- Set the model stop bit for window width.
-----*/

*((char *) (hRedMod+(4*Wide)-1)) |= 0x80;
*((char *) (hGrnMod+(4*Wide)-1)) |= 0x80;
*((char *) (hBluMod+(4*Wide)-1)) |= 0x80;

/*-----
-- Compute Red, Green, and Blue color sum array.
-----*/

High=(High*0.5)+Offset;
for( i=Offset, Row=Row0; i<High; Row+=2, i++ )
{
    *((i24 *)MOD_PTR) = hRedMod>>2;
    vdsp_corr_run(Row, Col0); vdsp_corr_sums();
    pColorSum[i].Red = *((i24 *)sch_EIL)*scale;

    *((i24 *)MOD_PTR) = hGrnMod>>2;
    vdsp_corr_run(Row, Col0); vdsp_corr_sums();
    pColorSum[i].Grn = *((i24 *)sch_EIL)*scale;

    *((i24 *)MOD_PTR) = hBluMod>>2;
    vdsp_corr_run(Row, Col0); vdsp_corr_sums();
    pColorSum[i].Blu = *((i24 *)sch_EIL)*scale;
}

/*-----
-- Clear the model stop bits.
-----*/

*((char *) (hRedMod+(4*Wide)-1)) &= 0x7F;
*((char *) (hGrnMod+(4*Wide)-1)) &= 0x7F;
*((char *) (hBluMod+(4*Wide)-1)) &= 0x7F;
}

/*****
////FUNCTION: color_csums
////SCOPE:     SUBSYSTEM
////PURPOSE:
//   Compute the average grain color using model.
//   Perform operation column-wise on correlator.
//
////INPUTS:
//   Called from colcor_images.
//   Parameters:  NAME      TYPE      DESC
//               -----
//               Row0      i24       Starting row

```



```

//          Col0      i24      Starting column
//          High       i24      Height of area
//          Wide       i24      Width of area
//          Offset     i24      Offset area
//
//////ASSUMPTIONS:
// Correlator is free.
//
//////OUTPUTS:
// Return Value:  NAME      TYPE      DESC
//                -----
//                pColorSum  i24      Resultant
//
//////AUTHOR:
// John McGeary, Acumen, Inc., 1995.
//

*****/

void color_csums(pColorSum,Row0,Col0,High,Wide,Offset)
register COLORSUM *pColorSum;
register i24      Row0;
register i24      Col0;
register i24      High;
register i24      Wide;
register i16      Offset;
{
    register i24  i, Col, Sum;
    register float scale=1.0/High;

/*-----
-- Set the model transform coefficients. | 256*cos  256*sin |
--                                     |-256*sin  256*cos |
-----*/

    K00 = 0;      K01 = -256;
    K10 = 256;    K11 = 0;

/*-----
-- Set the model stop bit for window width.
-----*/

    *((char *) (hRedMod+(4*High)-1)) |= 0x80;
    *((char *) (hGrnMod+(4*High)-1)) |= 0x80;
    *((char *) (hBluMod+(4*High)-1)) |= 0x80;

/*-----
-- Compute Red, Green, and Blue color sum array.
-----*/

    Wide=(Wide*0.5)+Offset;
    for( i=Offset,Col=Col0; i<Wide; Col+=2,i++ )
    {
        *((i24 *)MOD_PTR) = hRedMod>>2;
        vdsp_corr_run(Row0,Col); vdsp_corr_sums();
        pColorSum[i].Red = *((i24 *)sch_EIL)*scale;

        *((i24 *)MOD_PTR) = hGrnMod>>2;

```

```

    vdsp_corr_run(Row0,Col); vdsp_corr_sums();
    pColorSum[i].Grn = *((i24 *)sch_EIL)*scale;

    *((i24 *)MOD_PTR) = hBluMod>>2;
    vdsp_corr_run(Row0,Col); vdsp_corr_sums();
    pColorSum[i].Blu = *((i24 *)sch_EIL)*scale;
}

/*-----
-- Clear the model stop bits.
-----*/

*((char *) (hRedMod+(4*High)-1))&=0x7F;
*{(char *) (hGrnMod+(4*High)-1)}&=0x7F;
*((char *) (hBluMod+(4*High)-1))&=0x7F;
}

/*****
////FUNCTION: colcor_images
////SCOPE:     SUBSYSTEM
////PURPOSE:
//  Color correlate the leading (buffer 0) and trailing (buffer 1)
//  edge images into a grayscale enhanced contrast image in
//  converted (buffer 0).
//
////INPUTS:
//  Called from process_images.
//  Parameters:  NAME      TYPE      DESC
//              -----
//              x0         i16       first row to be converted
//              x1         i16       last row to be converted
//              y0         i16       first column to be converted
//              y1         i16       last column to be converted
//
////ASSUMPTIONS:
//  Correlator is free.
//
////OUTPUTS:
//  Return Value:  TYPE  DESC
//              -----
//              i24   0
//
////AUTHOR:
//  John Goulding, 1996.
//
*****/

i24 colcor_images(x0,x1,y0,y1)
i16 x0; //starting x coordinate
i16 x1; //ending x coordinate
i16 y0; //starting y coordinate
i16 y1; //ending y coordinate
{
    i24 i, wide, shift;

```

```

extern i24 colcor_sub();
register FPTRvoid corsub=(FPTRvoid)0xffff800;

COLORSUM *pRSum;
COLORSUM *pCSum;
char *tempmem;

/* Wait for external processes */
vvvid_wait(0,0);
vvvid_wait(1,0);

/* Assign memory for row and column statistics */
pRSum=(COLORSUM *)vvmem_alloc(1,512*sizeof(COLORSUM),I24_TYPE);
pCSum=(COLORSUM *)vvmem_alloc(1,512*sizeof(COLORSUM),I24_TYPE);
tempmem=(char *)vvmem_alloc(1,0x800,CHAR_TYPE); //0 is GP memory

/* Initialize operation variables */
wide = x1-x0;
y0 = (y0+3)&0xFFFC;
y1 = y1&0xFFFC;
shift = y1-y0;

/* Find color correlation statistics for leading-edge lit image */
vvimg_buffer(0);
color_rsums(pRSum,x0,y0,wide,shift,0);
color_csums(pCSum,x0,0,wide,512,0);

/* Find color correlation statistics for trailing-edge lit image */
vvimg_buffer(1);
color_rsums(pRSum,x0,y0,wide,shift,256);
color_csums(pCSum,x0,0,wide,512,256);

/* Initialize operation variables */
wide = wide>>1;
shift = shift>>1;
y0 = y0>>1;

/* Perform the color correlation */
vvmem_memcpy(tempmem, (FPTRvoid)0xffff000,0x800);
vvmem_memcpy(corsub,colcor_sub,0x700);
put_pcw(0x8003);
(*colcor_sub)(x0,x1,y0,wide,shift,pRSum,pCSum);
vvmem_memcpy((FPTRvoid)0xffff000,tempmem,0x800);
put_pcw(0x8033);

/* Free the memory */
/* The image was modified */
vvmem_free(pRSum);
vvmem_free(pCSum);
vvmem_free(tempmem);

/* Reday to perform shape correlation */
return(0);
}

/* Color correlation routine */
/* Do memcpy and run on DSP */
colcor_sub(x0,x1,y0,Wide,Shift,pRowSum,pColSum)

```

```

i16 x0,x1,y0;
i24 Wide,Shift;
COLORSUM *pRowSum;
COLORSUM *pColSum;
{
    /* Nine DSP registers */
    register i24 reg1, reg2, reg3, reg4, reg5, reg6, reg7, reg8, reg9;

    /* External variables */
    i24 Row1, Row10, Offset;
    i24 rowRed, rowGrn, rowBlu;
    i24 avgRed, avgGrn, avgBlu;
    i24 *avgGColor=(i24 *)0x400;
    i24 *avgColors=(i24 *)0xffff000;

    /* Determine the global grain color statistics */
    /* Divide the area into 64 global regions */
    for(reg1=0; reg1<65; reg1+=64)
    {
        /* Initialize correlation parameters */
        /* Perform row-wise operations */
        reg2 = 0;
        reg4 = reg1*4;
        reg5 = reg4 + Wide;
        reg6 = 0;
        reg7 = 0;
        reg8 = 0;
        reg9 = (i24)pRowSum + reg1*4*sizeof(COLORSUM);

        /* Sum each color channel statistic across rows */
        for(; reg4<reg5; reg4++)
        {
            reg6 += ((COLORSUM *)reg9)->Red;
            reg7 += ((COLORSUM *)reg9)->Grn;
            reg8 += ((COLORSUM *)reg9)->Blu;
            reg9 += sizeof(COLORSUM);
            reg2++;
        }

        /* Initialize correlation parameters */
        /* Perform column-wise operations */
        reg4 = reg1*4;
        reg5 = reg4 + Shift;
        reg9 = (i24)pColSum + reg1*4*sizeof(COLORSUM)
            + y0*sizeof(COLORSUM);

        /* Sum each color channel statistic across rows */
        for(; reg4<reg5; reg4++)
        {
            reg6 += ((COLORSUM *)reg9)->Red;
            reg7 += ((COLORSUM *)reg9)->Grn;
            reg8 += ((COLORSUM *)reg9)->Blu;
            reg9 += sizeof(COLORSUM);
            reg2++;
        }

        /* Determine the global average grain color by region */
        avgGColor[reg1] = (16*reg6)/reg2;
    }
}

```

```

    avgGColor[reg1+1] = (16*reg7)/reg2;
    avgGColor[reg1+2] = (16*reg8)/reg2;
}

/*****
/* SIGMOID COLOR CORRELATION */
*****/

/* Perform the color correlation over the image */
Row10 = IBPTR0+(x1*512+512)*PSTP; //stop pixel
reg1 = (IBPTR0+x0*512*PSTP);      //start pixel

/* Operate by coarse image rows */
for( Row1=0; reg1<Row10; Row1+=4 )
{
    /* Divide the row into 64 sub-regions */
    for( Offset=0; Offset<65; Offset+=64 )
    {
        /* Establish the sub-region parameters by row */
        if( Row1+6>Wide )
        {
            /* Maintain constant width */
            reg5 = Wide+Offset*4;
            reg4 = reg5-8;
        }
        else
        {
            if( Row1==0 )
            {
                /* Maintain constant width */
                reg4 = Offset*4+Row1;
                reg5 = reg4+8;
            }
            else
            {
                /* Maintain constant width */
                reg4 = Offset*4+Row1-2;
                reg5 = reg4+8;
            }
        }
    }

    reg6 = 0;
    reg7 = 0;
    reg8 = 0;
    reg9 = (i24)pRowSum + reg4*sizeof(COLORSUM);

    /* Sum the grain color stats across the rows */
    for(; reg4<reg5; reg4++)
    {
        reg6 += ((COLORSUM *)reg9)->Red;
        reg7 += ((COLORSUM *)reg9)->Grn;
        reg8 += ((COLORSUM *)reg9)->Blu;
        reg9 += sizeof(COLORSUM);
    }

    /* Save the local grain color stats by row */
    rowRed = reg6;

```

```

rowGrn = reg7;
rowBlu = reg8;

/* Set the column color stats */
reg4 = Offset*3;
reg5 = Offset*4;
reg6 = reg5+8;
reg7 = reg4+64;
avgRed = avgGColor[Offset];
avgGrn = avgGColor[Offset+1];
avgBlu = avgGColor[Offset+2];

/* Find 64 column stats */
/* Convert the stats to sigmoid selector bits */
for(; reg4<reg7; reg4++, reg5+=4, reg6+=4 )
{
    /* Determine stats for blue grain */
    reg9 = 0;
    reg8 = reg5;
    reg2 = (i24)&(pColSum[reg5].Blu);

    /* Sum the grain color column-wise */
    for(; reg8<reg6; reg8++, reg2+=sizeof(COLORSUM) )
    {
        reg9 += *(i24 *)reg2;
    }

    /* Add the row blue grain color */
    reg9 += rowBlu;

    /* IF local blue < average blue */
    if(reg9<avgBlu)
    {
        reg9 += (avgBlu-reg9)>>1;
    }

    /* Set the correlation selector bits directly */
    avgColors[reg4] = reg9&0x1E0; /* blue */

    /* Determine stats for red grain */
    reg9 = 0;
    reg8 = reg5;
    reg2 = (i24)&(pColSum[reg5].Red);

    /* Sum the grain colors column-wise */
    for(; reg8<reg6; reg8++, reg2+=sizeof(COLORSUM) )
    {
        reg9 += *(i24 *)reg2;
    }

    /* Add the row red grain color */
    reg9 += rowRed;

    /* IF local red < average red */
    if( reg9<avgRed )
    {
        reg9 += (avgRed-reg9)>>1;
    }
}

```

```

/* Set the correlation selector bits directly */
avgColors[reg4+64] = reg9&0x1E0; /* red */

/* Determine stats for green grain */
reg9 = 0;
reg8 = reg5;
reg2 = (i24)&(pColSum[reg5].Grn);

/* Sum the grain colors column-wise */
for(; reg8<reg6; reg8++, reg2+=sizeof(COLORSUM) )
{
    reg9 += *(i24 *)reg2;
}

/* Add the row green grain color */
reg9 += rowGrn;

/* IF local red < average green */
if( reg9<avgGrn )
{
    reg9 += (avgGrn-reg9)>>1;
}

/* Set the correlation selector bits directly */
avgColors[reg4+128] = reg9&0x1E0; /* green */

/* Adjust for end case */
if( reg4==0 || reg4==62 || reg4==192 || reg4==254 )
{
    reg5 -= 2;
    reg6 -= 2;
}
}

}

/*****/
/* COLOR CORRELATION */
/*****/

/* For every 8 rows */
for( reg5=0; reg5<8; reg5++ )
{
    /* Assume saturation on start */
    reg9 = 0x7FFF;

    /* For 64 columns */
    for( reg8=0; reg8<64; reg8++ )
    {
        /* Read the average color */
        reg7 = (i24)&(avgColors[reg8]);

        /* For 8 pixels each yields 512 wide */
        for( reg3=0; reg3<4; reg3++, reg1+=PSTP )
        {
            /* Convert the leading edge image */
            /* Read the pixel from image memory */

```

```

reg4 = *(i24 *)reg1;

/* Determine outliers via threshold */
/* Preserve pixel-to-pixel gradient */
if( (reg4>0x6BFF && reg9<0x17FF)
    || (reg4<0x17FF && reg9>0x6BFF) )
{
    /* Correlate the red pixels */
    reg9 = ((i24)corr_table[(i24)splt_table[reg4]|
        *(i24 *)reg7]<<11)&0x7C00;

    /* Correlate the green pixels */
    reg4 = reg4|0x8000; //green
    reg9 += ((i24)corr_table[(i24)splt_table[reg4]|
        *(i24 *)reg7+64*sizeof(i24)]<<6)&0x3E0;

