MACHINE VISION APPLIED TO WHEAT GRADING
-- INITIAL RESEARCH --

by

Edward H. Wright

A thesis
presented to the University of Manitoba
in partial fulfillment of the
requirements for the degree of
Master of Science
in
Electrical Engineering

Winnipeg, Manitoba

ISBN Ø-315-33864-4

MACHINE VISION APPLIED TO WHEAT GRADING
--INITIAL RESEARCH--

BY

EDWARD HAROLD WRIGHT

A thesis submitted to the Faculty of Graduate Studies of
the University of Manitoba in partial fulfillment of the requirements
of the degree of

MASTER OF SCIENCE

I hereby declare that I am the sole author of this thesis.

I authorize the University of Manitoba to lend this thesis to other institutions or individuals for the purpose of scholarly research.

Edward H. Wright

I further authorize the University of Manitoba to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

Edward H. Wright

The University of Manitoba requires the signatures of all persons using or photocopying this thesis. Please sign below, and give address and date.

Patricia Paton
217-80 Baylar

February 17/89.

ABSTRACT

The objective was to design the techniques and corresponding computer software initially required during the development of a successful, machine vision based, automated wheat grading system.

The research was guided by an understanding of the current Canadian wheat grading system. This understanding enabled the anticipation of the objects that could appear, the selection of a processing organization which is in accord with standard grading procedures and the determination of which features are required to judge grade.

Four general phases of software development occurred. First, the Image Manipulation Package was developed to operate a custom built digital imaging system. Second, four different techniques were invented for computer perception of the objects appearing in a wheat grading image. A technique was also implemented to provide shape description of these objects. Third, techniques were devised to detect two anatomical parts of the wheat kernel: the crease and the germ. Finally, a 2-D signal processing technique was applied to analyze the visible kernel surface texture. This provided four facilities: rotation normalization, texture model estimation, power density spectrum image generation and texture image synthesis.

All of the programs, except that for texture model estimation, generally performed well. Clear recommendations are given to rectify the deficiencies of this routine.

ACKNOWLEDGEMENTS

With respect to the completion of my research and thesis I wish to acknowledge my advisor, Dr. Edward Shwedyk, for his wise counsel, understanding and inspiration which carried me through to the end of this ordeal, Mr. Jack Sill for his assistance when I was programming the digital image acquisition system, Dr. Harry Sapirstein for his guidance in the plant sciences aspects of my research and for his strengthening of my base of references, and finally, Mr. Allan McKay for his photographic expertise in the production of the prints in this thesis and the slides in its presentation.

With respect to financial assistance I wish to acknowledge the Natural Sciences and Engineering Research Council both for a Strategic Grant which provided much of the equipment and for a Postgraduate Scholarship, the Students Finance Board of the Government of Alberta for student loans, and especially my grandfather, Mr. Harold E. Wright, for his more recent and invaluable financial assistance which ensured that I could complete this thesis, even when the going got tough.

TABLE OF CONTENTS

LIST OF FIGURES

Chapter I

INTRODUCTION

1.1 OBJECTIVES OF THIS RESEARCH

The Canadian wheat grading system is part of Canada's multi-billion dollar grain production and handling system. This grading system is designed to relate price to quality on a consistent basis and to facilitate grain handling. In the past this grading system has worked well for Canada's grain industry.

One of the problems with the Canadian wheat grading system is that it relies heavily on the visual assessment of grain samples. Consequently the grading procedure is very subjective. In fact, some of the degrading factors considered during the grading procedure, such as frost damage, have tolerances which are impossible to state in a manual. Rather their proper use is dependent on the experience and training of the grain inspector [Canadian Grain Commission, 1981].

Another problem is that, while the remainder of the grain handling and transportation system is undergoing a major shift to computerization [Candlish, 1984], the grading procedures used within the Canadian wheat grading system are

performed entirely by human inspectors. Computerization of the grading system is particularly attractive in light of today's efforts to streamline the grain handling and transportation system in Canada.

At the University of Manitoba, a long term research project is being undertaken jointly by the Department of Plant Science and the Department of Electrical Engineering to develop an automated, computer assisted wheat grading system and thereby alleviate both of the above problems. The research for this thesis was pursued in association with that project. The desired grading system would replace today's highly trained human inspector or would at least provide an objective quantitative assessment of the necessary grading features for an inspector. The system would be sufficiently intelligent to not require human interaction during its operation. Such a system could also interface with present day or future computer systems that control grain binning and inventory maintenance.

The research team is considering several approaches. The first is a stand-alone computer image analysis system. The second approach is to supplement the computer image analysis with other tests such as test weight, protein content and computer aided variety identification using the gliadin electrophoregram. The third approach is to develop a completely new set of grading criteria.

The focus of this thesis was the computer image analysis part of the research team's work. Since this was the first work on image analysis for the research team, this research has been of a rudimentary nature. The main objectives of this research were to create the software that would first enable basic computer understanding of a wheat grading image and then extract the basic features necessary for grading wheat. This has necessitated an understanding of current wheat grading techniques in Canada so that the correct image features could be chosen. The statistical analysis of the acquired features was not undertaken.

This research had limitations which may be lessened for later work by the research team. Only the hard red spring class of wheat was considered because it is the most popular class of wheat grown. Only black and white imaging was used, both for simplicity and since most of the visual degrading factors could be determined without the use of colour. Mechanical handling of individual objects was not assumed so that kernels could appear in any orientation. However it was assumed that only a single layer of scattered objects lying on a flat white surface would be produced by some prior elementary mechanical method. Consequently occluded objects were not allowed and only rarely would objects touch. Finally, the mode of illumination was restricted to diffuse white front lighting. Therefore the grading system would have to contend with some object shadow

and would view an object only with light reflected from its surface. Separate research is now being carried out by the team to determine the usefulness of transmitted light resulting from back lighting in determining wheat kernel vitreousness.

1.2 REVIEW OF RELATED RESEARCH

Computer vision is a large and rapidly growing field. Many effective low-level early image processing and high-level cognitive image understanding techniques have been developed over the past decade [Ballard, 1982]. Typical areas of application include biomedical imaging, aerial and satellite photo interpretation, industrial robotics and military and artificial intelligence research. However very little research has been done in applying computer image analysis to cereal grain inspection.

Two problems stand out in the sparse body of literature surrounding computerized cereal grain inspection. The first is the discrimination between types of grain such as corn, wheat, soybeans, oats and rye. Brogan and Edison [1974] and Brogan and Inguanzo [1978] have worked extensively on the pattern recognition aspects of this problem. The features that they extracted were limited to the length, width, depth and planiform area of each kernel. Each kernel was handled individually by a special mechanism to place it in the proper orientation before being optically scanned. This form of

size feature extraction is probably too cumbersome for use in real life grain inspection. Also, the problem that they dealt with is simpler than the grading problem. Grading features are more obscure than the dimensional features that they used to differentiate between grain types.

Recently, Draper and Travis [1984] used similar features obtained with a low-cost computerized imaging system to describe the shape of barley, wheat, lettuce, grass, wild oats and cleaver seeds. In addition they utilized the shape factor and aspect ratio. However they did not attempt to classify seeds based on these features.

The second problem dealt with in the relevant research is rice grading by using interactive computer image analysis. Recent research by Goodman and Rao [1984] found that an interactive computer image analysis system can measure rice kernel sizes more accurately and quickly for grading. Their work does not approach the problem of automating the grading procedure. Again, the features that they extract are not sufficient for grading wheat.

In summary, wheat grading is a new application for computer image analysis. However, many of the necessary techniques are already in existence and merely need to be applied to the problem.

1.3 WHEAT GRADING IN CANADA

1.3.1 Background

Grain grading in Canada is organized and regulated by the Canadian Grain Commission [Canada Grains Council, 1982: 49]. The Commission has determined numerically designated grades for Canadian wheat and has determined which factors are to be used in the grading procedure. The grade designations and grading factors were chosen based on their effect on the technological value of wheat, [Tipples, 1982], in other words milling and bread making quality, the ability of current technology to clean the grain and to a certain extent on the aesthetic appearance of the grain.

The statutory grades of hard red spring wheat in Canada which are of interest to the research team are No.1, No.2 and No.3 C.W. Red Spring and No.1 and No.2 Canada Utility wheat. Lower numbers indicate higher quality.

1.3.2 Grading Factors

(The general content and organization of this subsection is from Duke [1982: 8].)

The principal grading factors used in grading wheat are:

A) TEST WEIGHT

B) VARIETAL PURITY

C) VITREOUSNESS

D) SOUNDNESS

E) MAXIMUM LIMITS OF FOREIGN MATERIAL

A) TEST WEIGHT. This is one of the few objectively determined grading characteristics and it was historically one of the first to be used in grading grain. Under normal growing conditions it is of only small importance in determining the grade of a sample.

B) VARIETAL PURITY. The Marquis variety of wheat is the standard of quality for the top two grades of Hard Red Spring Wheat, although other varieties of quality not lower than Marquis can be mixed in a sample without lowering the grade. Other varieties not equal in quality to Marquis will qualify only for No.3 C.W. Red Spring or Canada Feed Wheat.

The grades of No.1 and No.2 Canada Utility are for the relatively new class of wheat called "Utility Wheat". The Glenlea variety is typical of this class.

Determining the varieties composing a sample is the most difficult task facing a wheat inspector. To facilitate in performing this task, the Canadian Grain Commission will licence a variety for production only if it is visually distinguishable from all other varieties or visually identical to another previously approved variety of similar quality.

Three visually determined kernel characters are the major factors in identifying varieties -- colour, texture and shape [Owen and Ainslie, 1971: 55].

Colour is one of the most obvious characters. It is most useful in distinguishing classes of wheat such as red spring, white spring, amber durum, etc. Colour is also used to classify the red wheats as light red, medium red or dark red. Classifying the degree of redness is useful in distinguishing variety. However, colour is very sensitive to surface defects such as starchiness or a bleached surface, which are caused by poor environmental conditions during growing and harvest. The resulting irregularity of colour as a feature indicates that the limitation to black and white images in this research is not a serious drawback to varietal determination.

Kernel texture is a relatively constant varietal characteristic which can help discriminate between hard, semi-hard or soft varieties of wheat. The endosperm of a kernel makes up the majority of the visible kernel surface and is the part whose surface texture is affected by the kernel hardness. A hard kernel has a dark translucent or vitreous endosperm while a soft kernel has a light or opaque endosperm.

The shape of the kernel outline can be described as short, midlong or long and ovate, elliptical or oval (Figure 1.1). This is one of the most important characters in

determining variety. It is also important to other stages in the grading process.

Characters related to various anatomical parts of the kernel are of less importance but are sometimes used in varietal identification. The characters of each part that are used are usually shape, size and texture. The parts considered are the cheeks and the crease lying between them, both of which lie on the ventral side of the kernel, the germ which includes the dormant embryo of the seed and lies on the dorsal side of the kernel, the brush which is on the opposite end from the germ, the bran or skin of the kernel, and the glumes which are kernel husks that sometimes remain in a sample after threshing. The kernel anatomy is shown in Figure 1.2.

C) VITREOUSNESS. Vitreous kernels have a sound surface and have a natural colour and translucence which denotes hardness of the kernel. Vitreous hard red spring wheat kernels have an overall dark and slightly red surface. Non-vitreous kernels are damaged or have a starchy surface which is indicated by its overall lighter shade.

Minimum percentages by weight of hard vitreous kernels are set for the top two grades of hard red spring wheat.

D) SOUNDNESS. Higher grades of wheat require a higher degree of kernel soundness. A broad range of kernel surface defects can degrade the degree of soundness of a wheat sample.

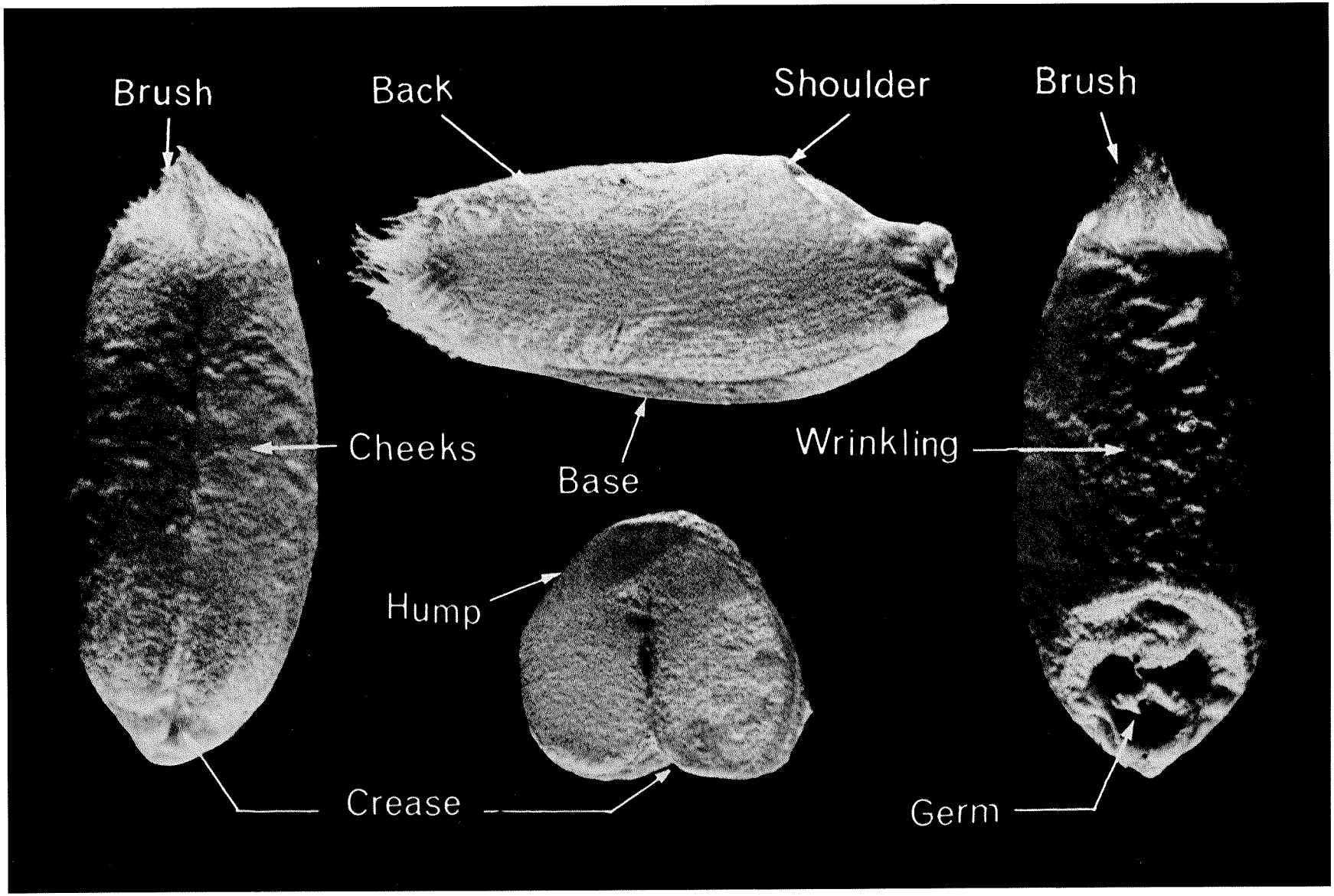
Figure 1.1: Major Shapes -- Dorsal View

Reproduced from Figure 46 on page 57 of
Owen and Ainslie [1971].

Elliptical Ovate Oval

Figure 1.2: Kernel Anatomy

Reproduced from Figure 47 on page 58 of
Owen and Ainslie [1971].


Brush
Back
Shoulder
Brush
Cheeks
Base
Wrinkling
Hump
Crease
Germ

Poor growing conditions can degrade wheat kernel soundness. Kernels from plants that were lacking water will often be immature, small and shrivelled. Kernels from plants which have had secondary growth after surviving a drought will often be small and bright green in colour. These are called "grass green" kernels. A very common surface defect is bleaching which occurs when the plant must survive a series of excessively rainy and dry periods. A bleached kernel has an overall medium light shaded surface and is opaque to transmitted light. Frost damage often will cause a pebbled or shrivelled kernel surface.

The degree of kernel soundness can be degraded by the effects of disease. Disease can occur while the plants are growing in the field, after they have been swathed and are lying in the field waiting to be combined, or while the resulting grain is in storage. Several fungous diseases start in the embryo of the kernel and cause a dark-brown discolouration of the germ called blackpoint, a common degrading factor. This discolouration can spread over more of the kernel and is then called smudge, another common degrading factor. Other diseases that are less common are ergot and sclerotinia. While in storage, grain can suffer the effects of mildew, mold and, in the final stages, rot. At the onset of mildew, gray tufts appear at the distal ends of the kernel.

Breakage is a common surface defect which degrades the degree of soundness. Breakage of wheat kernels results from excessive mechanical handling. It is characterized by a broken kernel surface and the exposed bright white patches of the starchy endosperm.

Many other less common forms of damage can degrade kernel surface soundness. Rainy conditions while swathed wheat lies in a field can cause sprouting from the embryo which is a result of premature germination. A sprouted kernel will have a distinctly swollen germ and the bran will be noticeably split over the germ from apparent growth. Several types of insect damage can occur. Midge damage takes the form of gouging along the kernel crease while grasshopper or armyworm damage normally appears as chewing along the kernel sides. Other less common factors that degrade kernel surface soundness are fireburning, binburning and staining from kernels coming in contact with foreign substances.

E) MAXIMUM LIMITS OF FOREIGN MATERIALS. Maximum limits of the amount of cereal grains other than wheat and of the amount of foreign material other than cereal grains are set for each grade of wheat. These limits, especially for foreign material other than cereal grains, are very small. Most foreign material in a wheat sample will be the same size as the wheat kernels since the grain is usually cleaned before sampling. Barley kernels are a common foreign material found in wheat samples. Barley kernels have a size

similar to that of wheat kernels but have more pointed ends and are more yellow in colour. Other examples of foreign material are large weed seeds, thistle heads or pieces of stems and stones. Most foreign objects can easily be discriminated from wheat by their shape.

1.3.3 Standard Samples

Each year standard samples of wheat and other grains are prepared for use by grain inspectors [Duke, 1982: 3] [Canada Grains Council, 1982: 53]. Two sets of samples are established, one for use at the producer or primary level and the other for use at the export level. Standard samples are necessary because, as has already been shown, the Canadian grain grading system depends to a large part on the ability of inspectors to visually assess the appearance of grain samples. The standard samples do not change the grade specification, but rather they are used as guides to grading. Standard samples are a visual interpretation of the grade specifications that reflect the growing conditions of the year for which they were prepared.

The use of standard samples stresses two important ideas. First it reaffirms that wheat grading in Canada is very subjective and that therefore the use of an effective automated grading system would be advantageous. Secondly, it indicates that the thresholds of the various visual features used in grading change subtly from year to year. Therefore,

any automated wheat grading system must be capable of periodic fine tuning through the manual entry of new standards. More importantly, the decision structure used in an automated grading system should roughly model the current grading procedure so that new grading criteria set by a Standards Committee can be easily mapped into the machine's decision structure.

1.4 REQUIRED IMAGE FEATURES

The grading factors used in the current Canadian wheat grading system, as outlined in the previous section, indicate certain image features that a computer image analysis system should be capable of extracting for grading wheat. This research has attempted to develop the techniques required to extract those features. This section is an overview of those features.

The most important class of features for wheat grading are those that describe the size and shape of an object's outline. The length, width and area are the most essential elements of this class of features, but others should be included so that a more accurate description of the shape can be made. Shape features used in wheat grading can be used to distinguish wheat kernels from other cereal grains or foreign objects. This function is particularly important to later stages of processing where only objects identified as wheat kernels by shape analysis should be scrutinized fur-

ther for grading factors. The shape of the object outline can also be used to discriminate between durum and common wheat classes, and between broken or shrivelled wheat kernels and plump wheat kernels.

The size and shape of certain anatomical parts of wheat kernels can sometimes be used for varietal determination. However, these features are less regular because of the subtle differences between wheat varieties and so their use has not been pursued in this research.