    /* Correlate the blue pixels */
    reg4 = reg4|0x10000; //blue
    reg9 += ((i24)corr_table[(i24)splt_table[reg4]|
        *(i24 *)reg7+128*sizeof(i24)]<<1)&0x1F;
}
/* Average pixels when no gradient */
else
{
    /* Start with average */
    reg9 &= 0x7BDE;
    reg9 /= 2;

    /* Correlate the red pixels */
    reg9 += ((i24)corr_table[(i24)splt_table[reg4]|
        *(i24 *)reg7]<<10)&0x7C00;

    /* Correlate the green pixels */
    reg4 = reg4|0x8000; //green
    reg9 += ((i24)corr_table[(i24)splt_table[reg4]|
        *(i24 *)reg7+64*sizeof(i24)]<<5)&0x3E0;

    /* Correlate the blue pixels */
    reg4 = reg4|0x10000; //blue
    reg9 += (i24)corr_table[(i24)splt_table[reg4]|
        *(i24 *)reg7+128*sizeof(i24)]];
}

/* converted color image in trailing edge buffer */
*(i24 *)reg1+IBPTR1-IBPTR0 = (i24)reg9;

/* Put results in leading edge buffer */
reg4 = 1;
reg4 += (reg9&0x7C00)>>11;
reg4 += (reg9&0x3E0)>>6;
reg4 += (reg9&0x1F)>>1;

/* Check for saturation */
if( reg4>0x1F )
{
    /* Save as saturated */
    *(i24 *)reg1 = 0x1F;
}

```



```

else
{
    /* Save sigmoid result directly */
    *(i24 *)reg1 = (i24)reg4;
}

/* Convert the trailing edge image */
/* Read the pixel from image memory */
reg1 += PSTP;
reg4 = *(i24 *) (reg1+IBPTR1-IBPTR0);

/* Determine outliers via threshold */
/* Preserve pixel-to-pixel gradient */
if( (reg4>0x6BFF && reg9<0x17FF)
    || (reg4<0x17FF && reg9>0x6BFF) )
{
    /* Correlate the red pixels */
    reg9 = ((i24)corr_table[(i24)splt_table[reg4]|
        *(i24 *)reg7]<<11)&0x7C00;

    /* Correlate the green pixels */
    reg4 = reg4|0x8000; //green
    reg9 += ((i24)corr_table[(i24)splt_table[reg4]|
        *(i24 *) (reg7+64*sizeof(i24))]<<6)&0x3E0;

    /* Correlate the blue pixels */
    reg4 = reg4|0x10000; //blue
    reg9 += ((i24)corr_table[(i24)splt_table[reg4]|
        *(i24 *) (reg7+128*sizeof(i24))]<<1)&0x1F;
}
/* Average pixels when no gradient */
else
{
    /* Start with average */
    reg9 &= 0x7BDE;
    reg9 /= 2;

    /* Correlate the red pixels */
    reg9 += ((i24)corr_table[(i24)splt_table[reg4]|
        *(i24 *)reg7]<<10)&0x7C00;

    /* Correlate the green pixels */
    reg4 = reg4|0x8000; //green
    reg9 += ((i24)corr_table[(i24)splt_table[reg4]|
        *(i24 *) (reg7+64*sizeof(i24))]<<5)&0x3E0;

    /* Correlate the blue pixels */
    reg4 = reg4|0x10000; //blue
    reg9 += (i24)corr_table[(i24)splt_table[reg4]|
        *(i24 *) (reg7+128*sizeof(i24))];
}

/* converted color image in trailing edge buffer */
*(i24 *) (reg1+IBPTR1-IBPTR0) = (i24)reg9;

/* Put results in leading edge buffer */
reg4 = 1;
reg4 += (reg9&0x7C00)>>11;

```

```

reg4 += (reg9&0x3E0)>>6;
reg4 += (reg9&0x1F)>>1;

/* Check for saturation */
if( reg4>0x1F )
{
    /* Save as saturated */
    *(i24 *)reg1 = 0x1F;
}
else
{
    /* Save sigmoid result directly */
    *(i24 *)reg1 = (i24)reg4;
}

    } //for 8 pixel contiguous pixels from every other image
    } //for 64 columns in 8-row groups (512 pixel columns)
    } //for 8-row color correlation (80 rows max)
} //for row-wise image color correlation (640 pixel rows)