Surface texture is another broad class of features that is necessary for wheat grading. The description of surface texture usually includes the average gray level of a surface and some description of the variation of surface gray level versus distance and direction. Normally some form of spatial frequency transformation is employed, which facilitates in differentiating between micro-texture (fine) and macro-texture (coarse) components of the texture.

The surface texture of the overall visible kernel surface is useful. Vitreousness can be indicated by a dark endosperm. An exposed starchy endosperm appears as extremely light patches and indicates a broken surface.

A dorsal view of a wheat kernel is the most productive on which to perform a texture analysis. The surface texture on the dorsal side is not dominated by the effects of the crease, which appears on the opposite side of the kernel.

Also, the germ is visible in a dorsal view. A localized texture analysis of just the germ should be performed for each kernel for which a dorsal view is available. The germ is usually the area first and most affected by fungous diseases and is the area in which sprouting occurs. Thus a texture analysis of the germ should facilitate the detection of diseases such as blackpoint and smudge and the detection of sprouting.

The ventral side of the kernel is the most probable side to face up due to the kernel shape. Therefore the ventral view is more common. On this side a localized texture analysis along the crease could detect midge damage.

Wrinkling is a fine surface defect that, when present, affects the entire kernel surface. Wrinkling is caused by frost damage or poor growing conditions. It is most easily discernable when the incident illumination makes a small angle with the local surface. Thus wrinkling should be searched for by using a localized texture analysis near the lateral limb or edge of the kernel image where light from the source of illumination only grazes the surface.

Many of the previously described texture features are location dependent. Obviously, some features are required not only to determine kernel orientation, dorsal or ventral side up, but also to detect and specify the location of the major anatomical features in the image of the visible kernel sur-

face. The detection and specification of only two anatomical features can provide all of the location data required. First, the crease is the most obvious part of the kernel anatomy and its presence or absence specifies the kernel orientation, either dorsal or ventral side up. Therefore the detection of the crease and the specification of its location is necessary to determine the kernel orientation and to guide texture analysis or shape analysis of the crease. Second, if the kernel is dorsal side up then the germ should be located in order to guide texture analysis of the germ.

Colour is a feature that was not used in this research. When restricting the grading problem to the hard red spring class of wheat, as was done in this research, colour is not normally important. However, it is useful in detecting "grass green" wheat kernels and white wheat classes as foreign material. Therefore the use of colour should be considered in future research.

A summary of the desired features discussed in this section is as follows. First, a description of the contour shape of all the objects in an image is required. Second, a decision and a location specification for either the crease or germ, depending on kernel orientation, is required for each object identified as wheat. Finally, a description of the texture of each visible area of interest should be obtained. The texture analysis should characterize the energy of the texture and describe its spatial frequency spectrum.

The areas of interest for texture analysis on the visible kernel surface are the overall visible kernel surface, either or both lateral limbs of the kernel and, when either is available, the germ and the crease.

1.5 OUTLINE OF THIS THESIS

Chapter 2 of this thesis describes the machine language package, IMP, which is the software portion of the custom made digital image acquisition system used during this research. IMP performs many of the typical functions found on commercial imaging systems. Certainly, IMP is not tailored specifically to the wheat grading problem, but the writing of it was necessary for the completion of this research and the needs of the research team.

Chapter 3 presents the object perception problem. Several approaches were developed during the evolution of a satisfactory object detection method and these are presented first. Then, a superior approach is presented which utilizes a priori knowledge of wheat grading image characteristics to constrain the object perception problem. The resulting object detection algorithm is efficient, robust and produces estimated object contours that are accurate even for many poor images. Finally the shape analysis method, that is used for providing features describing each object's contour, is presented.

Chapter 4 describes the techniques devised to detect and specify the anatomical parts of objects identified as wheat kernels. Specifically, the parts detected are the crease and the germ.

Chapter 5 presents a two-dimensional signal processing technique that was used for the texture analysis problem. With it texture is modelled as a statistical process so that not only can features be calculated, but also an estimated power density spectrum and a synthetic texture obeying the model can be produced.

The Software Manual Supplement to this thesis, a volume separate from this one, lists the source code of the programs which implemented the procedures discussed in the above mentioned chapters. These listings are in the same order in which the corresponding procedures are presented.

Chapter II

IMAGE MANIPULATION PACKAGE

2.1 INTRODUCTION

The Image Manipulation Package, hereafter called IMP, is the software portion of the digital image acquisition system used during this research. IMP is a package of 8086 assembler [Intel, 1983] routines written for this research to provide software facilities typically found on commercial imaging systems. The user possessing only an elementary understanding of the image processing architecture of IMP will find the terminal discourse offered by IMP self-explanatory, most of it being in a menu format. IMP is loaded and executed using the Digital Research CP/M-86 operating system [Digital Research, 1982]. Some of the CP/M I/O and disk operating functions are in turn used by IMP. Figure 2.1 shows the main IMP command menu. Figure 2.2 on page 25 shows a block diagram of image processing in IMP.

The hardware portion of the system was previously built by Jack Sill of the Department of Electrical Engineering at the University of Manitoba. The combined, custom made imaging system is intended for use with the automated wheat grading project. The images processed in this research were all initially acquired by it and some of the intermediate

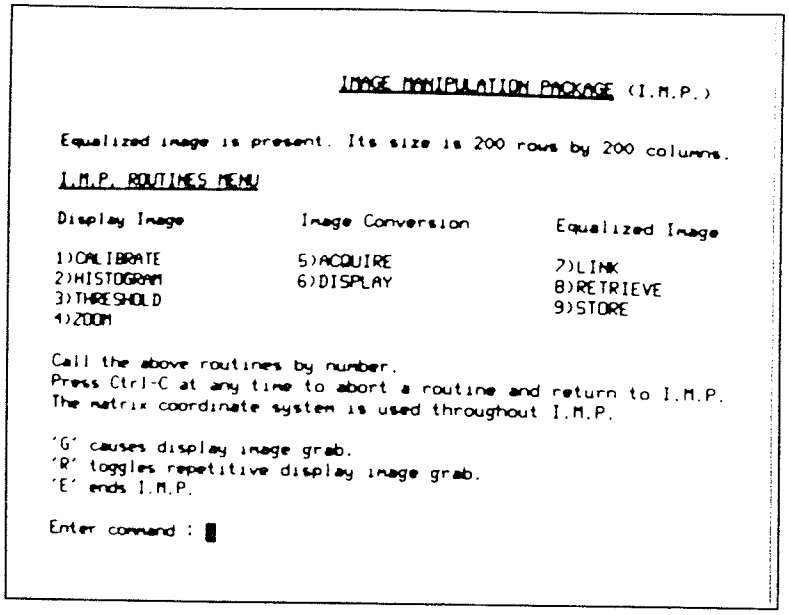


Figure 2.1: Main IMP Command Menu

processed images shown in this manuscript were obtained from the system image display monitor. The reasons for building a custom made imaging system were the substantially reduced cost and the certainty that all of the desired features would be available.

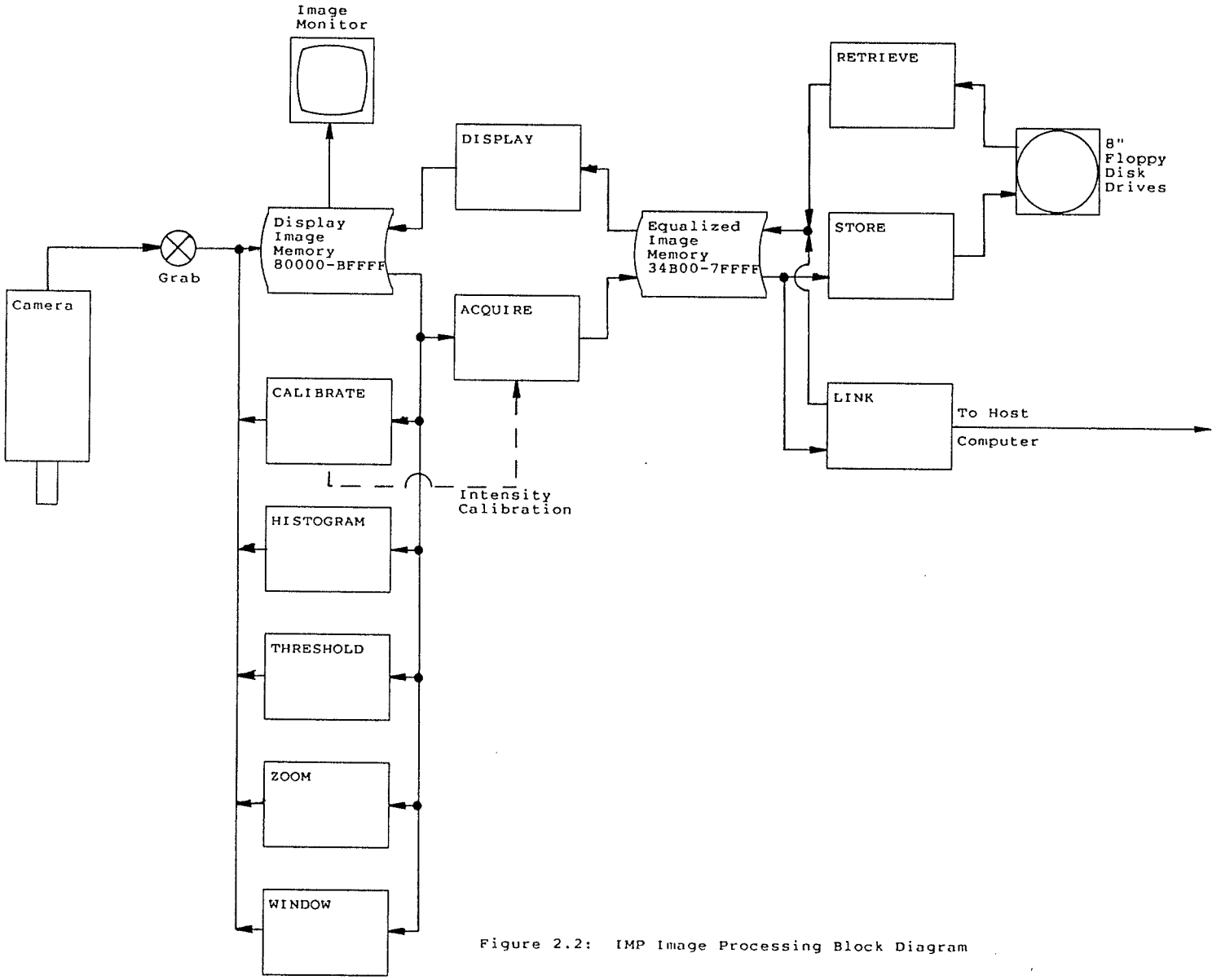


Figure 2.2: IMP Image Processing Block Diagram

2.2 SUMMARY OF THE IMAGING HARDWARE

A block diagram of the digital image acquisition system hardware is shown in Figure 2.3.