return(0);
}

```

APPENDIX B

EXAMPLES OF COLOR CORRELATED WOOD IMAGES

This appendix presents eighteen examples of color correlated wood images. Each wood image was color correlated using the algorithms presented in Appendix A. Each wood image contains at least one knot defect. The knot defects and other defect candidates are black pixels in the color correlated images. Clear and clear-grain non-defect pixels are white pixels in the color correlated images. The various species of wood and the various defect types shown are as follows:

1. Solid red knots in Arizona White Pine;
2. Core defects in Arizona White Pine;
3. Spiked ring knot in Oregon Ponderosa Pine;
4. Core defects in Oregon Ponderosa Pine;
5. Not solid ring knot in Oregon Ponderosa Pine;
6. Knot with high-angle grain in Oregon Ponderosa Pine;
7. Bird's-Eye knots with knot in Oregon Ponderosa Pine;
8. Not solid bark-edge knot in Oregon Ponderosa Pine;
9. Knot in heartwood and sapwood in Ponderosa Pine;
10. Core defects in rapid-growth Pine;
11. Solid red knot in weathered and dirty old-growth Pine;
12. Red ring knot with blue stain in Pine;
13. High-angle spike knot in Pine;
14. Knot with growth rings in Pine;
15. Long spike knot, pitch pocket, and pith in Pine;
16. Low-angle spike knot in Pine;
17. Ring knot with sap stain in Pine;
18. Knot with sap stain in Pine;

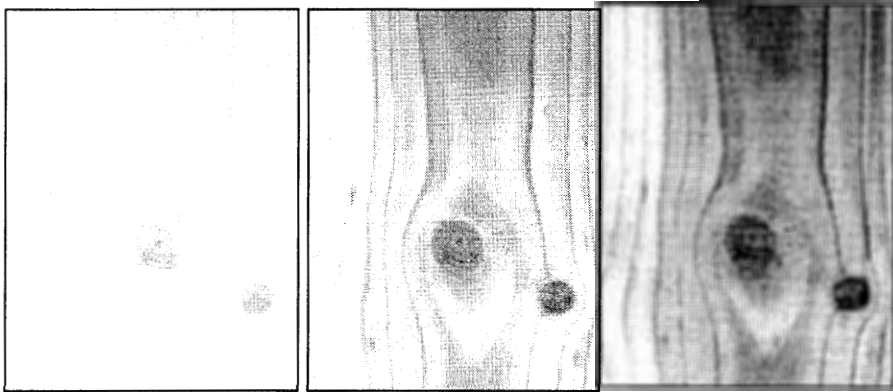


Figure 50. Image of solid red knots in Arizona White Pine wood shown as red, green, and blue color channel images, respectively.

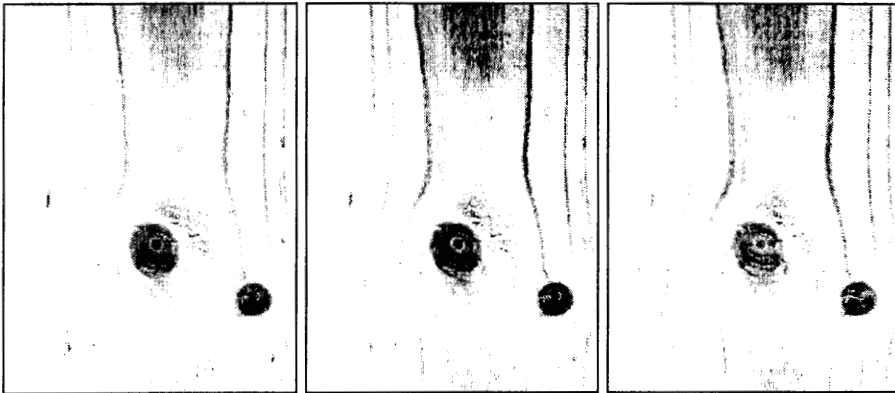


Figure 51. Color correlated image of Figure 50 show as red, green, and blue color channel images, respectively.

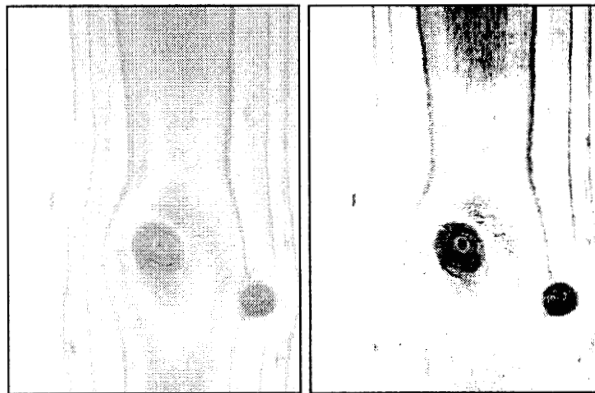


Figure 52. Black & white images of the color images presented in Figure 50 and Figure 51, respectively.



Figure 53. Image of core defects in Arizona White Pine wood shown as red, green, and blue color channel images, respectively.

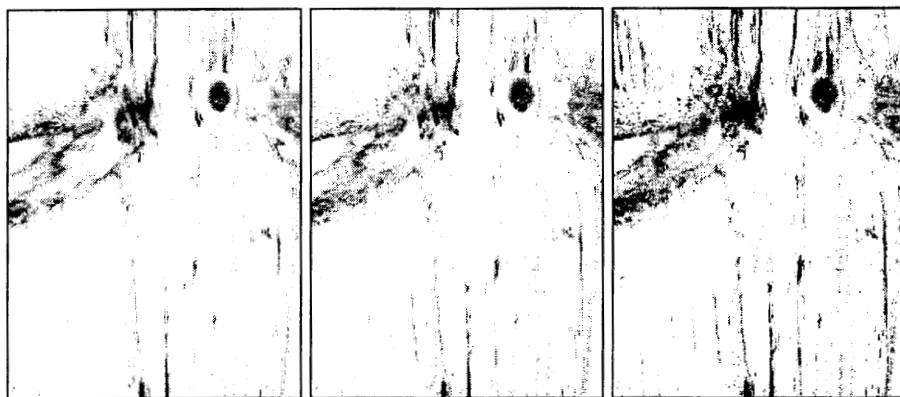


Figure 54. Color correlated image of Figure 53 show as red, green, and blue color channel images, respectively.

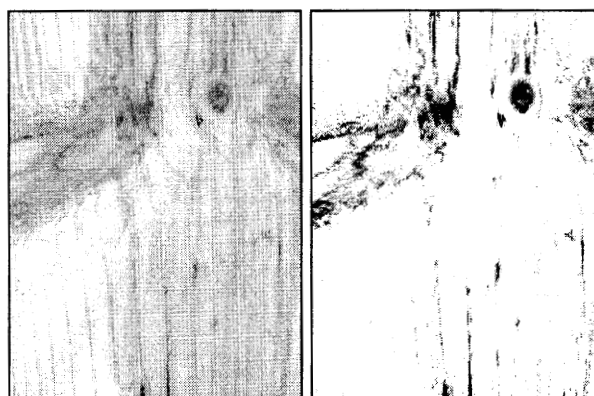


Figure 55. Black & white images of the color images presented in Figure 53 and Figure 54, respectively.

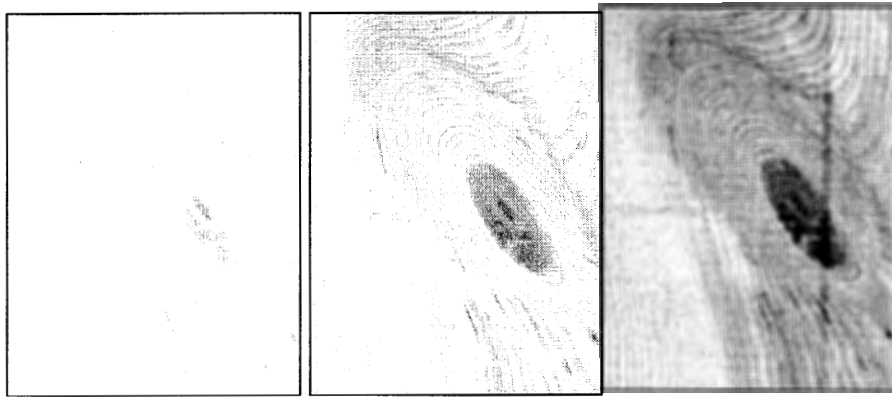


Figure 56. Image of spiked ring knots in Oregon Ponderosa Pine wood shown as red, green, and blue color channel images, respectively.

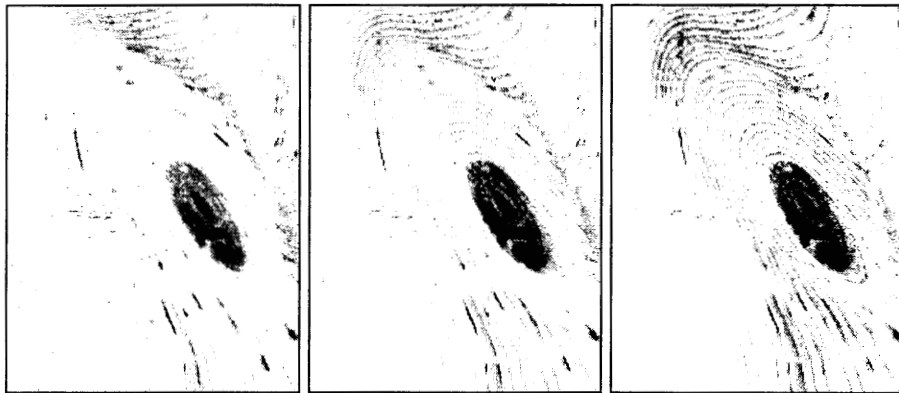


Figure 57. Color correlated image of Figure 56 show as red, green, and blue color channel images, respectively.

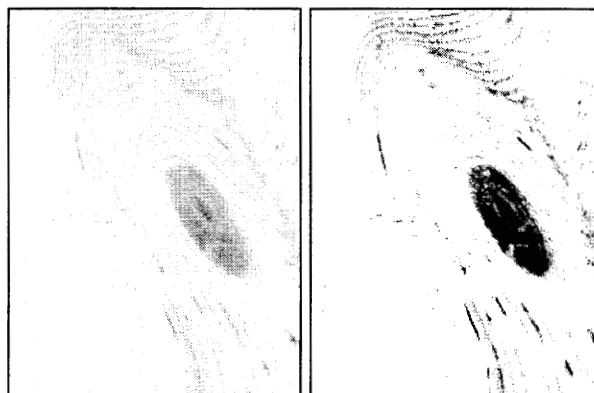


Figure 58. Black & white images of the color images presented in Figure 56 and Figure 57, respectively.



Figure 59. Image of core defects in Oregon Ponderosa Pine wood shown as red, green, and blue color channel images, respectively.

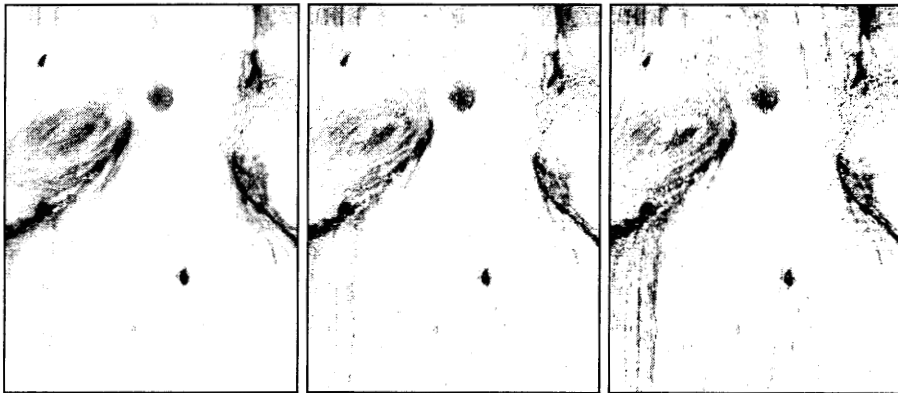


Figure 60. Color correlated image of Figure 59 show as red, green, and blue color channel images, respectively.



Figure 61. Black & white images of the color images presented in Figure 59 and Figure 60, respectively.

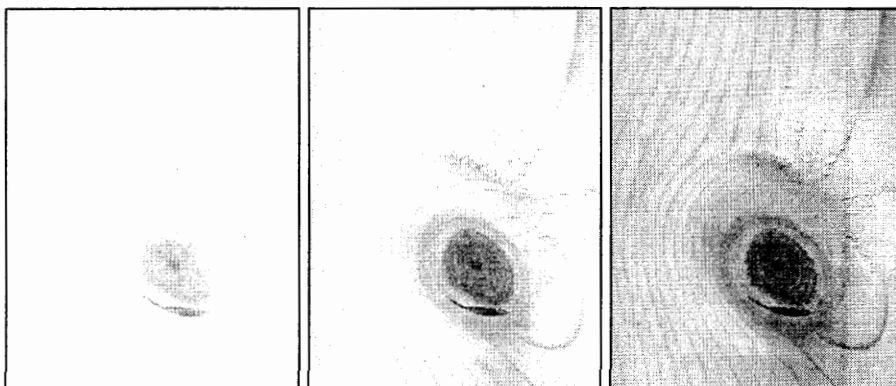


Figure 62. Image of not solid ring knot in Oregon Ponderosa Pine wood shown as red, green, and blue color channel images, respectively.

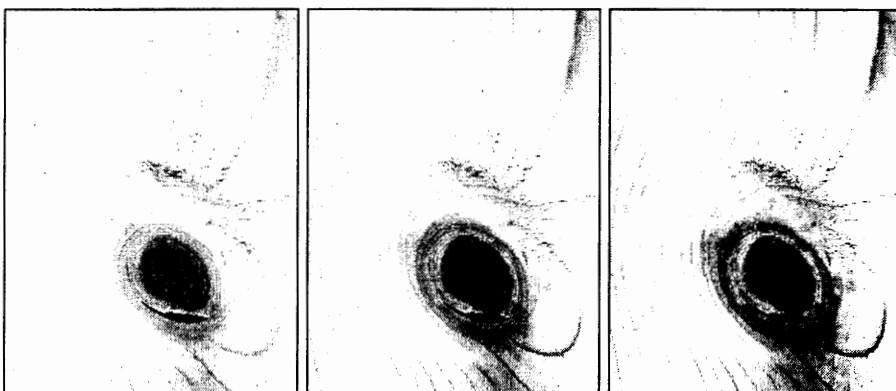


Figure 63. Color correlated image of Figure 62 show as red, green, and blue color channel images, respectively.



Figure 64. Black & white images of the color images presented in Figure 62 and Figure 63, respectively.

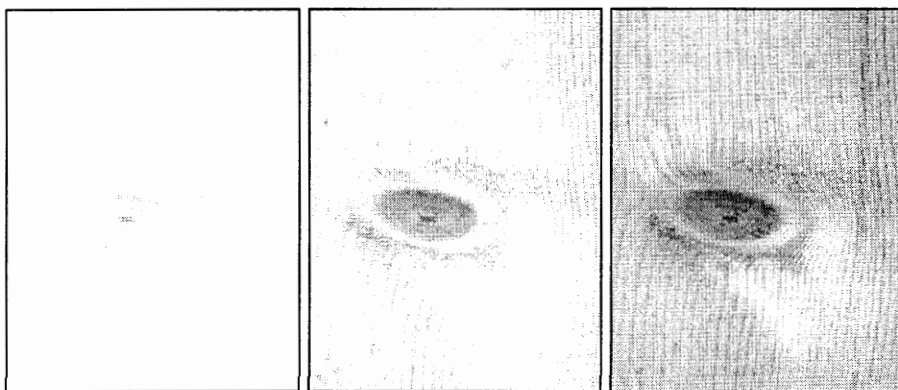


Figure 65. Image of knot with high-angle grain in Oregon Ponderosa Pine wood shown as red, green, and blue color channel images, respectively.

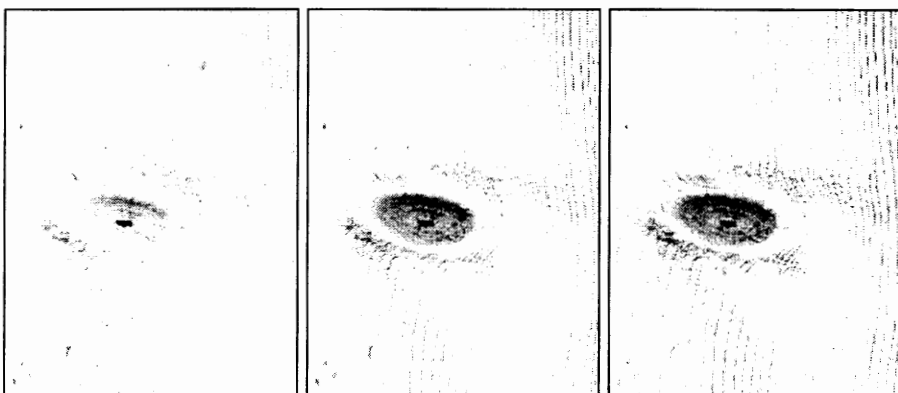


Figure 66. Color correlated image of Figure 65 show as red, green, and blue color channel images, respectively.

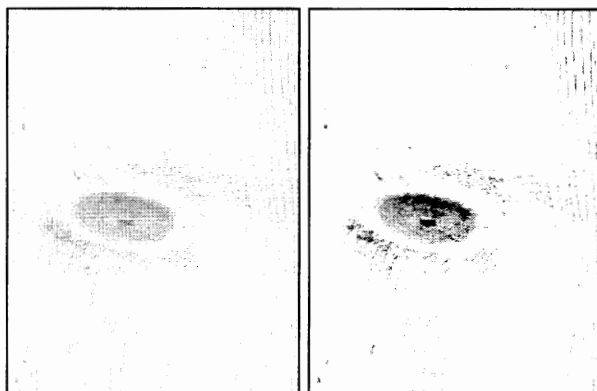


Figure 67. Black & white images of the color images presented in Figure 65 and Figure 66, respectively.

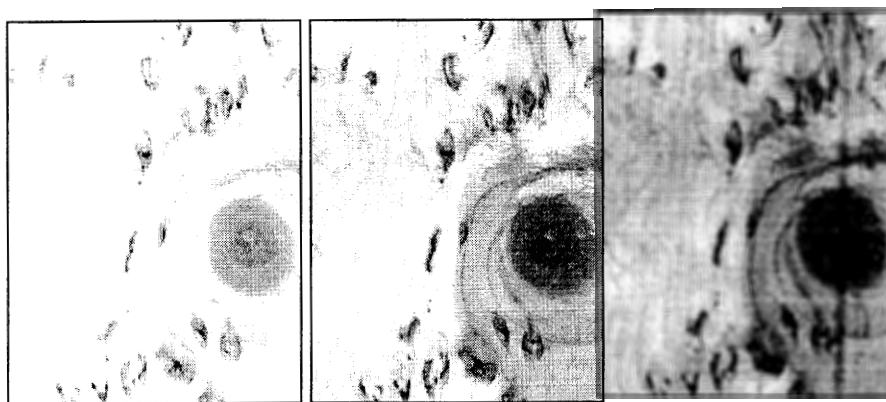


Figure 68. Image of bird's-eye knots with knot in Oregon Ponderosa Pine wood shown as red, green, and blue color channel images, respectively.

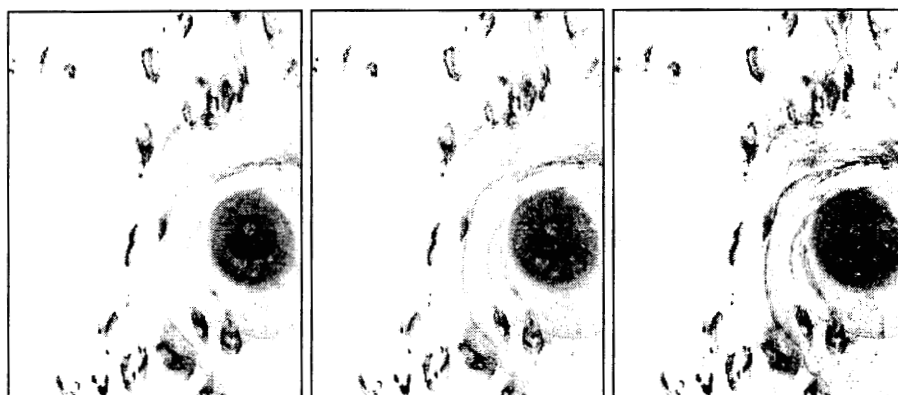


Figure 69. Color correlated image of Figure 68 show as red, green, and blue color channel images, respectively.

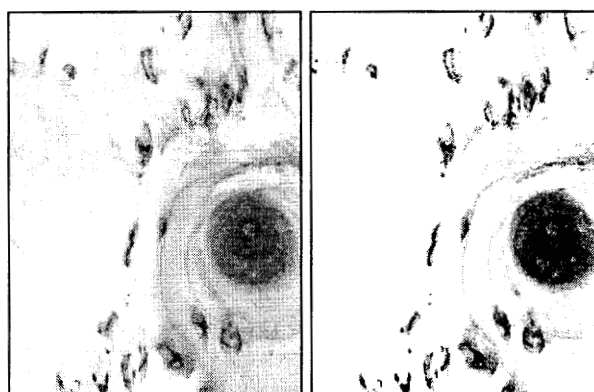


Figure 70. Black & white images of the color images presented in Figure 68 and Figure 69, respectively.

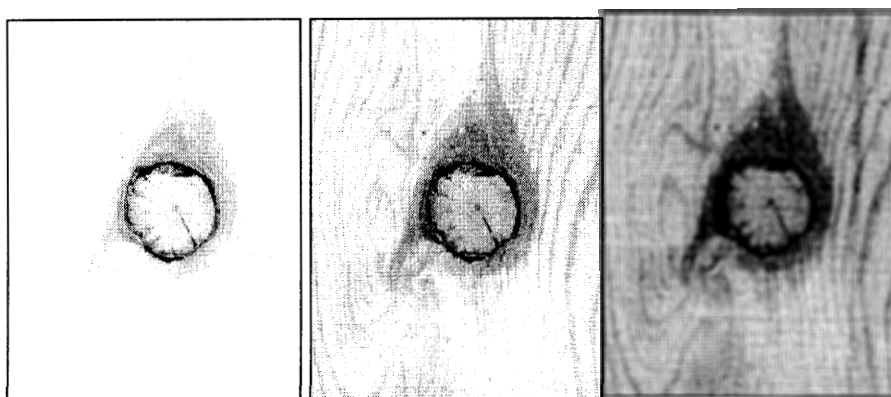


Figure 71. Image of not solid bark-edge knot in Oregon Ponderosa Pine wood shown as red, green, and blue color channel images, respectively.

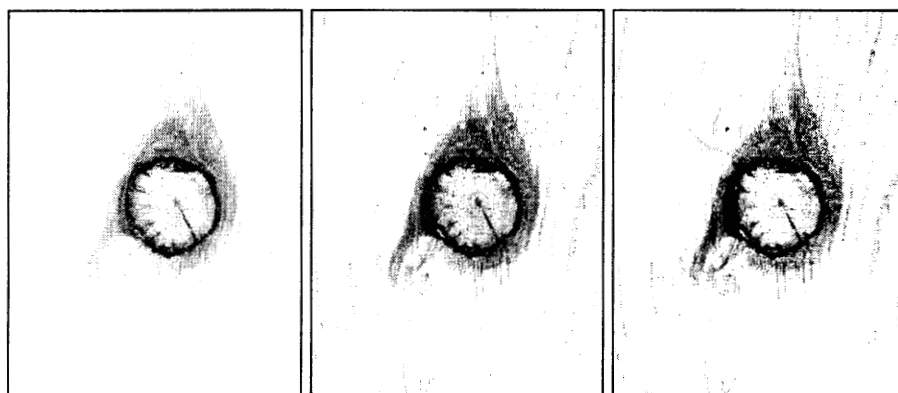


Figure 72. Color correlated image of Figure 71 show as red, green, and blue color channel images, respectively.

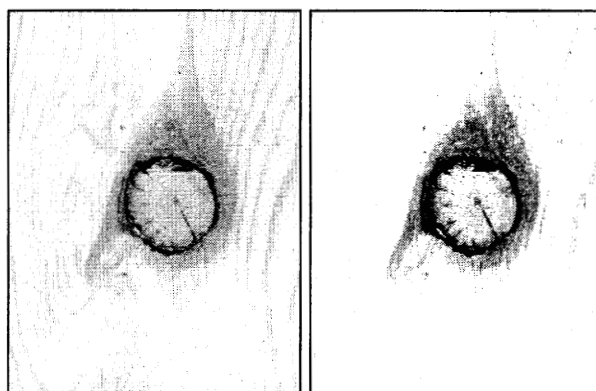


Figure 73. Black & white images of the color images presented in Figure 71 and Figure 72, respectively.

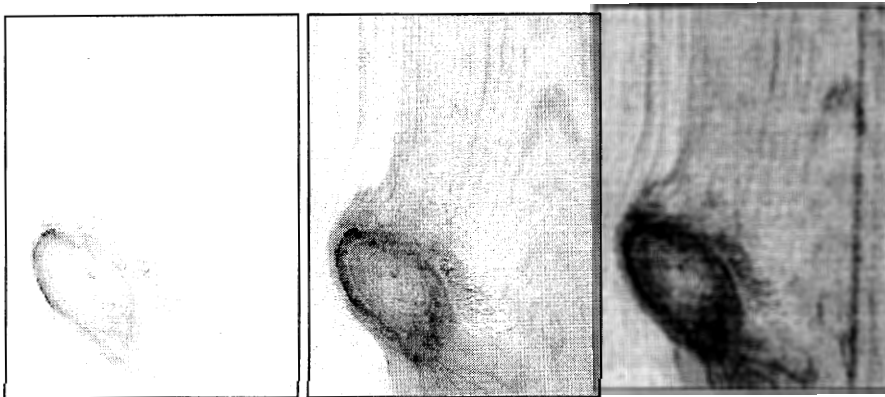


Figure 74. Image of knot in heartwood and sapwood in Ponderosa Pine wood shown as red, green, and blue color channel images, respectively.

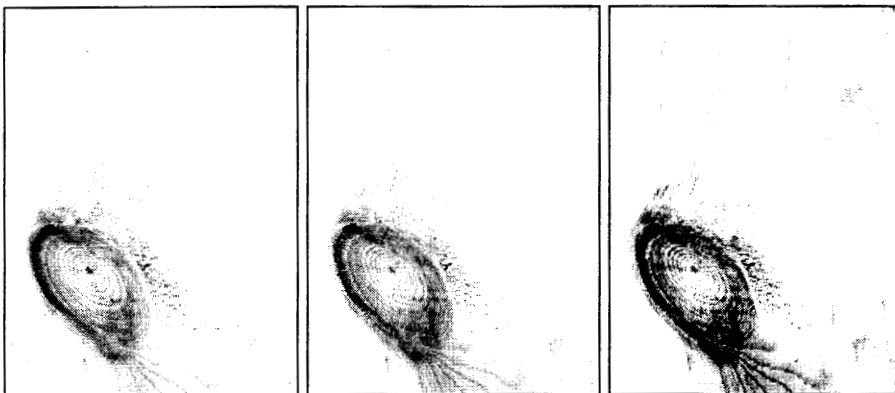


Figure 75. Color correlated image of Figure 74 show as red, green, and blue color channel images, respectively.

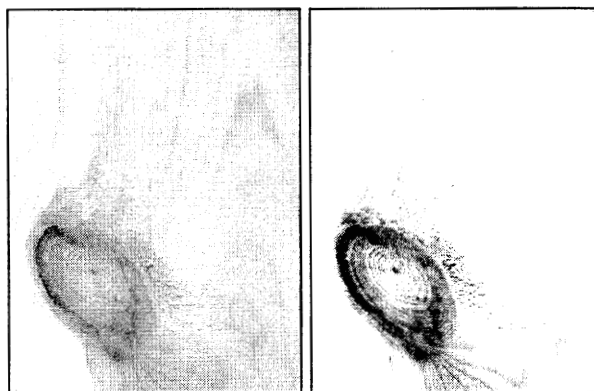


Figure 76. Black & white images of the color images presented in Figure 74 and Figure 75, respectively.



Figure 77. Image of core defects in rapid-growth Pine wood shown as red, green, and blue color channel images, respectively.

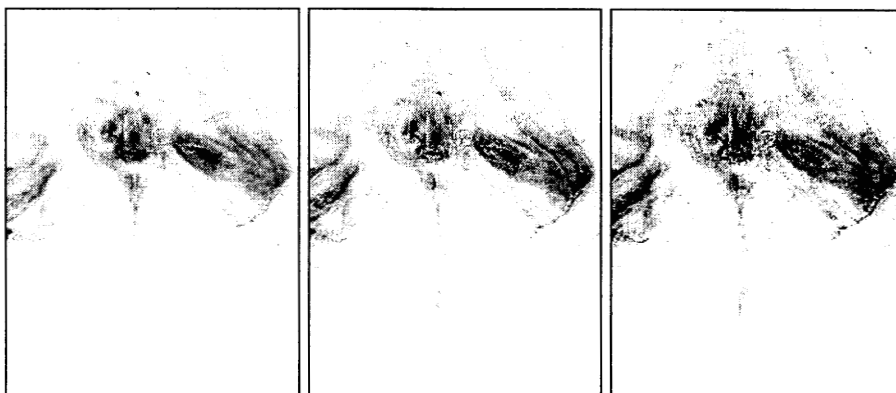


Figure 78. Color correlated image of Figure 77 show as red, green, and blue color channel images, respectively.

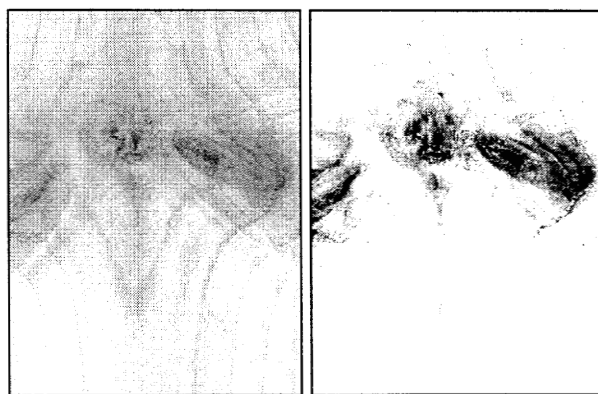


Figure 79. Black & white images of the color images presented in Figure 77 and Figure 78, respectively.

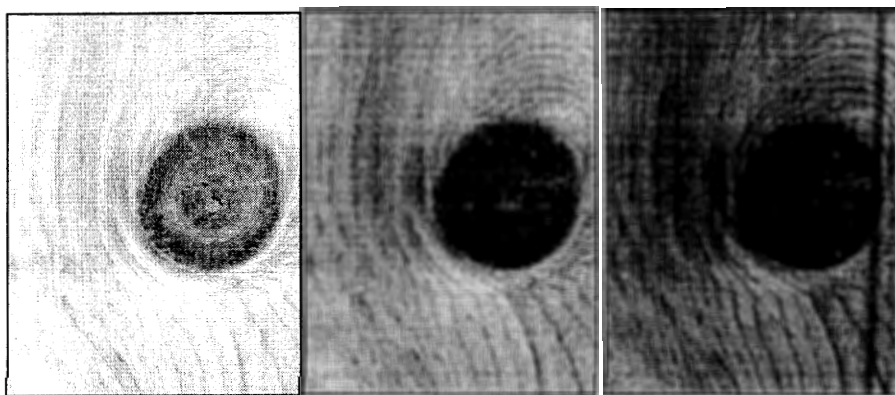


Figure 80. Image of solid red knot in weathered and dirty Pine wood shown as red, green, and blue color channel images, respectively.

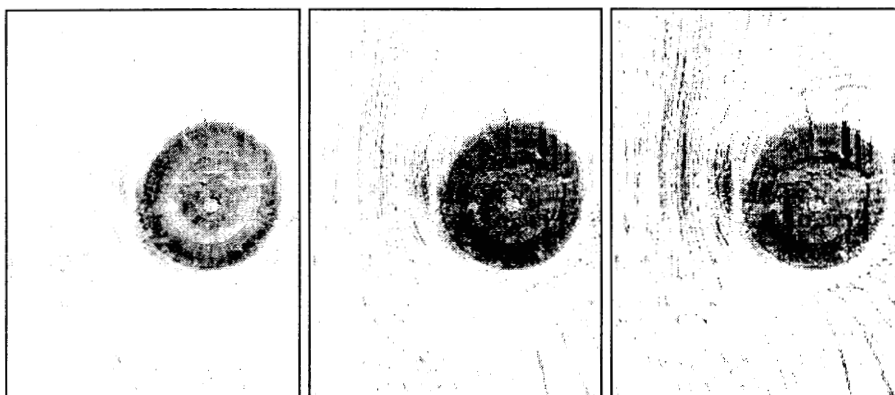


Figure 81. Color correlated image of Figure 80 show as red, green, and blue color channel images, respectively.

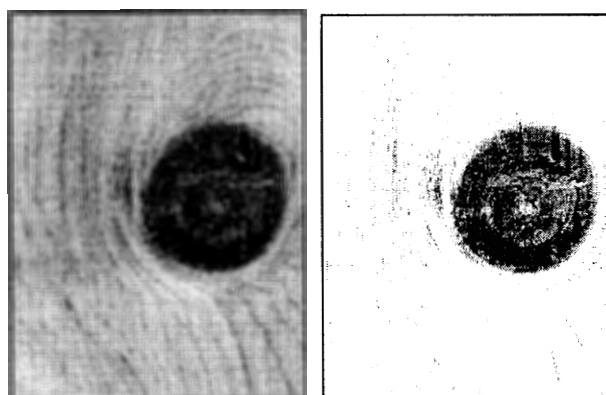


Figure 82. Black & white images of the color images presented in Figure 80 and Figure 81, respectively.

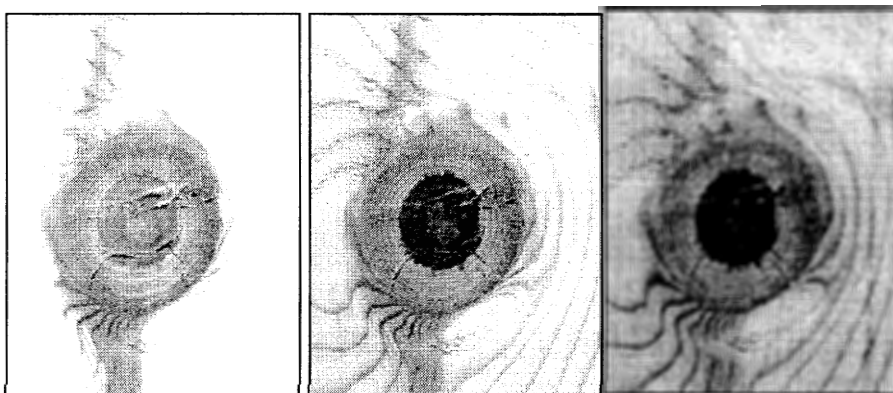


Figure 83. Image of red ring knot with blue stain in Pine wood shown as red, green, and blue color channel images, respectively.

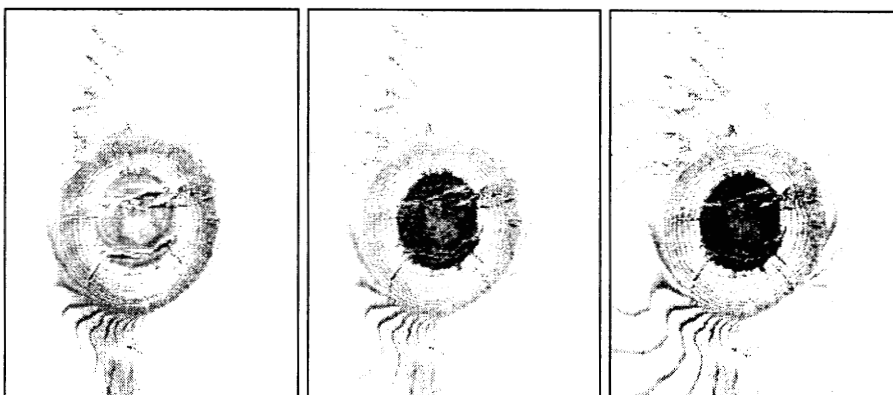


Figure 84. Color correlated image of Figure 83 show as red, green, and blue color channel images, respectively.

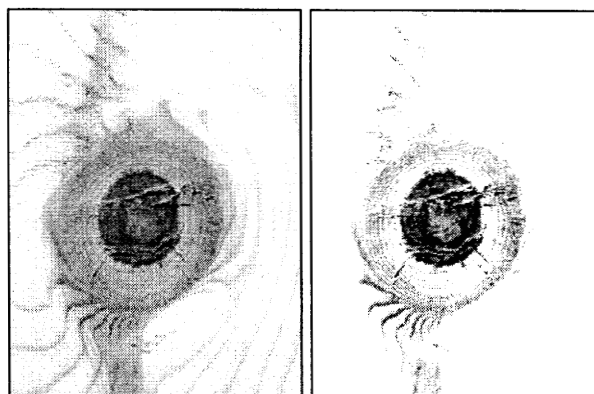


Figure 85. Black & white images of the color images presented in Figure 83 and Figure 84, respectively.

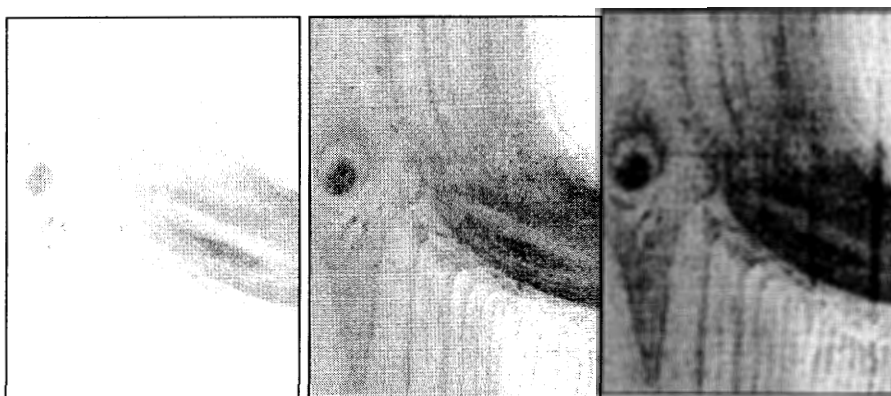


Figure 86. Image of high-angle spike knot in Pine wood shown as red, green, and blue color channel images, respectively.

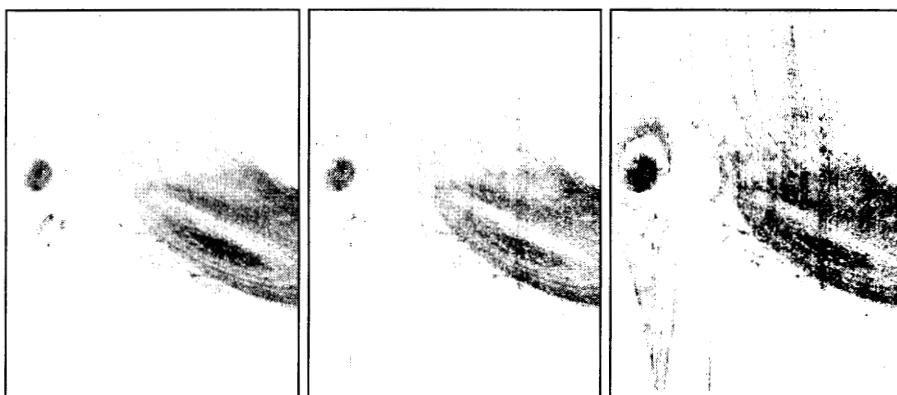


Figure 87. Color correlated image of Figure 86 show as red, green, and blue color channel images, respectively.

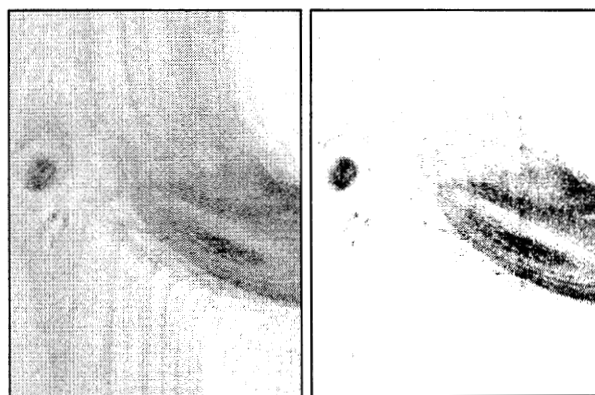


Figure 88. Black & white images of the color images presented in Figure 86 and Figure 87, respectively.