Images are provided by a Fairchild CCD3000 black and white video camera in the conventional N. T. S. C. video format. The image detector in this camera is a Charge Coupled Device array which produces 482 rows by 377 columns of usable pixel values. The photosites in this array are arranged on a rectilinear lattice and are precisely spaced 18 microns apart vertically and 30 microns apart horizontally. This arrangement produces images which have more vertical spatial pixel density than horizontal spatial pixel density, a problem which is later solved by the image equalization facility of IMP.

The advantages of this camera over a comparable vidicon tube camera are its higher dynamic intensity range of 1000 to 1, rugged construction and, most importantly for this application, synchronization signals that are made available to the external world. These signals consist of the composite blanking and sync, the field index pulse and the master clock signal. The frame index pulse indicates the start of a new frame when the scan is in the top left corner of field one. The master clock signal indicates when the level of the analog video output corresponds to a new pixel value. These synchronization signals provide timing for the other elements of the imaging system and, in part, permit the digitization of the analog video signal.

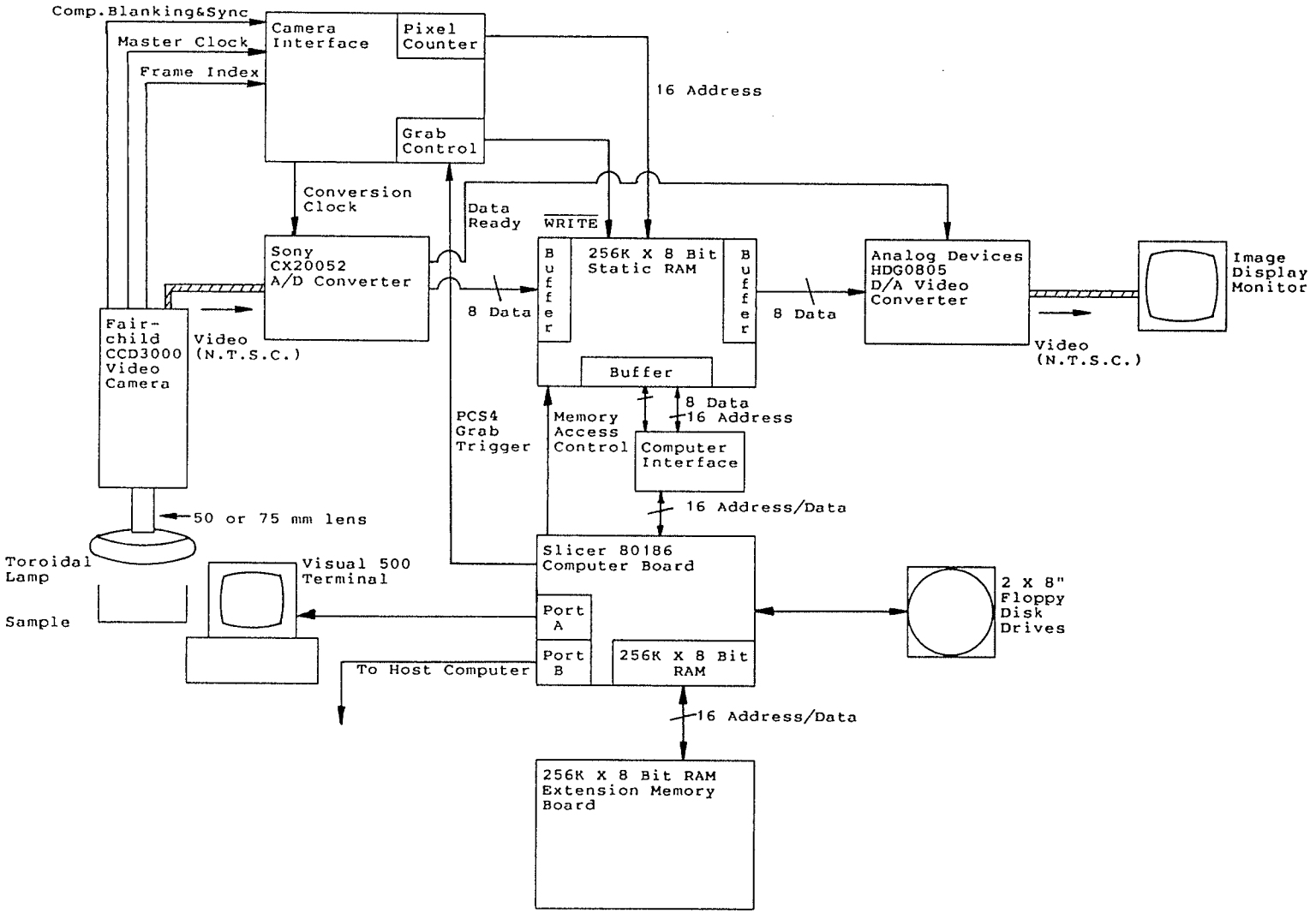


Figure 2.3: Digital Image Acquisition System

The lens system employed for the video camera consists of a 50 mm and a 75 mm 'C' mount fixed focus lens and an extension tube set. This lens system provides a wide range of distortion free image magnifications. At high magnification a single wheat kernel will occupy the entire image, but at the standard magnification used in this research a 256 by 256 pixel window will represent a 2 cm by 2 cm subject area.

The source of illumination for sample lighting was a toroidal fluorescent lamp mounted on an adjustable arm. For normal operation this lamp would be mounted horizontally over the wheat sample and positioned so that the average angle of illumination to the imaged background surface was 45 degrees. The camera would then view the sample from above along the axis of the lamp. This setup produced images with minimal shadow and diffuse omnidirectional illumination of the subject.

Digitization of the analog video camera output signal is accomplished using a Sony CX20052 Analog to Digital flash converter board. This product digitizes a single video frame "real time", or in other words, within the confines of the fifteenth of a second duration frame. This extremely high data rate, which amounts to 7.16 million pixel values per second, requires that fast 100 nS RAM be used for image storage. Each pixel value is represented by one byte thereby allowing 256 distinct gray levels. White is represented as 255 and black as 0. Each A/D conversion is synchronized

with the camera master clock by using a derivative signal from the camera interface board so that a conversion is performed each time a new pixel level is available.

Storage of the digitized image is accomplished using 256 kilobytes of fast 100 nS static RAM. This memory section is equipped with sufficient multiplexing to allow access to the memory bus by the A/D converter during image storage, by the D/A converter during image display and by the Slicer microcomputer during computer access. Access by the computer overrides the other two forms of memory access.

The video signal used to represent a stored image is generated by an Analog Devices HDG0805 Digital to Analog Video Converter board. During image display, data from the image memory is continuously inputted to the D/A converter along with blanking and sync signals from the camera and a data ready pulse from the A/D converter. The conversion of each pixel value is synchronized with the A/D converter data ready pulse which indicates the availability of a new pixel value on the memory data bus. The output from the D/A converter board is a N. T. S. C. analog video signal that is fed to the image display monitor. The image display monitor cannot directly access the camera video output. Consequently the operator is not provided with an instantaneous display of the camera output.

The image digitization, storage and display elements of the imaging system are coordinated by the camera interface board. A major part of this board is the pixel counter. This counter resets at the beginning of each frame and then counts up with the camera master clock, thereby counting pixel locations. The output from this counter continuously addresses the display image memory except during computer access of the this memory section. Thus the display image memory is continuously scanned in synchronism with the camera image scan during both image storage and display. This synchronized scan guarantees that during image storage, any one pixel's value is always stored into the same memory location. However, the video output is stored in the interlaced N. T. S. C. format and includes all of the margin blanking. Therefore any software package attempting to access a particular pixel value must account for the awkward storage format when calculating the value's address. The camera interface board also controls the duration of the image storage, also called frame grabbing, process. Normally the imaging system will display the image stored in display memory until the system's microcomputer initiates a frame grab. A frame grab is triggered by a pulse on the PCS4 line from the microcomputer board. However, the actual storage does not begin until the next start of a video frame, which is signalled by the frame index pulse. When this occurs, the write line to the image memory is set low by the interface board and storage begins, lasting one fifteenth of a

second until the next frame index pulse. This frame grab control ensures that the stored image data represents one individual video frame in its entirety.

The computational power indigenous to the imaging system is provided by a Slicer Computers Inc. single board computer [Slicer Computers, 1983]. This computer is based on the Intel iAPX 80186 CPU. The board is equipped with 256 kilobytes of dynamic RAM. In addition, the board has two asynchronous serial ports, ROM with a monitor program, a floppy disk controller and a hard disk controller interface port. Asynchronous Port 1 communicates with the operator's terminal and Port 2 is used for sending data to and from a larger host computer.

The remaining elements of the imaging system are a Visual 500 terminal, two 8 inch floppy disk drives and a 256 kilobyte RAM extension board. The terminal has graphics capability which is effective for presenting graphics displays such as intensity histograms. The disk drives provide permanent storage for programming and image data. The extension memory provides space for temporary storage and processing of an image while not being displayed.

In total the Slicer computer has 768 kilobytes of RAM under its control. The on-board 256 kilobytes of RAM have the lowest address range, 00000H to 3FFFFH. Interrupt vector addresses, the Digital Research CP/M operating system, any

user programming and some of the non-displayed image processing scratch pad area reside in this memory section. The extension board's 256 kilobytes of RAM occupy the middle address range, 40000H to 7FFFFH. This section is used entirely for non-displayed image processing scratch pad memory. The display image memory has the highest RAM address range, 80000H to BFFFFH. When it is not being accessed by the Slicer computer or accessed for storing a grabbed frame, the contents of this section are continuously being displayed.