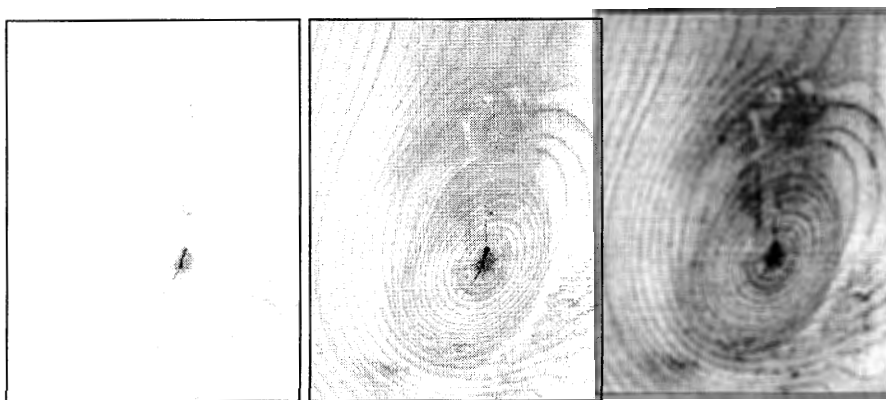


Figure 89. Image of knot with growth rings in Pine wood shown as red, green, and blue color channel images, respectively.



Figure 90. Color correlated image of Figure 89 show as red, green, and blue color channel images, respectively.

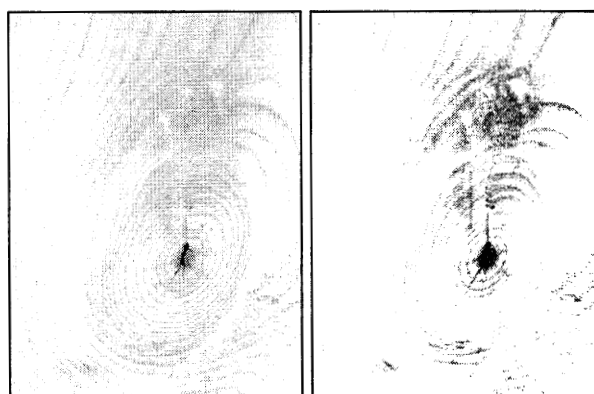


Figure 91. Black & white images of the color images presented in Figure 89 and Figure 90, respectively.

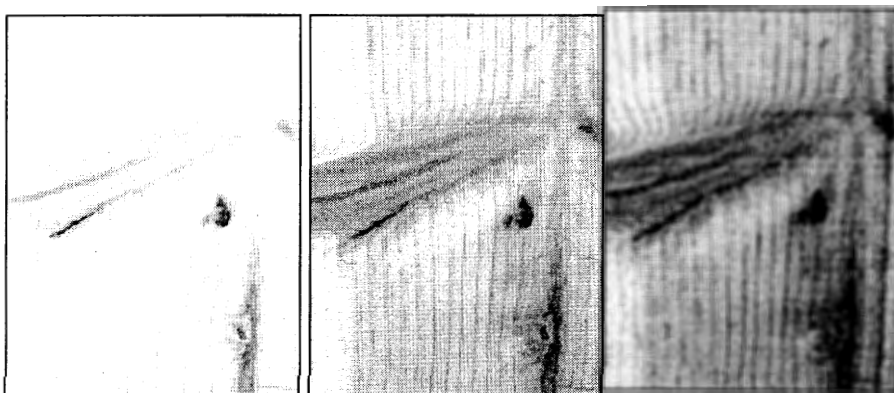


Figure 92. Image of long spike knot, pitch pocket, and pith in Pine wood shown as red, green, and blue color channel images, respectively.



Figure 93. Color correlated image of Figure 92 show as red, green, and blue color channel images, respectively.

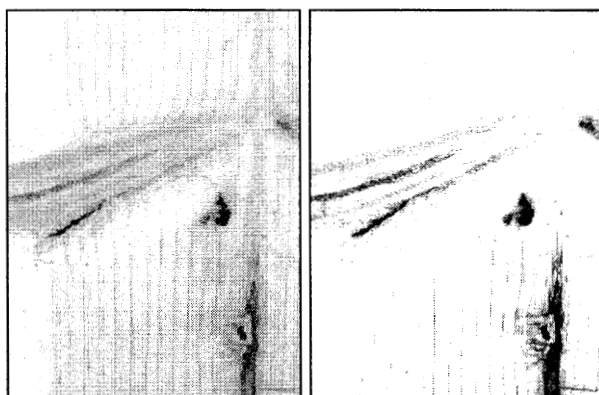


Figure 94. Black & white images of the color images presented in Figure 92 and Figure 93, respectively.

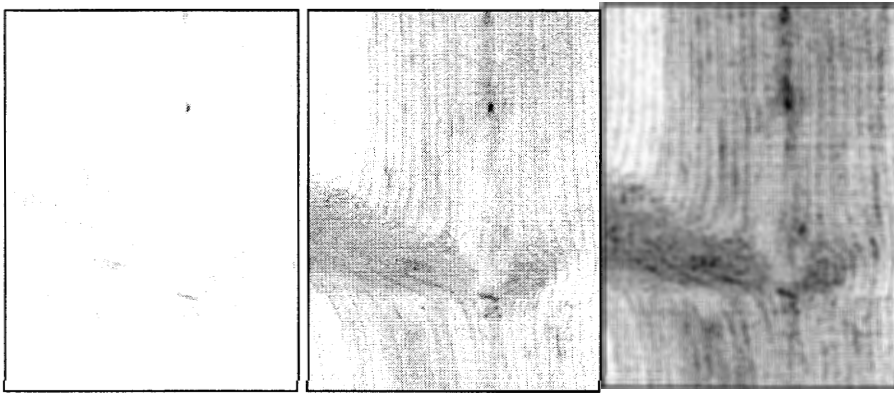


Figure 95. Image of low-angle spike knot in Pine wood shown as red, green, and blue color channel images, respectively.

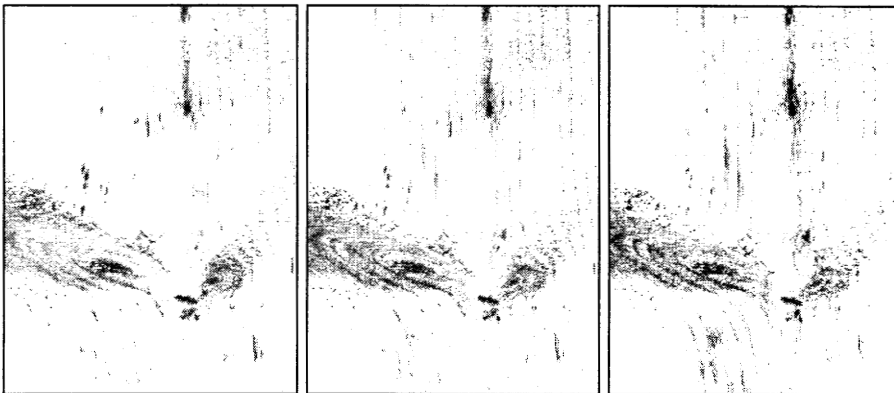


Figure 96. Color correlated image of Figure 95 show as red, green, and blue color channel images, respectively.

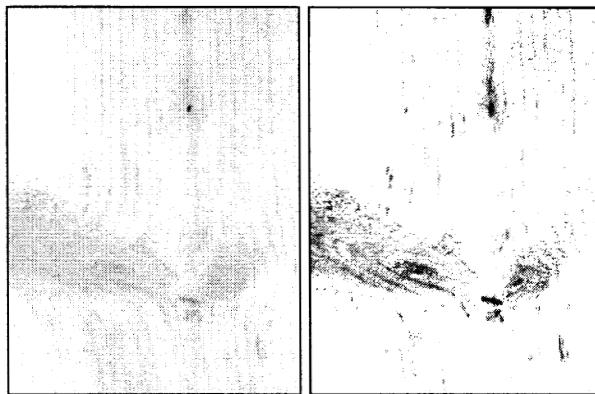


Figure 97. Black & white images of the color images presented in Figure 95 and Figure 96, respectively.



Figure 98. Image of ring knot with sap stain in Pine wood shown as red, green, and blue color channel images, respectively.

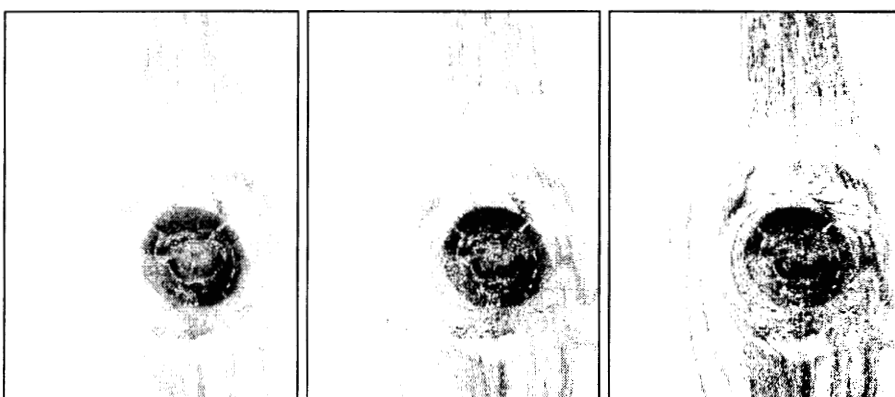


Figure 99. Color correlated image of Figure 98 show as red, green, and blue color channel images, respectively.

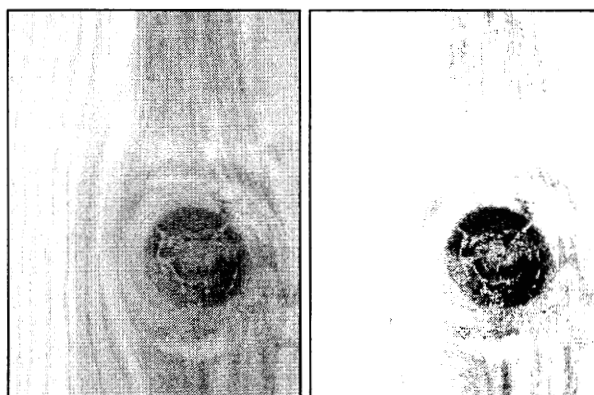


Figure 100. Black & white images of the color images presented in Figure 98 and Figure 99, respectively.



Figure 101. Image of knot with sap stain in Pine wood shown as red, green, and blue color channel images, respectively.

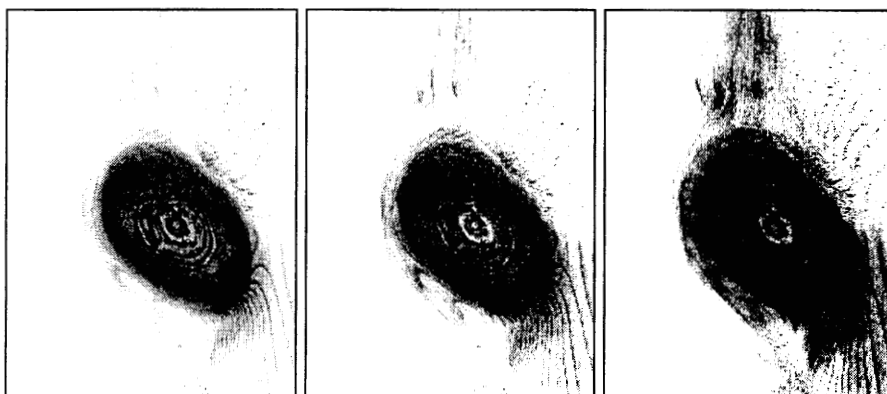


Figure 102. Color correlated image of Figure 101 show as red, green, and blue color channel images, respectively.

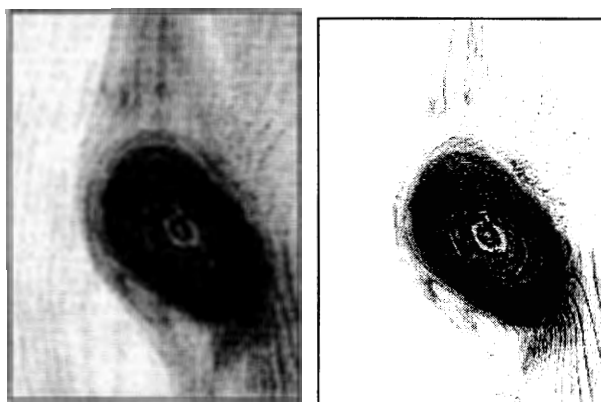


Figure 103. Black & white images of the color images presented in Figure 101 and Figure 102, respectively.

APPENDIX C

BOARD VALUE OPTIMIZER COMPUTER C CODE

This appendix presents the computer software programming functions used to perform value optimization on the board model. All programming code is written in the C language and is designed to be compiled for and implemented on a personal computer platform. The first programming function, called “find_best_bd,” selects the highest value rip solution. This function selects the highest market value rip solution if the rip-saw rip-width production rate is less than the crosscut saw consumption rate.

Weighted value selection uses four weight classes: 1) the finger-joint class weights the value of cut products less than 13 inches long, 2) the shorts class weights the value of 13 inches to 72 inches long cut products, and 3) the longs class weights the value of cut products greater than 72 inches long. The shorts weight is applied using a unique trapezoidal algorithm. After selection, the short and long-term rip-saw rip-width production rates are adjusted and the three weight classes are adjusted for the new production rates.

The second function, called “cutaframerip,” is used to place cut product onto rip-widths. The function uses defect data from the board model, and allows both clear and single-sided-defect products, called frame grade. The function saves the rip-width results into an array which is then post-processed for all rips filling-out the board width.

```

#define KERF          0.125F
#define WIDE_RIP      2.5F
#define EDGE_CUT      0.375F
#define WIDTH_TO_BDFT 0.00868F
#define INCH_TO_FOOT  0.08333F
#define SEC_PER_MIN    60.0F
#define 3_MIN_RATE    0.017F //5% per 3 minute
#define 5_MIN_RATE    0.19F  //95% per 5 minutes

/*****

////TITLE:      FIND BEST BOARD RIP SOLUTION
////FUNCTION:    find_best_bd
////SCOPE:      SUBSYSTEM
////PURPOSE:
//  Select the highest value solution given production constraints.
//
////INPUTS:
//  Called from ripitup.
//  Ripsaw production rate statistics:
//      typedef struct {
//          float lnft;
//          float st_lnft;
//          float lt_lnft;
//          float production;
//          float lnft_minute;
//          float lt_status;
//          float consumption;
//      }prodage;
//      extern prodage production[MAXRIPS];
//
//  Chopsaw consumption rate statistics:
//      typedef struct {
//          float rip_width;
//          float lnft_minute;
//          float status;
//          float st_status;
//          float lt_status;
//      }consage;
//      extern consage consumption[MAXCUTTABLE];
//
//  Alternate rip solution data:
//      typedef struct {
//          float base;
//          float atmarket;
//          float lealedge;
//          float mktP_sum[MAXRIPS];
//          float pmix[MAXRIPS];
//          float ripn[MAXRIPS];
//          float mktP_core[MAXRIPS];
//          float lnin_core[MAXRIPS];
//          float mktP_fj[MAXRIPS];
//          float lnin_fj[MAXRIPS];
//          float lnin_shorts[MAXRIPS];
//          float mktP_long[s[MAXRIPS];
//          float lnin_long[s[MAXRIPS];
//          float mktP_shorts[MAXSHORTS][MAXRIPS];
//      }bds;

```

```

//      extern bds board[386];
//
//////ASSUMPTIONS:
//      At least one solution found.
//
//////OUTPUTS:
//      Return Value:  TYPE  DESC
//                      ----  ----
//                      int   Return Code
//
//      Rip solution:
//      typedef struct {
//          int      bdnum;
//          float    edging;
//          float    waste;
//          float    rip_width[MAXRIPS];
//          float    bd_length;
//          float    bd_width;
//          float    bdft_core[MAXRIPS];
//          float    mktP_core[MAXRIPS];
//          float    bdft_fj[MAXRIPS];
//          float    mktP_fj[MAXRIPS];
//          float    bdft_shorts[MAXRIPS];
//          float    bdft_longs[MAXRIPS];
//          float    mktP_longs[MAXRIPS];
//          float    mktP_shorts[MAXRIPS];
//          float    footage[MAXRIPS];
//          float    value;
//          }ripsols;
//      extern ripsols rip_solution, atmarket_solution;
//
//////AUTHOR:
//      John Goulding, 1996.
//

*****/

int find_best_bd (void)
{
    int      i, j, k, l;
    int      gotone, best, max_mkt;
    float    temp, temp2, tmp_sum, tmp_sum2;
    float    minimum_value, weighted_value, bd_lnft;
    float    del_rip[MAXRIPS];
    float    rip_lnft[MAXRIPS];
    char     txt_tmp[31];

    /*****/
    /* SAVE THE MAXIMUM VALUE RIP SOLUTION */
    /*****/

    /* Set the rip solution to number 0 */
    /* The rip solutions are sorted by decending value */
    /* So, number 0 is the maximum value solution */
    best = 0;

```



```

/* Sum the solution width to determine waste */
tmp_sum = board[bd_index[best]].leadedge;
atmarket_solution.edging = tmp_sum;

/* Copy the rip pattern and sum the solution width */
for(i=0; i<6 || board[bd_index[best]].ripn[i]==0.0F; ++i)
{
    atmarket_solution.rip_width[i] = board[bd_index[best]].ripn[i];
    tmp_sum += board[bd_index[best]].ripn[i] + KERF;
}
for(;i<6; ++i)
{
    atmarket_solution.rip_width[i] = 0.0F;
}

/* Determine the outside waste */
tmp_sum -= KERF;
atmarket_solution.waste = maximum_edge - tmp_sum;
if(atmarket_solution.waste<0.0F)
{
    atmarket_solution.waste = 0.0001F;
}

/* Save other solution parameters */
atmarket_solution.bd_length = bd_length;
rip_solution.bd_length = atmarket_solution.bd_length;
atmarket_solution.bd_width = maximum_edge;
rip_solution.bd_width = atmarket_solution.bd_width;

/* Save the solution footages by rip-width and length class*/
/* Save the solution dollar value by rip-width and length class */
atmarket_solution.value = 0.0F;
for(j=0; j<MAXRIPS; ++j)
{
    /* Determine the board footage for the rip-width */
    temp = ripvalue[j].rip_width * WIDTH_TO_BDFT;

    /* Determine the board footage of core products */
    atmarket_solution.bdft_core[j] = board[bd_index[best]].lnin_core[j]
        * temp;

    /* Copy the value of core for the rip-width */
    if(board[bd_index[best]].mktP_core[j]>0.01F)
    {
        atmarket_solution.mktP_core[j] =
            board[bd_index[best]].mktP_core[j];
    }
    else
    {
        atmarket_solution.mktP_core[j] = 0.01F;
    }

    /* Sum the total value */
    atmarket_solution.value += atmarket_solution.mktP_core[j];

    /* Determine the board footage of finger-joint products */
    atmarket_solution.bdft_fj[j] = board[bd_index[best]].lnin_fj[j]
        * temp;
}

```

```

/* Copy the value of finger-joint for the rip-width */
if(board[bd_index[best]].mktP_fj[j]>0.01F)
{
    atmarket_solution.mktP_fj[j] = board[bd_index[best]].mktP_fj[j];
}
else
{
    atmarket_solution.mktP_fj[j] = 0.01F;
}

/* Sum the total value */
atmarket_solution.value += atmarket_solution.mktP_fj[j];

/* Determine the board footage of longs products */
atmarket_solution.bdft_long[j] =
    board[bd_index[best]].lnin_long[j] * temp;

/* Copy the value of longs for the rip-width */
if(board[bd_index[best]].mktP_long[j]>0.01F)
{
    atmarket_solution.mktP_long[j] =
        board[bd_index[best]].mktP_long[j];
}
else
{
    atmarket_solution.mktP_long[j] = 0.01F;
}

/* Sum the total value */
atmarket_solution.value += atmarket_solution.mktP_long[j];

/* Determine the board footage of shorts products */
atmarket_solution.bdft_short[j] =
    board[bd_index[best]].lnin_short[j] * temp;

/* Copy the value of shorts for the rip-width */
atmarket_solution.mktP_short[j] = 0.01F;
for(k=0; k<MAXSHORTS; ++k)
{
    /* Copy the value of the shorts sub-classes */
    if(board[bd_index[best]].mktP_short[k][j]>0.01F)
    {
        atmarket_solution.mktP_short[j] +=
            board[bd_index[best]].mktP_short[k][j];
    }
}

/* Sum the total value */
atmarket_solution.value += atmarket_solution.mktP_short[j];
} //for j<MAXRIPS loop

/*****/
/* CHECK THE MAXIMUM VALUE RIP SOLUTION */
/*      IS PRODUCTION <= CONSUMPTION      */
/*****/

```

```

/* Calculate the lnft for each rip-width */
bd_lnft = bd_length * INCH_TO_FOOT;
for(i=0; i<MAXRIPS && ripvalue[i].rip_width!=0.0F; ++i)
{
    rip_lnft[i] = 0.0F;

    /* Search the solution for the rip-width */
    for(j=0; j<6 && atmarket_solution.rip_width[j]!=0.0F; ++j)
    {
        /* If present in solution, add the lnft */
        if(ripvalue[i].rip_width==atmarket_solution.rip_width[j])
        {
            rip_lnft[i] += bd_lnft;
        }
    }
}

/* Calculate lnft per minute for each rip */
/* Null hypothesis: production less than consumption */
gotone = 1;
for(i=0; i<MAXRIPS && ripvalue[i].rip_width!=0.0F; ++i)
{
    /* Zero time since last production */
    del_rip[i] = 0.0F;
    if(rip_lnft[i]==0.0F)
    {
        continue;
    }

    /* Calculate production time difference */
    del_rip[i] = (time_bd_out - time_rip[i]) / CLOCKS_PER_SEC;

    /* Interval per minute */
    if(del_rip[i]>SEC_PER_MIN)
    {
        del_rip[i] = SEC_PER_MIN;
    }
    else if(del_rip[i]<=0.0F)
    {
        del_rip[i] = SEC_PER_MIN;
    }

    /* Convert to production rate per board */
    del_rip[i] = SEC_PER_MIN / del_rip[i];

    /* Add short and long-term rate contribution */
    /* Contribute 5% per 3 minute interval */
    /* Contribute 95% per 5 minute interval */
    tmp_sum = rip_lnft[i] * del_rip[i] * 3_MIN_RATE;
    tmp_sum += rip_lnft[i] * del_rip[i] * 5_MIN_RATE;

    /* Check if board production is greater than consumption */
    /* Check if historical production is greater than consumption */
    if(tmp_sum>production[i].consumption
        || production[i].production>production[i].consumption)
    {
        gotone = 0;
    }
}

```

```

        break;
    }
}

/*****
/* FIND THE BEST WEIGHTED-VALUE SOLUTION */
*****/

/* Find the weighted value solution for the best board combos */
if(!gotone)
{
    /* Begin with a null solution */
    minimum_value = 0.0F;

    /* Process all saved rip solutions */
    /* Save a pointer to the highest weighted value solution */
    for(i=0; i<numbd_saved && i<385; ++i)
    {
        gotone = 0;

        /* calculate lnft for each rip */
        for(k=0; k<MAXRIPS && ripvalue[k].rip_width!=0.0F; ++k)
        {
            rip_lnft[k] = 0.0F;

            /* Search the solution for the rip-width */
            for(j=0; j<6 && board[bd_index[i]].ripn[j]!=0.0F; ++j)
            {
                /* If present in solution, add the lnft */
                if(ripvalue[k].rip_width==board[bd_index[i]].ripn[j])
                {
                    rip_lnft[k] += bd_lnft;
                }
            }
        }

        /* determine production-capacity weighting relationship */
        for(i=0; i<MAXRIPS && ripvalue[i].rip_width!=0.0F; ++i)
        {
            /* Zero time since last production */
            del_rip[i] = 0.0F;
            if(rip_lnft[i]==0.0F)
            {
                continue;
            }

            /* Calculate production time difference */
            del_rip[i] = (time_bd_out - time_rip[i]) / CLOCKS_PER_SEC;

            /* Interval per minute */
            if(del_rip[i]>SEC_PER_MIN)
            {
                del_rip[i] = SEC_PER_MIN;
            }
            else if(del_rip[i]<=0.0F)
            {
                del_rip[i] = SEC_PER_MIN;
            }
        }
    }
}

```

```

}

/* Convert to production rate per board */
del_rip[i] = SEC_PER_MIN / del_rip[i];

/* Add short and long-term rate contribution */
/* Contribute 5% per 3 minute interval */
/* Contribute 95% per 5 minute interval */
tmp_sum = rip_lnft[i] * del_rip[i] * 3 MIN_RATE;
tmp_sum += rip_lnft[i] * del_rip[i] * 5 MIN_RATE;

/* Check if production < consumption */
/* Check if historical production > consumption */
/* Check if board production > consumption */
if((production[j].production>production[j].consumption
|| tmp_sum>production[j].consumption)
&& (board[bd_index[i]].pmix[j]>=board[bd_index[0]].pmix[j]
|| ripvalue[j].rip_width<WIDE_RIP))
{
    /* NO, so apply weights directly and take maximum loss */
    gotone++;
    break;
}
}

/* Initialize a base value for negative weights */
/* Penalties are used to factor-in fixed production costs */
weighted_value = board[bd_index[i]].base;

/* YES, apply the trapezoidal weights and take less loss */
if(gotone==0)
{
    /* Calculate the weighted value for each rip width */
    for(j=0; j<MAXRIPS && ripvalue[j].rip_width!=0.0F; ++j)
    {
        /* Add core values directly */
        weighted_value += board[bd_index[i]].mktP_core[j];

        /* Check for process penalties */
        if(board[bd_index[i]].mktP_fj[j]>0.0F)
        {
            /* Multiply finger-joint by finger-joint weight */
            weighted_value += board[bd_index[i]].mktP_fj[j]
                * ripvalue[j].weight_fj;
        }
        else
        {
            /* Preserve negative penalty value */
            weighted_value += board[bd_index[i]].mktP_fj[j]
                * ((1.0F - ripvalue[j].weight_fj) + 1.0F);
        }
    }

    /* Multiply longs by longs weight */
    weighted_value += board[bd_index[i]].mktP_long[s][j]
        * ripvalue[j].weight_long[s];

    /* Apply trapezoidal weight to the shorts by sub-classes */
    if(ripvalue[j].weight_shorts>0.05F)

```

```

{
    /* Determine the relative weight index */
    k = (int)((1.0F - ripvalue[j].weight_shorts)
        * (float)(ripvalue[j].numshorts + 2));

    /* Use 4 multipliers for trapezoidal shape */
    if(k==1)
    {
        /* Trapezoid is one product deep */
        weighted_value += board[bd_index[i]].mktP_shorts[0][j]
            * 0.7F;
    }
    else if(k>1 && (k-1)<MAXSHORTS)
    {
        /* Trapezoid is several products deep */
        weighted_value +=
            board[bd_index[i]].mktP_shorts[(k-1)][j] * 0.7F;
        weighted_value +=
            board[bd_index[i]].mktP_shorts[(k-2)][j] * 0.3F;
    }
    else if(k>1 && (k-2)<MAXSHORTS)
    {
        /* Only tip of trapezoid multiplies */
        weighted_value +=
            board[bd_index[i]].mktP_shorts[(k-2)][j] * 0.3F;
    }

    /* Sum the total weighted value by product sub-class */
    for(; k<MAXSHORTS && k<ripvalue[j].numshorts; ++k)
    {
        weighted_value += board[bd_index[i]].mktP_shorts[k][j];
    }
}
}

/* NO, so apply the weights directly and take maximum loss */
else
{
    /* Calculate the weighted value for each rip width */
    for(j=0; j<MAXRIPS && ripvalue[j].rip_width!=0.0F; ++j)
    {
        /* Add the value of core directly */
        weighted_value += board[bd_index[i]].mktP_core[j];