2.3 DISPLAY IMAGE FACILITIES

Two display image facilities of IMP do not have their own program section, but rather are handled within the master IMP program section. These are the single display image grab, caused by entering 'G', and the repetitive display image grab, toggled by entering 'R'. Both of these commands trigger the storage of the next camera video frame in the display memory by pulsing the Slicer PCS4 line. However, with the repetitive display image grab, the frame grab is triggered once every half second until another command is entered. This allows the operator to adjust the experimental setup and observe the results on the image display monitor.

The remaining display image facilities have their own program sections which are called from the master IMP program section. They operate on the display image memory and

do not affect an equalized image that may be stored in the mid-range memory. These routines are described in the following subsections.

2.3.1 Windowing

The routine WINDOW occupies the largest program section within IMP. WINDOW is not directly called by the master IMP section, but instead is called by some of the display image routines. WINDOW assists the operator in determining and specifying a rectangular subimage of the displayed image. On return from WINDOW, the calling routine is supplied with the subimage coordinates selected by the operator.

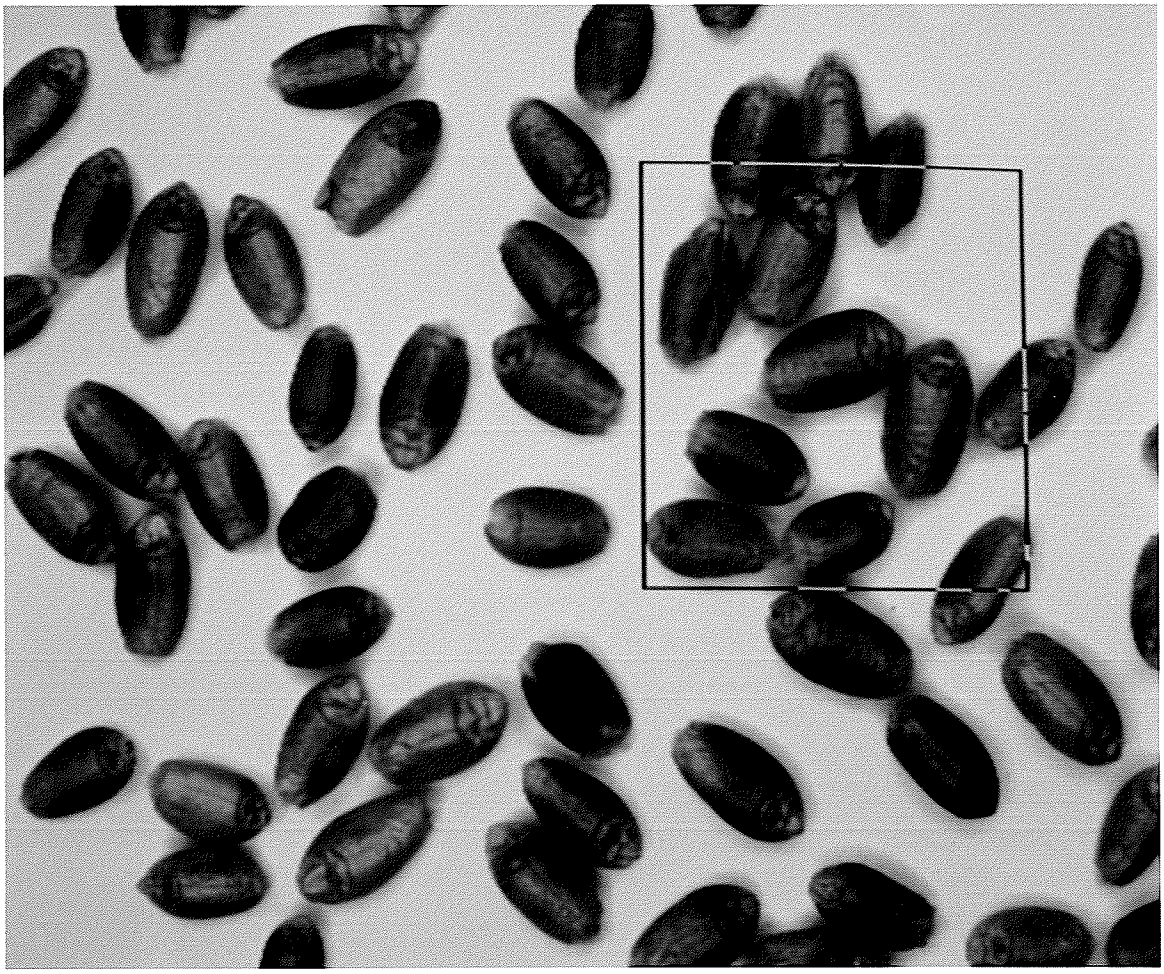
WINDOW draws a rectangular border around a subimage of the display image. Each pixel on the border is set to either black or white, whichever is most different from its previous value. The display image is not affected permanently by WINDOW since the old values of the border pixels are saved and then restored when the window is moved or removed.

Two modes are available for positioning the displayed window. In the "direct" entry mode, the window coordinates are directly entered by the operator. In the "adjust" entry mode, the position or size of a previously existing window can be adjusted. In this mode the numeric keypad keys are used to adjust the position and the cursor positioning keys are used to adjust the window size.

The subimage coordinates obtained by using WINDOW are not only used by the calling routine, but also saved in the IMP status block for use by other routines until the next frame grab. Thus a subimage that is specified during ZOOM can be HISTOGRAMmed and later ACQUIREd without having to respecify it with WINDOW.

An example image with WINDOW in use is shown in Figure 2.4.

Figure 2.4: WINDOWed Subimage



2.3.2 Intensity Calibration

The routine CALIBRATE generates the intensity calibration curve that can be used to calibrate a display image which is being equalized and transferred to the equalized image memory. This curve is represented by a 256 byte look-up table.

The use of calibrated intensity images ensures that the relationship between pixel value and surface reflectance is reproducible from image to image. In properly calibrated images, surface reflectance and the corresponding pixel value are linearly related. A pixel value of 0 represents a reflection coefficient of 0 which is absolute black, and a pixel value of 255 represents a reflection coefficient of 1 which is total diffuse reflection of incident light, in other words total white. Intensity calibration is particularly important for images from which gray level texture features are to be extracted.

Once the camera, lens and lighting are prepared for a calibrated imaging session, the operator performs an iterative procedure with CALIBRATE. First, an image of a gray step of a calibrated paper gray scale is grabbed. Then the gray step itself is WINDOWed by the operator and CALIBRATE determines the average pixel value of the subimage. Finally the operator enters the calibrated reflection coefficient of the gray step, as determined by the manufacturer. CALIBRATE multiplies the inputted value by 256 to yield the corresponding calibrated pixel value. In this way, several

points along the calibration curve are obtained. At the end of any iteration, the operator can have CALIBRATE display the current list of calibration curve coordinates or cause CALIBRATE to start the entire procedure over again.

After the operator is satisfied that a sufficient number of data pairs have been entered to specify the calibration curve, CALIBRATE generates the curve. The minimum number of pairs allowed by CALIBRATE is two since two points can define a line. The maximum number is arbitrarily set to fifteen. CALIBRATE first enters each data pair into the curve look-up table by setting the value of the byte representing the average pixel value to the corresponding calibrated pixel value. Then the curve is completed by linear interpolation between the data points and by linear extrapolation from the outer data points to both end limits of the curve. Before acknowledging that a successful curve has been generated, CALIBRATE ascertains that the curve generated represents a one-to-one strictly increasing function. If the generation process is satisfactory, CALIBRATE displays the resulting curve to the operator using the graphics capability of the Visual 500 terminal before returning to the IMP master program. A typical graphics display of a calibration curve is shown in Figure 2.5.

Following the generation of a calibration curve, the camera, lens and lighting setup are not changed until the end of the calibrated imaging session. To do so would invali-

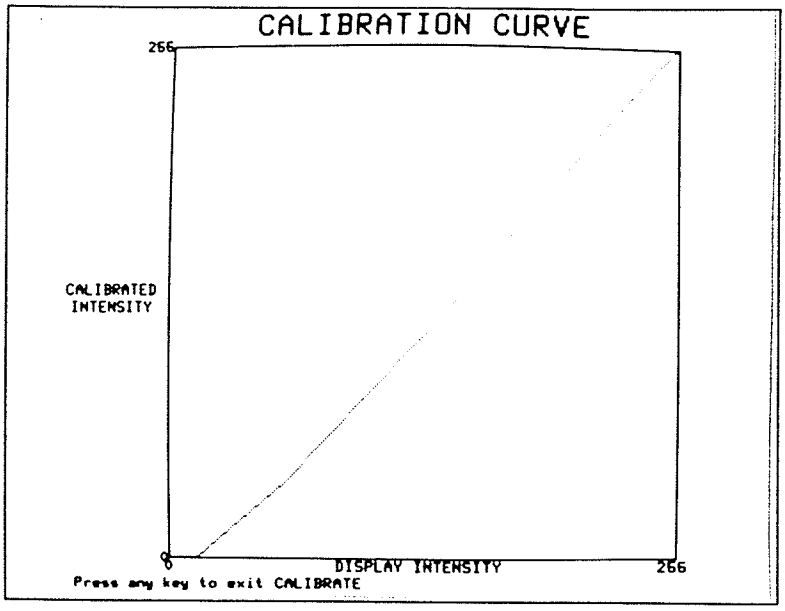


Figure 2.5: Typical Calibration Curve

date the calibration curve. During the session, the image equalization routine ACQUIRE will, when commanded by the operator, use the curve to intensity calibrate an image being moved into the equalized image memory.

2.3.3 Intensity Histogramming

The routine HISTOGRAM produces a histogram of the pixel values in the display image or of the pixel values in a rectangular subimage selected by using WINDOW. This histogram is displayed graphically on the Visual 500 terminal.

A pixel intensity histogram describes the pixel value distribution of an image. Knowledge of this distribution can assist an experimenter in estimating the effectiveness

of intensity thresholding for image segmentation. If the pixel values representing an object and the pixel values representing its background are tightly clustered about widely differing average values, then thresholding will be effective for segmentation. The histogram of a suitable image for thresholding will be distinctly bimodal. In addition, the minimum between the two histogram peaks will indicate the best threshold value.

The histogram plot produced by HISTOGRAM has a normalized vertical scale. The maximum value on the vertical frequency of occurrence axis is set to the maximum frequency of occurrence found by HISTOGRAM for any pixel value in the selected subimage. Thus the histogram bar for any pixel value will fit within the histogram plot. HISTOGRAM also displays the total pixels within the selected subimage and the maximum and minimum pixel values found. The histogram of the subimage shown in Figure 2.4 is shown in Figure 2.6.

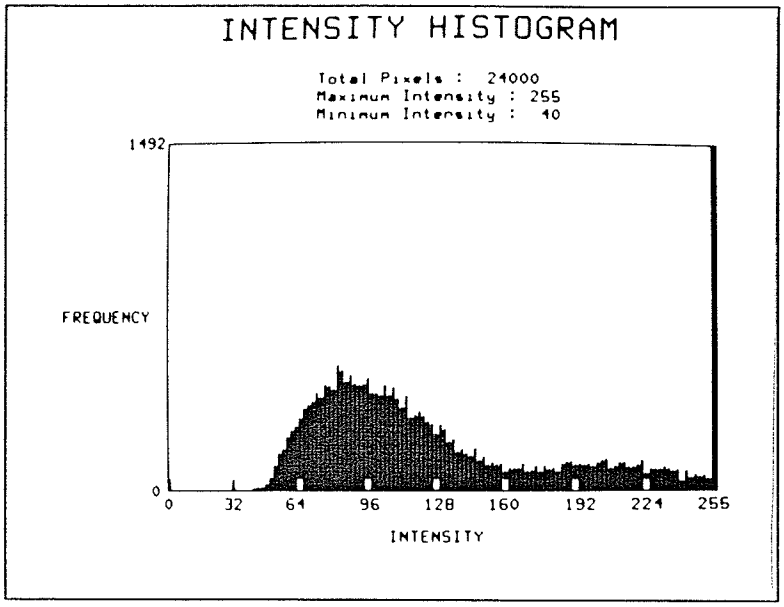


Figure 2.6: Histogram Display

2.3.4 Intensity Thresholding

Display images can be intensity thresholded by using the routine THRESHOLD. The result is a binary image which is placed in the display image memory. A binary image consists of totally white or totally black pixel values.

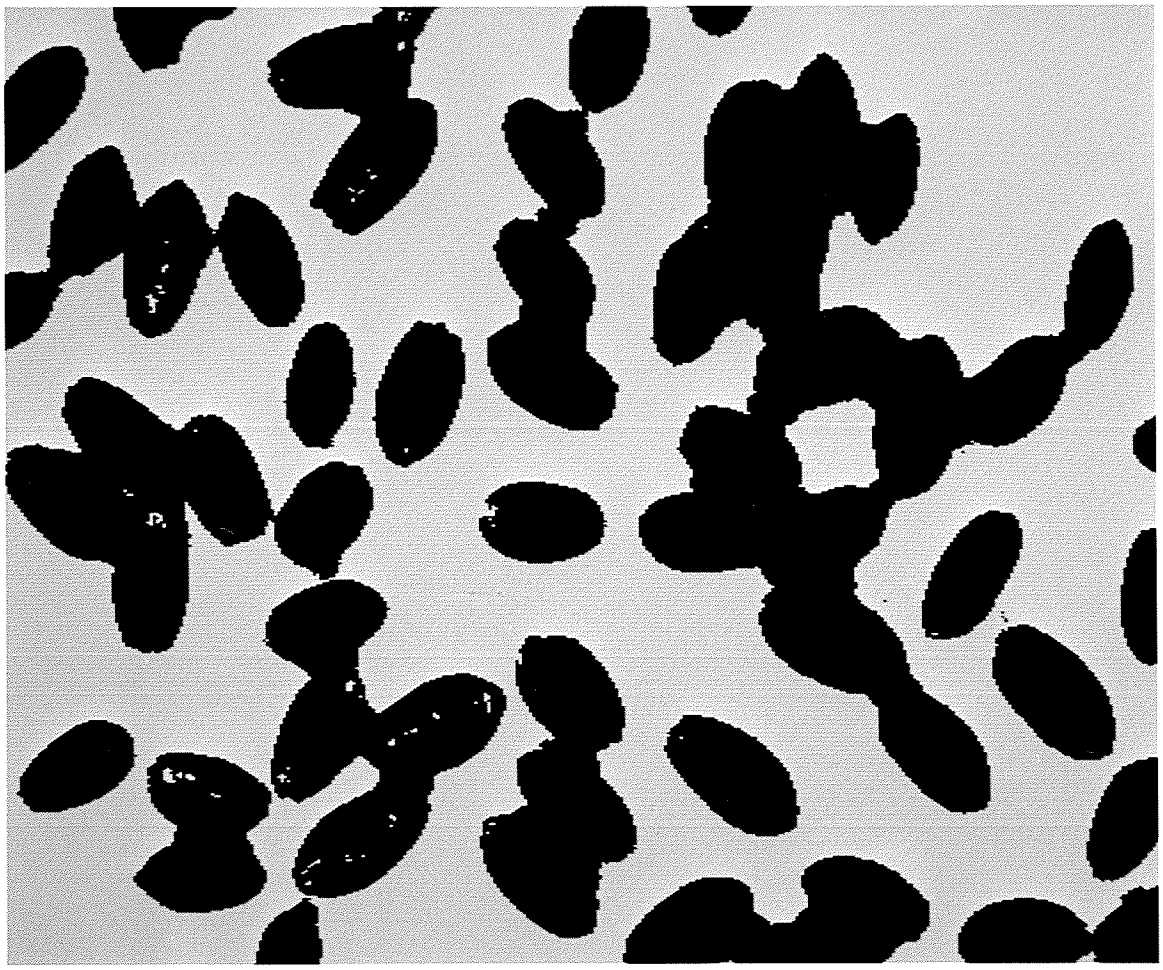
THRESHOLD is useful for testing the suitability of intensity thresholding for segmenting a particular image. Typically an experimenter would use HISTOGRAM to determine a suitable threshold value before using THRESHOLD to view the result of thresholding an image at the selected pixel value.

THRESHOLD stores an exact copy of the display image into equalized image memory beginning at 40000H provided that a previous equalized image will not be overwritten or the op-

erator has agreed to the equalized image's destruction. Then the operator can have THRESHOLD display binary versions of the stored image thresholded at any possible pixel value, in either positive or negative form. In addition, THRESHOLD can grab and store new versions for thresholding. When THRESHOLD is exitted, the stored display image is copied back into the display memory automatically.

Figure 2.7 shows the result of thresholding the image in Figure 2.4 at 160.

Figure 2.7: THRESHOLDed Image



2.3.5 Subimage Zoom

Enlarged views of rectangular subimages of the displayed image are produced by the routine ZOOM. This facility can be an important aid for the operator in determining whether a small subimage possesses a desired feature. An enlarged view is generated by expanding the subimage to fill the entire display. Each pixel in the original subimage is represented by a rectangle of pixels that have the same value as the represented pixel.

When ZOOM is called, the display image is immediately stored in equalized image memory, provided that a previous equalized image will not be overwritten or the operator has agreed to its destruction. The display image is stored in 384 byte segments, one for each image row, in non-interlaced format beginning at 52D00H. This memory organization allows an equalized image, that has at the most 192 rows, to reside in the equalized image memory while ZOOM is in operation. Then WINDOW is called so that the operator can specify the subimage which is to be enlarged. Once the window coordinates have been obtained, ZOOM first creates a row and a column look-up table. These look-up tables specify, for each pixel in the new zoomed display image, the row and column of the pixel in the stored old image which is to be represented. A row range and a column range of display pixels will represent one stored pixel so that a rectangle of display pixels will take the same value as one stored pixel.

Then an iterative procedure is performed until the display image is complete. This procedure maximizes the speed of execution. For each display row range corresponding to one stored subimage row, the corresponding display image row is stored in a data segment array. Then, since all rows in the display row range are the same, a version of the data array is copied to each row of the range. A trait of this method is that the zoom operation is faster for smaller subimages since they require fewer display image row ranges.

When ZOOM is exitted, the stored image is restored to the display image memory in exactly the same form as it was in before ZOOM was called.

Figure 2.8 shows the result of zooming the subimage shown in Figure 2.4.

Figure 2.8: ZOOMed Image

2.4 IMAGE CONVERSION FACILITIES

The primary purpose of the imaging system, of which IMP is a part, is to form digital images and after suitable preparation dispatch them to the external world for storage and/or processing. ACQUIRE, one of the two image conversion routines, performs the required image preparation. This preparation includes fashioning the desired image into the standard row/column format, equalizing it so that the row and column spatial densities are identical, and if necessary intensity calibrating its pixel values. The most important image preparation, however, is to extract, when necessary, only a rectangular subimage of the grabbed display image. By having a single extracted image already selected by the operator and prepared by ACQUIRE, the routines that send the image to the external world do not require their own image preparation facilities. Instead they are presented with a simple well defined image that is to be dealt with in its entirety and which is ready to send. This prepared image has its own storage area called the equalized image memory which is addressed from 34B00H to 7FFFFH.

The other image conversion routine is DISPLAY. It is the complement of ACQUIRE. DISPLAY copies a converted form of an equalized image in the equalized image memory to the display image memory. As a result the operator is provided with a displayed version of the image resident in the equalized image memory. This image would normally originate from

the outside world. The DISPLAY conversion consists of reversing the equalization process, interlacing the image and finally formatting it so that it will appear centred on the image monitor. DISPLAY does not harm the equalized image. Once called, DISPLAY functions automatically without requiring any further commands from the operator.