        /* Check for process penalties */
        if(board[bd_index[i]].mktP_fj[j]>0.0F)
        {
            /* Multiply finger-joint by finger-joint weight */
            weighted_value += board[bd_index[i]].mktP_fj[j]
                * ripvalue[j].weight_fj;
        }
        else
        {
            /* Preserve negative penalty value */
            weighted_value += board[bd_index[i]].mktP_fj[j]
                * ((1.0F - ripvalue[j].weight_fj) + 1.0F);
        }
    }
}

```

```

/* Multiply longs by longs weight */
weighted_value += board[bd_index[i]].mktP_long[j]
    * ripvalue[j].weight_long;

/* apply trapezoidal weight to the shorts by sub-classes */
if(ripvalue[j].weight_shorts>0.05F)
{
    /* Determine the relative weight index */
    k = (int)((1.0F - ripvalue[j].weight_shorts)
        * (float)(ripvalue[j].numshorts + 2));

    /* Use 4 multipliers for trapezoidal shape */
    if(k==1)
    {
        /* Trapezoid is one product deep */
        weighted_value +=
            board[bd_index[i]].mktP_shorts[0][j] * 0.7F;
    }
    else if(k>1 && (k-1)<MAXSHORTS)
    {
        /* Trapezoid is several products deep */
        weighted_value +=
            board[bd_index[i]].mktP_shorts[(k-1)][j] * 0.7F;
        weighted_value +=
            board[bd_index[i]].mktP_shorts[(k-2)][j] * 0.3F;
    }
    else if(k>1 && (k-2)<MAXSHORTS)
    {
        /* Only tip of trapezoid multiplies */
        weighted_value +=
            board[bd_index[i]].mktP_shorts[(k-2)][j] * 0.3F;
    }

    /* Sum the total weighted value by product sub-class */
    for(; k<MAXSHORTS && k<ripvalue[j].numshorts; ++k)
    {
        weighted_value += board[bd_index[i]].mktP_shorts[k][j];
    }
}

}

/* Check for a new highest value solution */
if(weighted_value>minimum_value)
{
    /* Valid weighted solution */
    /* Reset the highest value */
    /* Save a pointer to the solution */
    minimum_value = weighted_value;
    best = i;
}
}

/*****
/* SAVE THE BEST RIP SOLUTION */
*****/

```

```

/* From GOTO branch */
save_rip_solution:

/* Sum the solution width to determine waste */
tmp_sum = board[bd_index[best]].leadedge;
rip_solution.edging = tmp_sum;

/* Copy the rip pattern and sum the solution width */
for(i=0; i<6 || board[bd_index[best]].ripn[i]==0.0F; ++i)
{
    rip_solution.rip_width[i] = board[bd_index[best]].ripn[i];
    tmp_sum += board[bd_index[best]].ripn[i] + KERF;
}
for(;i<6; ++i)
{
    rip_solution.rip_width[i] = 0.0F;
}

/* Determine the outside waste */
tmp_sum -= 0.KERF;
rip_solution.waste = maximum_edge - tmp_sum;
if(rip_solution.waste<0.0F)
{
    rip_solution.waste = 0.0001F;
}

/* Save other solution parameters */
gotone = 0;
rip_solution.value = 0.0F;

/* Save the solution footages by rip-width and length class*/
/* Save the solution dollar value by rip-width and length class */
for(j=0; j<MAXRIPS; ++j)
{
    /* Determine the board footage for the rip-width */
    temp = ripvalue[j].rip_width * WIDTH_TO_BDFT;

    /* Determine the board footage of core products */
    rip_solution.bdft_core[j] = board[bd_index[best]].lnin_core[j]
        * temp;

    /* Copy the value of core for the rip-width */
    if(board[bd_index[best]].mktP_core[j]>0.01F)
    {
        rip_solution.mktP_core[j] = board[bd_index[best]].mktP_core[j];
    }
    else
    {
        rip_solution.mktP_core[j] = 0.01F;
    }

    /* Sum the total value */
    rip_solution.value += rip_solution.mktP_core[j];

    /* Determine the board footage of finger-joint products */
    rip_solution.bdft_fj[j] = board[bd_index[best]].lnin_fj[j] * temp;
}

```



```

/* Copy the value of finger-joint for the rip-width */
if (board[bd_index[best]].mktP_fj[j]>0.01F)
{
    rip_solution.mktP_fj[j] = board[bd_index[best]].mktP_fj[j];
}
else
{
    rip_solution.mktP_fj[j] = 0.01F;
}

/* Sum the total value */
rip_solution.value += rip_solution.mktP_fj[j];

/* Determine the board footage of longs products */
rip_solution.bdft_long[j] = board[bd_index[best]].lnin_long[j]
    * temp;

/* Copy the value of longs for the rip-width */
if (board[bd_index[best]].mktP_long[j]>0.01F)
{
    rip_solution.mktP_long[j] =
        board[bd_index[best]].mktP_long[j];
}
else
{
    rip_solution.mktP_long[j] = 0.01F;
}

/* Sum the total value */
rip_solution.value += rip_solution.mktP_long[j];

/* Determine the board footage of shorts products */
rip_solution.bdft_short[j] = board[bd_index[best]].lnin_short[j]
    * temp;

/* Copy the value of shorts for the rip-width */
rip_solution.mktP_short[j] = 0.01F;
for(k=0; k<MAXSHORTS; ++k)
{
    /* Copy the value of the shorts sub-classes */
    if (board[bd_index[best]].mktP_short[k][j]>0.01F)
    {
        rip_solution.mktP_short[j] +=
            board[bd_index[best]].mktP_short[k][j];
    }
}

/* Sum the total value */
rip_solution.value += rip_solution.mktP_short[j];
}

/*****
/* DETERMINE THE OPTIMAL RIP PRODUCTION RATE */
/* SET THE FEEDER TIMING AT THE RIPS AW */
*****/

/* Set the best processing hold time for feeder 6 */

```

```

/* calculate lnft for each rip */
/* Calculate the lnft for each rip-width */
for(i=0; i<MAXRIPS && ripvalue[i].rip_width!=0.0F; ++i)
{
    rip_lnft[i] = 0.0F;

    /* Search the solution for the rip-width */
    for(j=0; j<6 && atmarket_solution.rip_width[j]!=0.0F; ++j)
    {
        /* If present in solution, add the lnft */
        if(ripvalue[i].rip_width==atmarket_solution.rip_width[j])
        {
            rip_lnft[i] += bd_lnft;
        }
    }
}

/* Get the time the solution will be sent */
time2 = clock() + 10;
while(time_bd_out>time2)
{
    /* Calculate lnft per minute for each rip */
    /* Null hypothesis: production less than consumption */
    for(i=0; i<MAXRIPS && ripvalue[i].rip_width!=0.0F; ++i)
    {
        /* Calculate production time difference */
        temp = (time_bd_out - time_rip[i]) / CLOCKS_PER_SEC;

        /* Interval per minute */
        if(temp>SEC_PER_MIN)
        {
            temp = SEC_PER_MIN;
        }
        else if(temp<=0.0F)
        {
            temp = SEC_PER_MIN;
        }

        /* Convert to production rate per board */
        temp = SEC_PER_MIN / temp;

        /* Add short and long-term rate contribution */
        /* Contribute 5% per 3 minute interval */
        /* Contribute 95% per 5 minute interval */
        tmp_sum = rip_lnft[i] * temp * 3_MIN_RATE;
        tmp_sum += rip_lnft[i] * temp * 5_MIN_RATE;

        /* Check if production greater than consumption */
        if(tmp_sum>production[i].consumption)
        {
            goto best_bd_out;
        }
    }

    /* Can it be processed faster? */
    time_bd_out -= 10;
}

```

```

/*****
/*      ADJUST THE WEIGHTS USING      */
/*  INDIVIDUAL RIP PRODUCTION RATE  */
*****/

/* From GOTO branch */
best_bd_out:

/* Update lnft per minute production */
for(i=0; i<MAXRIPS && ripvalue[i].rip_width!=0.0F; ++i)
{
    /* Calculate the production time difference */
    temp = (time_bd_out - time_rip[i]) / CLOCKS_PER_SEC;

    /* Re-adjust time if production */
    if(rip_lnft[i]>0.0F)
    {
        time_rip[i] = time_bd_out;
    }

    /* Cap the time */
    if(temp>SEC_PER_MIN)
    {
        temp = SEC_PER_MIN;
    }
    else if(temp<=0.0F)
    {
        temp = SEC_PER_MIN;
    }

    /* Convert to minimum metric */
    if(temp>4.0F)
    {
        temp = SEC_PER_MIN / temp; //per minute
    }
    else
    {
        temp = 15.0F; //4 sec nominal
    }

    /* Save production rate in lineal feet per rate */
    production[i].lnft = rip_lnft[i] * temp;
    temp = 1.0F - (1.0F / temp);

    /* Maintain short-term statistics */
    production[i].st_lnft *= 1.0F - temp;
    production[i].st_lnft += rip_lnft[i];

    /* Cap the short-term stat */
    if(production[i].st_lnft>500.0F)
    {
        production[i].st_lnft = 500.0F;
    }
    if(production[i].st_lnft<0.0F)
    {
        production[i].st_lnft = 0.0F;
    }
}

```

```

/* Maintain long-term stats per 5 minutes */
production[i].lt_lnft *= 1.0F - (temp * 0.2F);
production[i].lt_lnft += rip_lnft[i] * 0.2F;

/* Cap the 5 min stats */
if(production[i].lt_lnft>400.0F)
{
    production[i].lt_lnft = 400.0F;
}
if(production[i].lt_lnft<=0.0F)
{
    production[i].lt_lnft = 0.0F;
}

/* Update production */
production[i].production *= 0.6F;
production[i].production += production[i].lt_lnft * 0.4F;

production[i].production *= 0.8F;
production[i].production += production[i].st_lnft * 0.2F;

production[i].production *= 0.92F;
production[i].production += production[i].lnft * 0.08F;

if(production[i].production>400.0F)
{
    production[i].production = 400.0F;
}
if(production[i].production<=0.0F)
{
    production[i].production = 0.0F;
}

/* Update the production weights */
/* Decrement the weights if production > consumption */
if(production[i].production > production[i].consumption)
{
    /* Adjust the finger-joint weights first */
    /* Check for allowed value loss in shorts or longs */
    if((mktP_loss[i].shorts[lift_grade]<0.5F
        || mktP_loss[i].longs[lift_grade]<0.5F)
        && ripvalue[i].weight_fj>0.2F)
    {
        /* Decrement by fraction of production difference */
        ripvalue[i].weight_fj -= temp * 0.2F;
    }
    /* Adjust the shorts weights after adjusting finger-joint */
    else if(ripvalue[i].weight_shorts
        >mktP_loss[i].shorts[lift_grade]
        && ripvalue[i].weight_shorts>0.1F)
    {
        /* Decrement by fraction of production difference */
        ripvalue[i].weight_shorts -= temp * 0.1F;
    }
    /* Adjust the longs weights after adjusting shorts */
    else if(ripvalue[i].weight_long>mktP_loss[i].longs[lift_grade]
        && ripvalue[i].weight_long>0.05F)

```

```

{
    /* Decrement by fraction of production difference */
    ripvalue[i].weight_longs -= temp * 0.05F;
}

/* Cap the finger-joint weights at zero value */
if(ripvalue[i].weight_fj<=0.2F)
{
    ripvalue[i].weight_fj = 0.0F;
}

/* Check for maximum value loss case */
if(ripvalue[i].weight_shorts<=mktP_loss[i].shorts[lift_grade])
{
    ripvalue[i].weight_shorts = mktP_loss[i].shorts[lift_grade];
}
/* Cap the shorts weights at zero value */
else if(ripvalue[i].weight_shorts<=0.1F)
{
    ripvalue[i].weight_shorts = 0.0F;
}

/* Check for maximum value loss case */
if(ripvalue[i].weight_longs<=mktP_loss[i].longs[lift_grade])
{
    ripvalue[i].weight_longs = mktP_loss[i].longs[lift_grade];
}
/* Cap the longs weights at zero value */
else if(ripvalue[i].weight_longs<=0.05F)
{
    ripvalue[i].weight_longs = 0.0F;
}
}
/* Increment the weights if production <= consumption */
else
{
    /* Start by increasing the longs weights */
    if(ripvalue[i].weight_longs<1.0F)
    {
        /* Increment by fraction of production difference */
        ripvalue[i].weight_longs += temp * 0.025F;
    }
    /* Next increase the shorts weights */
    else if(ripvalue[i].weight_shorts<1.0F)
    {
        /* Increment by fraction of production difference */
        ripvalue[i].weight_shorts += temp * 0.01F;
    }
    /* Finally, increase the longs weights */
    else if(ripvalue[i].weight_fj<1.0F)
    {
        /* Increment by fraction of production difference */
        ripvalue[i].weight_fj += temp * 0.005F;
    }
}

/* Cap the longs weights at market value */
if(ripvalue[i].weight_longs>1.0F)
{

```

```

        ripvalue[i].weight_long = 1.0F;
    }

    /* Cap the shorts weights at market value */
    if(ripvalue[i].weight_shorts>1.0F)
    {
        ripvalue[i].weight_shorts = 1.0F;
    }

    /* Cap the finger-joint weights given shorts value loss */
    if(mktP_loss[i].shorts[lift_grade]>0.5F)
    {
        /* Cap the shorts weights at market value */
        if(ripvalue[i].weight_fj>1.0F)
        {
            ripvalue[i].weight_fj = 1.0F;
        }
    }
    else
    {
        /* Cap the shorts weight at half market value */
        /* Effect is an artificial production target */
        /* Assumption is weight has previously decremented */
        if(ripvalue[i].weight_fj>0.5F)
        {
            ripvalue[i].weight_fj = 0.5F;
        }
    }
}

/* All done! */
/* Send Rip Solution ASAP */
return(0);
}

/*****

//////TITLE:      CUT A FRAME RIP
//////FUNCTION:   cutaframerip
//////SCOPE:      SUBSYSTEM
//////PURPOSE:
//   Take cuts from a single rip in a board.
//   Tally each cut stock into "old" structure.
//   Saw kerfs are included.
//
//////INPUTS:
//   Called from cutitup.
//   Variables:  TYPE      Name      DESC
//               ----      -
//               int       ripnum    Id of rip-width to process
//               float     ylead     leading edge of rip-width
//               float     ytrail    trailing edge of rip-width
//
//   Cut order data:
//   typedef struct {
//       float rip_width;

```