Equalization and the reverse process, hereafter called de-equalization, are the most computationally expensive processes involved in image conversion. As explained in section 2.2, the image detector array of the system's camera has a rectilinear lattice of photosites which are precisely spaced 18 microns apart vertically and 30 microns apart horizontally. The image produced has 482 rows by 377 columns of valid pixel values. The inequality in spatial pixel density is corrected by IMP since conventional digital images possess equal vertical and horizontal pixel densities. Many image processing programs on a host computer would expect this format. The equalization process creates an image that imitates the image that would result if the imaging array spacing were 18 microns by 18 microns. The maximum equalized image size then is 482 rows by 627 columns. It can be shown using two-dimensional sampling theory that since the vertical pixel density is not being changed, image equalization can be considered a one-dimensional problem wherein image rows are dealt with separately. The process adopted for equalization is linear interpolation. During equalization

groups of three contiguous display image row pixels are converted into five contiguous equalized image row pixels. The reverse occurs during de-equalization. If an equalized image is unmodified, the de-equalized process will create from it an image identical to the display image from which it originated, ignoring round-off error. Both processes are

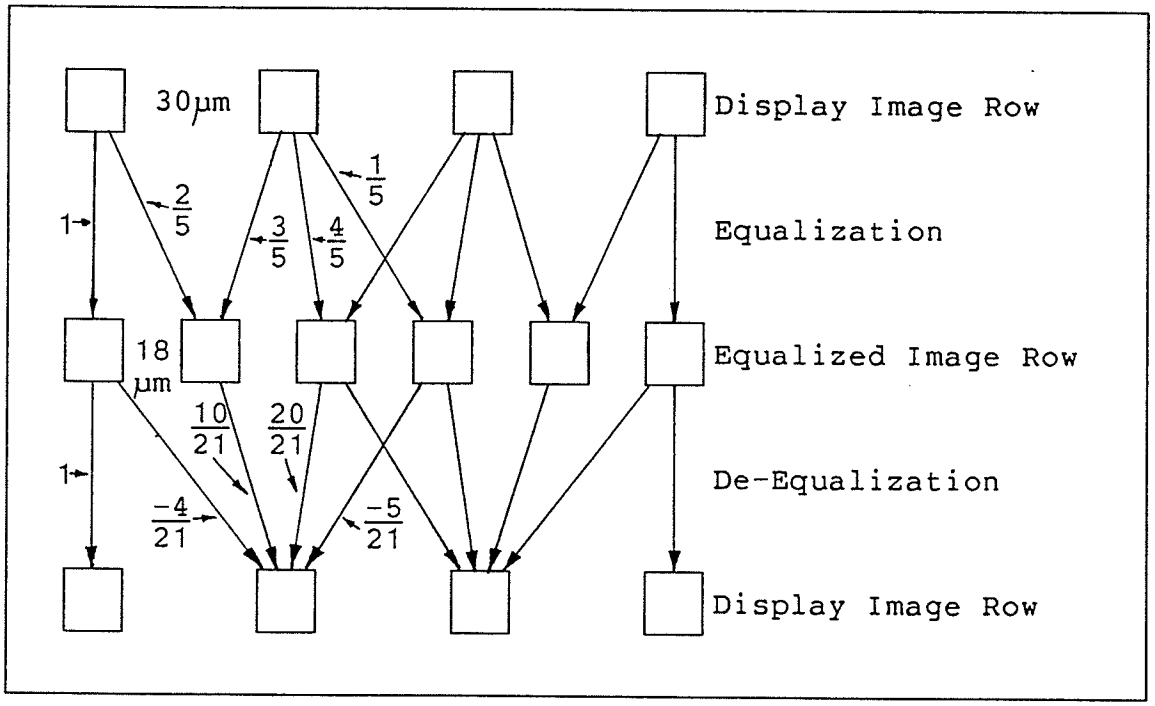


Figure 2.9: Equalization and De-equalization Process

shown graphically in Figure 2.9. Linear interpolation was chosen for two reasons. First it is much less computationally expensive than the exhaustive sampling theory solution. Second, the exhaustive sampling theory solution will cause

the formation of multiple vertical edges in the equalization of a display memory image containing only one vertical edge.

ACQUIRE gives the operator several options before acquisition begins. The entire grabbed display image can be ACQUIREd or either WINDOW can be called to specify a subimage to be ACQUIREd or the coordinates of a previous subimage, if available, can be used to specify the subimage to be ACQUIREd. If an intensity calibration curve is available and calibration is requested by the operator, ACQUIRE will translate each pixel value to its calibrated value during the ACQUIRE operation.

Figure 2.10 shows the result of DISPLAYing the ACQUIREd version of the subimage shown in Figure 2.4.

Figure 2.10: DISPLAYed Subimage

2.5 EQUALIZED IMAGE FACILITIES

The equalized image routines provide an interface between IMP and the external world. Their purpose is to send a prepared image in the equalized image memory to the external world for storage or processing and to accept a properly formatted image from the external world for temporary storage in the equalized image memory from which it can be moved to the display memory for inspection by the operator. The routine LINK handles transmission of equalized images to and from a host computer. In addition, LINK can connect the imaging system terminal to the host computer allowing the operator to communicate directly with the host computer. The routines STORE and RETRIEVE, as their names imply, handle the permanent storage and retrieval of equalized images on floppy disk. The equalized image routines do not affect the display image memory contents. These routines are described in the following subsections.

2.5.1 Data Linking to a Host Computer

The routine LINK performs all of the functions of data linking IMP to a host computer. These functions are image transmission to the host computer, image reception from the host computer and a transparent mode in which the imaging system's terminal is virtually connected to the host computer. When called, LINK initially enters the connect mode. While in connect mode LINK relays all characters from the host computer to the terminal and relays all characters from

the terminal console, except three control characters, to the host computer. These control characters are Ctrl-P, Ctrl-R and Ctrl-T. Ctrl-T initiates image transmission, provided an image is resident in the equalized image memory. Ctrl-R prepares LINK for the reception of image data. Ctrl-P allows the operator to terminate the LINK session.

The data channel for which LINK is designed is an asynchronous duplex serial line which employs the XON/XOFF protocol for data flow control and ASCII for information coding. This type of channel is slow for the massive amounts of data needed to represent most images. Many sophisticated imaging systems use a high-speed synchronous parallel data channel for rapid image transfer. However several reasons motivated the use of the slower channel for image transfer with LINK. First, special hardware is normally not required for an asynchronous serial line since most computers are already equipped with this type of line for connecting ports to terminals. Consequently, the installation of this type of communication line is simple and inexpensive. Additionally, the 9600 baud Data General port line used in this research typically permits the transfer of a 200 row by 200 column image in 45 seconds. This is not an unreasonable time requirement for transferring an image having the standard size used in this research. Finally, by connecting LINK to the port line of a host computer, only one communications line is required for the operator both to interactively command

the host computer and to transfer image data. Imaging systems that employ a high-speed data line normally require a separate slower terminal communication line that allows the operator to command the host computer.

LINK does not quite occupy the largest program section in IMP, yet it is the most complicated of the IMP routines. LINK's complexity is a result of its unusual structure. LINK consists of a set of asynchronously executed program units which are continuously polled in a cyclical fashion. Some units are always executed once every cycle while others are included in the polling cycle only during certain operations. In essence, these units exchange the information, which LINK is transferring, amongst themselves and the outside world. Hence an effective way of visualizing the operation of these units is as an information circuit. The LINK information circuit is shown in Figure 2.11 on the next page.

The LINK information circuit contains five entities which both consume and produce information. These are the terminal, the host computer, the receive and transmit character buffers and the equalized image memory. The equalized image memory is implied but is not shown in Figure 2.11. The 16 byte character buffers are necessary because of the XON/XOFF protocol. Without buffering, the delay caused by converting from serial ASCII to an internal byte representation and back to serial ASCII during the relay of information between