```

//      float prime_cutln;
//      int  onsch;
//      int  oncore;
//      int   numshorts;
//      float length[22];
//      float weight_fj;
//      float weight_shorts;
//      float weight_longs;
//      float wanewidth;
//      float shrink;
//      float max_defect;
//      float min_defect;
//      int   oneside;
//      float slicer;
//      int   numknots2;           //2 times number of knots in array
//      float x[MAXKNOTS];
//      float y[MAXKNOTS];
//      int   side[MAXKNOTS];
//      }ripvalage;
//      extern ripvalage ripvalue[MAXRIPS];
//
//////ASSUMPTIONS:
//      Solution is pre-screened and possible.
//
//////OUTPUTS:
//      Alternate rip solution data:
//      typedef struct {
//          float base;
//          float atmarket;
//          float lealedge;
//          float mktP_sum[MAXRIPS];
//          float pmix[MAXRIPS];
//          float ripn[MAXRIPS];
//          float mktP_core[MAXRIPS];
//          float lnin_core[MAXRIPS];
//          float mktP_fj[MAXRIPS];
//          float lnin_fj[MAXRIPS];
//          float lnin_shorts[MAXRIPS];
//          float mktP_longs[MAXRIPS];
//          float lnin_longs[MAXRIPS];
//          float mktP_shorts[MAXSHORTS][MAXRIPS];
//      }bds;
//      extern bds board[386];
//
//////AUTHOR:
//      John Goulding, 1996.
//

```

*****/

```

void cutaframerip(int ripnum, float ylead, float ytrail)
{
    int i, j, ii, jj, kk;           //scan counters
    int gotframe, gotclear;
    int gotmore, gotlongs;
    int topknot, botknot;           //frame cut flags
    int top_tmp, bot_tmp;

```

```

int ndx_tmp, shortndx;          //cut to what index
float xold_frame, xold_clear;   //previous positions
float xnew_frame, xnew_clear;   //possible cut lengths
float xmax_frame, xmax_clear;
float xtmp_frame, xtmp_clear;
float xage_frame;
float mktP_frame, mktP_clear;  //temp value storage

/* Initialize stuff */
kk = 0;
topknot = 0;
botknot = 0;

/* The following loop scans all the knots for an */
/* intersection with the current rip. It uses knot */
/* data specific to the rip width being examined. */

/* Scan all knots starting with 3/8 edge cleanup cut */
xold_frame = EDGE_CUT;
xold_clear = EDGE_CUT;

for(i=0, j=1; i<ripvalue[ripnum].numknots2; i+=2, j+=2)
{
    /* Search for new knots in rip */
    if(ripvalue[ripnum].y[i]<ytrail && ripvalue[ripnum].y[j]>ylead)
    {
        /* Check if overlapping knot */
        if(ripvalue[ripnum].x[i] < xold_clear)
        {
            /* Bigger knot then overlapping one? */
            if(ripvalue[ripnum].x[j] > xold_clear)
            {
                topknot += ripvalue[ripnum].side[i];
                botknot += ripvalue[ripnum].side[j];

                /* Change the old frame start point? */
                if(topknot>0 && botknot>0)
                {
                    topknot = ripvalue[ripnum].side[i];
                    botknot = ripvalue[ripnum].side[j];

                    /* Do we still have valid frame? */
                    if(topknot>0 && botknot>0)
                    {
                        topknot = 0;
                        botknot = 0;
                        xold_frame = ripvalue[ripnum].x[j] + KERF;
                    }
                    else
                    {
                        xold_frame = xold_clear;
                    }
                }
            }
        }
        /* Old frame start is still valid */
        /* Change the old clear start point */
        xold_clear = ripvalue[ripnum].x[j] + KERF;
    }
}

```



```

    }
}
/* found new knot */
else
{
    /* Reset clear cut parameters */
    xnew_clear = ripvalue[ripnum].x[i];
    xmax_clear = xnew_clear - xold_clear;

    topknot += ripvalue[ripnum].side[i];
    botknot += ripvalue[ripnum].side[j];

    /******
    /* START FRAME OR CLEAR */
    /* PRODUCT SEARCH */
    /******

    /* Should we look ahead for a frame cut? */
    if(topknot>0 && botknot>0) //no
    {
        /* Reset frame cut parameters */
        gotmore = 0;
        xnew_frame = xnew_clear;
        xage_frame = xnew_frame;
        xmax_frame = xnew_frame - xold_frame;
    }
    /* Look for long clears in the frame cut also */
    else //yes
    {
        /* Set temporary cut scratchpad */
        gotmore = 1;
        xage_frame = ripvalue[ripnum].x[j];
        xtmp_clear = xage_frame + KERF;

        top_tmp = topknot;
        bot_tmp = botknot;

        /* Measure frame distance and compare to product */
        for(ii=i, jj=j; (top_tmp==0 || bot_tmp==0)
            && (ii<ripvalue[ripnum].numknots2)
            && (xmax_frame<framesch[ripnum].maxxshorts);
            ii+=2, jj+=2)
        {
            /* Search for new knots in rip */
            if(ripvalue[ripnum].y[ii]<ytrail
                && ripvalue[ripnum].y[jj]>ylead)
            {
                top_tmp += ripvalue[ripnum].side[ii];
                bot_tmp += ripvalue[ripnum].side[jj];

                /* Find highest value clear in frame */
                xtmp_clear = ripvalue[ripnum].x[ii] - xtmp_clear;
                if(xtmp_clear>xmax_clear)
                {
                    /* Set maximum clear to remaining clear */
                    xmax_clear = xtmp_clear;
                }
            }
        }
    }
}

```

```

        /* Keep next longest frame ahead of longest clear */
        if(topknot>0 && botknot>0)
        {
            xage_frame = ripvalue[ripnum].x[ii];
        }
        else
        {
            xage_frame = ripvalue[ripnum].x[jj];
        }
    }

    /* Accumulate product length plus saw kerf */
    xtmp_clear = ripvalue[ripnum].x[jj] + KERF;
}

/* Reset frame cut parameters */
xnew_frame = ripvalue[ripnum].x[ii-2];
xmax_frame = xnew_frame - xold_frame;
}

/* Basic loop to find frame over clear */
/* Find highest frame value, if any, possibly after frame */
while(xmax_frame >= framesch[ripnum].minxshorts)
{
    /* Set null hypothesis */
    gotframe = 0;
    mktP_frame = 0.0F;
    mktP_clear = 0.0F;

    /* First find frame longs, if any */
    ndx_tmp = framesch[ripnum].numlongs;
    for(ii=0; ii<ndx_tmp; ++ii)
    {
        /* Take long if less than frame length */
        if(framesch[ripnum].xlongs[ii]<=xmax_frame)
        {
            gotframe = 1;
            gotlongs = 1;
            xtmp_frame = framesch[ripnum].xlongs[ii];
            mktP_frame = framesch[ripnum].Plongs[ii];
            break;
        }
    }

    /* Find frame shorts, if reqd/any */
    if(!gotframe)
    {
        /* Next look for frame shorts, if any */
        ndx_tmp = framesch[ripnum].numshorts;
        for(ii=0; ii<ndx_tmp; ++ii)
        {
            /* Take shorts if less than frame length */
            if(framesch[ripnum].xshorts[ii]<=xmax_frame)
            {
                gotframe = 1;
                gotlongs = 0;
                shortndx = framesch[ripnum].shortndx[ii];
            }
        }
    }
}

```

```

        xtmp_frame = framesch[ripnum].xshorts[ii];
        mktP_frame = framesch[ripnum].Pshorts[ii];
        break;
    }
}

/* Next, look for clear product */
/* Check if clear length > minimum product */
if(xmax_clear >= clearsch[ripnum].minxshorts)
{
    /* Set null hypothesis */
    gotclear = 0;

    /* Find clear lineal, if any */
    if(xmax_clear >= clearsch[ripnum].xlineal)
    {
        gotclear = 1;
        mktP_clear = xmax_clear * clearsch[ripnum].Plineal;
    }

    /* Find clear longs, if reqd/any */
    if(!gotclear)
    {
        /* Search for longs */
        ndx_tmp = clearsch[ripnum].numlongs;
        for(ii=0; ii<ndx_tmp; ++ii)
        {
            /* Take longs if less than clear length */
            if(clearsch[ripnum].xlongs[ii] <= xmax_clear)
            {
                gotclear = 1;
                mktP_clear = clearsch[ripnum].Plongs[ii];
                break;
            }
        }
    }

    /* Find clear shorts, if reqd/any */
    if(!gotclear)
    {
        /* Search for shorts */
        ndx_tmp = clearsch[ripnum].numshorts;
        for(ii=0; ii<ndx_tmp; ++ii)
        {
            /* Take shorts if less than clear length */
            if(clearsch[ripnum].xshorts[ii] <= xmax_clear)
            {
                mktP_clear = clearsch[ripnum].Pshorts[ii];
                break;
            }
        }
    }
} //end of IF find clear

/* Determine what product has greatest value */
/* Use crosscut operator methodology */
/* Consider only the immediate decision */

```

```

/* Take frame now and perhaps clear next */
if(mktP_frame > mktP_clear)
{
    /* Reset all x units */
    xold_frame += xtmp_frame + 0.125F;
    xold_clear = xold_frame;
    xmax_frame = xnew_frame - xold_frame;
    xmax_clear = xnew_clear - xold_frame;

    /* Tally cut stock */
    if(gotlongs)
    {
        /* Accumulate longs in rip solution array */
        board[bd_index[384]].lnin_long[ripnum] += xtmp_frame;
        board[bd_index[384]].mktP_long[ripnum] += mktP_frame;
    }
    else
    {
        /* Accumulate shorts in rip solution array */
        board[bd_index[384]].lnin_short[ripnum] += xtmp_frame;
        board[bd_index[384]].mktP_short[shortndx][ripnum] +=
            mktP_frame;
    }
}
/* Take clear now and perhaps frame next */
else
{
    /* Check for additional searches */
    if(gotmore)
    {
        /* Try short frame over first short clear, if any */
        gotmore = 0;
        xmax_clear = xnew_clear - xold_clear;
        xnew_frame = xage_frame;
        xmax_frame = xnew_frame - xold_frame;
    }
    /* Quit frame case and find clear */
    else
    {
        break;
    }
}
} //end of WHILE find frame over clear

/*****/
/* START ALL CLEAR */
/* PRODUCT SEARCH */
/*****/

/* Reset the maximum clear board length */
xmax_clear = xnew_clear - xold_clear;

/* Find clear, if any and possibly after frame and clear */
/* But first take fj, if any and if clear */
if(xmax_clear >= clearsch[ripnum].minxshorts)
{
    /* Reset old x units */

```

```

xtmp_frame = xold_clear - xold_frame;
xold_frame = xold_clear;

/* Take F.J. now */
if(xtmp_frame>framesch[ripnum].xminfj)
{
    board[bd_index[384]].lnin_fj[ripnum] += xtmp_frame;
}

/* Jump label */
find_clear:

/* Set null hypothesis */
gotclear = 0;

/* Find clear lineal, if any */
if(xmax_clear>=clearsch[ripnum].xlineal)
{
    gotclear = 1;
    xtmp_clear = xmax_clear;

    /* Tally cut stock */
    board[bd_index[384]].lnin_long[ripnum] += xtmp_clear;
    board[bd_index[384]].mktP_long[ripnum]
        += xtmp_clear * clearsch[ripnum].Plineal;
}

/* Find clear long, if any */
if(!gotclear)
{
    /* Search through long products */
    ndx_tmp = clearsch[ripnum].numlongs;
    for(ii=0; ii<ndx_tmp; ++ii)
    {
        /* Take a long if it fits into clear board */
        if(clearsch[ripnum].xlongs[ii]<=xmax_clear)
        {
            gotclear = 1;
            xtmp_clear = clearsch[ripnum].xlongs[ii];

            /* Tally cut stock */
            board[bd_index[384]].lnin_long[ripnum] +=
                xtmp_clear;
            board[bd_index[384]].mktP_long[ripnum] +=
                clearsch[ripnum].Plongs[ii];
            break;
        }
    }
}

/* Find clear short, if any */
if(!gotclear)
{
    /* Search through short products */
    ndx_tmp = clearsch[ripnum].numshorts;
    for(ii=0; ii<ndx_tmp; ++ii)
    {
        /* Take a short if it fits into clear board */

```

```

        if(clearsch[ripnum].xshorts[ii]<=xmax_clear)
        {
            gotclear = 1;
            xtmp_clear = clearsch[ripnum].xshorts[ii];

            /* Tally cut stock */
            board[bd_index[384]].lnin_shorts[ripnum] +=
                xtmp_clear;
            board[bd_index[384]].
                mktP_shorts[clearsch[ripnum].shortndx[ii]][ripnum]
                += clearsch[ripnum].Pshorts[ii];
            break;
        }
    }

    /* Reset old x units ie take clear */
    xold_clear += xtmp_clear + 0.125F;
    xold_frame = xold_clear;
    xmax_clear = xnew_clear - xold_clear;

    /* Find more clear cut stock? */
    if(xmax_clear >= clearsch[ripnum].minxshorts)
    {
        goto find_clear; //yes
    }

} //end of IF clear

/* Change the old frame start point? */
if(topknot>0 && botknot>0 && xold_frame<ripvalue[ripnum].x[j])
{
    /* Check knot face */
    topknot = ripvalue[ripnum].side[i];
    botknot = ripvalue[ripnum].side[j];

    /* Do we still have valid frame? */
    if(topknot>0 && botknot>0) //no
    {
        topknot = 0;
        botknot = 0;
        xtmp_frame = ripvalue[ripnum].x[i] - xold_frame;
        xold_frame = ripvalue[ripnum].x[j] + 0.125F;

        /* Take core */
        /* Calculate value later */
        board[bd_index[384]].lnin_core[ripnum] +=
            ripvalue[ripnum].x[j] - ripvalue[ripnum].x[i];

        /* Take F.J. */
        if(xtmp_frame>framesch[ripnum].xminfj)
        {
            /* Calculate value later */
            board[bd_index[384]].lnin_fj[ripnum] += xtmp_frame;
        }
    }
}

```

```

        /* Advance clear xold for a new clear case */
        if(xold_clear<ripvalue[ripnum].x[j])
        {
            xold_clear = ripvalue[ripnum].x[j] + 0.125F;
        }

        } //end of ELSE new knot
    } //end of IF new knot in rip
} //end of FOR search knots

/* Calculate value of F.J. now */
board[bd_index[384]].mktP_fj[ripnum] +=
    board[bd_index[384]].lnin_fj[ripnum]
    * framesch[ripnum].Pfj;

/* Special tally for wane */
if(ripvalue[ripnum].rip_width<2.0F
    && (ylead<=bd_inside_wane || ytrail>=bd_outside_wane))
{
    board[bd_index[384]].mktP_core[ripnum] +=
        (board[bd_index[384]].lnin_fj[ripnum]
        + board[bd_index[384]].lnin_core[ripnum]) *
        framesch[ripnum].Pcore;
}

/* Special tally for core */
else if(ripvalue[ripnum].oncore && ylead<=w9 && ytrail>=w9)
{
    board[bd_index[384]].mktP_core[ripnum] +=
        board[bd_index[384]].lnin_core[ripnum]
        * framesch[ripnum].Pcore;
}

} //all done!

```