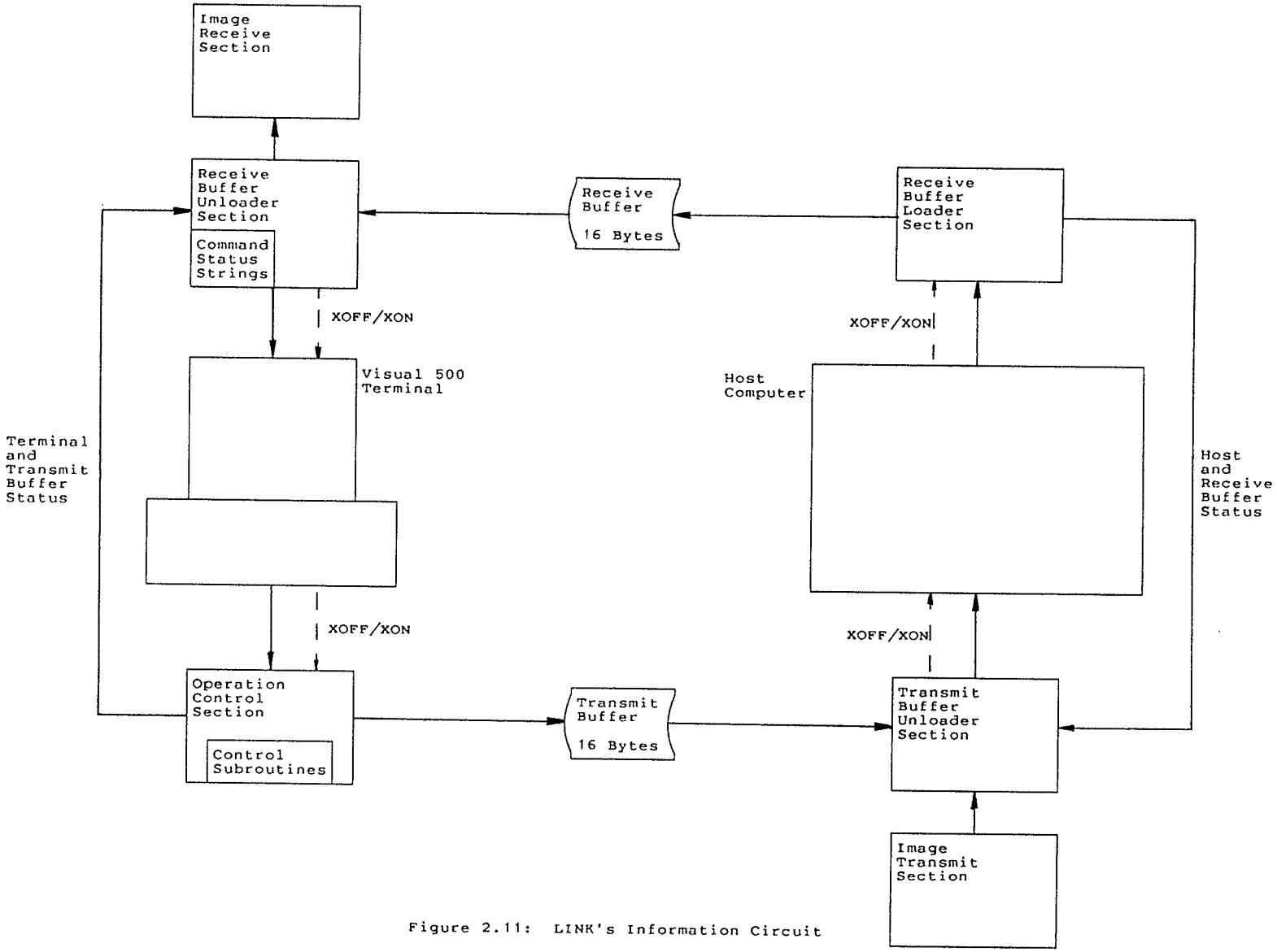


Figure 2.11: LINK's Information Circuit

the terminal and the host computer can allow the input buffer of either the terminal or host computer to overflow before the sender is signalled to wait. Each character buffer used by LINK possesses its own input data flow control enabling it to have XON/XOFF control over the entity sending to it. The terminal and host computer appear to LINK as I/O ports since the Slicer interfaces to the serial asynchronous lines of each of these entities by using its on-board Signetics SCN2681 Dual Universal Asynchronous Receiver/Transmitter (DUART). Channel A of the DUART communicates with the terminal at 9600 baud and Channel B of the DUART communicates with the host computer.

The connections between the above entities are provided by six independent program sections. These sections communicate to each other by setting status flags, incrementing pointers and changing subroutine start addresses. The operation control section, which is subdivided into five subroutines, allows the operator to control LINK. This section receives all characters entered at the terminal console by the operator. During connect mode the received characters are copied into the transmit buffer unless one of the three control characters happens to be intercepted. If a control character is detected, the operation control section changes LINK's mode of operation to either receive mode, transmit mode or control mode, depending on which control character was entered. The control mode is not readily apparent to

the operator. It is entered when the operator has issued a questionable command or at the end of transmit or receive mode. In command mode all other processing stops while LINK prompts the operator for a further entry. The transmit buffer unloader section, the receive buffer loader section and the receive buffer unloader section perform the operation implied by each of their names. The receive buffer unloader section has the additional task of sending command status information character strings to the terminal. These character strings are kept in a data block and are referenced by a character pointer and a next string pointer. They enable LINK to prompt the operator during command mode or inform the operator of an image transfer operation's status during transmit or receive mode. All four of the above program sections take part in the XON/XOFF protocol by determining a buffer's status when loading or unloading it and by transmitting XON or XOFF to a sender when required. The last two program sections are the image transmit section and the image receive section. Each of these sections performs the function suggested by its name. During transmit and receive mode, the operator is continuously informed of the current number of pixel values transferred. During image reception a tally of character, parity, missing row and row size errors is kept. This tally is presented to the operator at the end of reception.

LINK transfers image data using an elementary code intended for use on ASCII communication channels. The use of this code is necessary to avoid the unintentional transmission of control characters and to avoid the use of the parity bit. A raw pixel value is represented by a byte or eight bits. However in ASCII the eighth bit is only used for parity checking and therefore it should not be used when transmitting data. In addition, the lowest 32 characters of the 128 character ASCII code are control characters and their unintentional use on some host computers can produce unexpected results. Consequently only 96 possible characters can be safely transmitted as data using ASCII. The image transmission code uses 3 of these characters, <!>, <"> and <#>, to specify the range of the succeeding pixel values to be 0 to 84, 85 to 169 or 170 to 255 respectively. These are called the pixel base characters. Once a value range is set by a pixel base character, each of the succeeding contiguous pixel values in the current row that falls within that range is transmitted using a pixel offset character. Pixel offset characters are the 86 characters between <$> and <y> inclusive. Each corresponds to an offset value between 0 and 85 which specifies the pixel value by its offset within the range. When a pixel value is encountered which lies in a different range, a new pixel base character is transmitted before continuing. Images are transmitted in row major form. Each row begins with a pixel base character and ends with one <|> character. The image is terminated with two

<|> characters. Each record of transmitted image data is 79 characters long, the first being a blank. The first record contains the image row and column sizes as decimal numerals. For most images, this algorithm can represent an image by a total number of character transmissions which is only slightly greater than the number of pixels making up the image.

The present version of LINK is designed for connection to either a Data General or a DEC 9600 baud port line. LINK prepares for this environment by first querying the operator for the type of host computer and then sending the appropriate setup control data when it begins operation. One control string is sent to the Visual 500 terminal placing it in either the D. G. Dasher 200 or the DEC VT52 terminal emulation mode. Other control information is sent to the DUART to set up its Channel B interfacing with the port line. The receive buffer unloader section sends command status control strings appropriate for the current terminal emulation mode. These are obtained from a control string table. If LINK is required in future to communicate with a different host computer on a different serial asynchronous communication line, LINK can be reconfigured simply by changing the setup control data and adding the correct control strings to the control string table.

2.5.2 Storing and Retrieving Images on Floppy Disk

Equalized images are stored and retrieved on floppy disk on the imaging system's two 8 inch floppy disk drives by using the routines STORE and RETRIEVE respectively. Both routines utilize the BIOS disk operating functions of the CP/M operating system.

The routines STORE and RETRIEVE allow the operator to permanently archive essential images for future use. This facility is particularly attractive when the host computer has insufficient storage space for the requirements of the operator. If this is the case, the operator can store both unmodified equalized images and images that were the result of processing on the host computer and subsequently transferred to the imaging system using LINK. Approximately 16 images of the standard size used during this research, 200 rows by 200 columns, can be stored on one two sided double density floppy disk.

Both STORE and RETRIEVE perform rigorous checks on the validity of the operator's instructions. STORE verifies that the disk drive specified has been logged-in by CP/M, that the drive is not in R/O (read only) mode and that the specified file name is legitimate and not already present on the disk. STORE automatically appends ".IMG" to the specified file name so that image files are readily distinguishable. STORE also tests for directory and data area overflow each time it writes a record thereby ascertaining that the

disk has sufficient storage space for the image file. If overflow does occur, STORE erases the new image file and aborts. RETRIEVE verifies that the specified disk drive is logged-in and that the specified image file does exist on the disk. RETRIEVE also automatically appends ".IMG" to the specified file name.

Each time the CP/M disk operating system functions read data from or write data to disk, an individual record of 128 bytes is transferred. This record is exchanged between CP/M and a calling routine through the use of a 128 byte RAM memory array called the DMA. The DMA is part of the File Control Block which specifies the required attributes of a disk access. The first 128 byte record stored in an image file contains two two byte numbers which specify the row and column size of the image and an 80 byte array which may contain a one line description of the image. If the operator chooses to describe the image during storage, STORE will place the line of description in the 80 byte array. During retrieval of the image, RETRIEVE displays this line of description to the operator. The remaining records in an image file consist of image data. An image is stored in row major form. Both STORE and RETRIEVE manipulate these records in sequential access mode. Most of the computation performed by both these routines involves formatting the image data transferred to or from the DMA so that pixels are handled contiguously without overwriting or interspersed gaps.

Chapter III

OBJECT PERCEPTION

3.1 INTRODUCTION

Once an image of a grain sample has been suitably formed through imaging, digitization and storage, the single most important task for the computerized image analysis system is the perception of all pertinent objects appearing in the image. The initial object perception operation must provide data which directs subsequent image analysis only on regions of the image which are of interest. Logically the object perception operation should be the first image analysis performed. In this research the object perception routine was invoked before any further analysis could begin on an image.

Object perception here refers to the detection of an object appearing in an image and the exact specification of its location and contour. Although this is a rather limited form of perception, it forms the basis for the primitive image understanding which the machine must possess.

In this research an object's position and contour of length ℓ were specified with a version of the Freeman chain code, $((r_0,c_0),a_0,a_1, \ldots ,a_{\ell-1})$. The contour was traced sequentially from the point (r_0,c_0) in single steps, the di-

rection of each being specified by a directional element, a_n, of the code as shown in Figure 3.1. r_0 was the row and

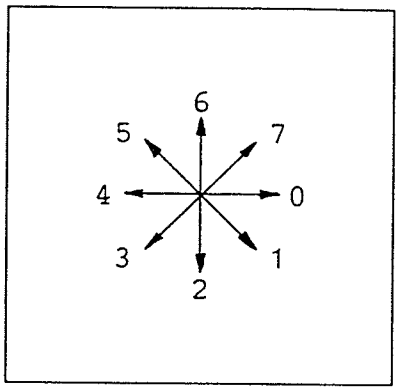


Figure 3.1: Elements of the Chain Code

c_0 the column coordinate of this point. The ℓth step returned the trace to the point (r_0,c_0). This code allowed contours to be 8-connected. That is, two contiguous pixels on a contour could be vertical, horizontal or diagonal neighbors.

The objects that were to be perceived in this research were usually kernels of wheat although other objects, such as cereal kernels other than wheat and foreign material other than cereal grains, could sometimes be expected. It was safely assumed that at least each object would be similar in size to a wheat kernel since, as described in Subsection 1.3.2, grain samples are partially cleaned before being analyzed. The object perception algorithm used would have to ignore grossly differently sized objects such as specks of

dust or finger print smudges on the background. The object perception algorithm would also have to avoid classifying a part of an object, such as a wheat germ, or a group of separate objects as being one individual object. In addition to the limited size range, another characteristic of the objects to be imaged further simplified the object perception problem. The objects would only appear in a single layer on a flat white background. Therefore, as stated in Section 1.1, objects would not occlude one another and would only sometimes touch. With this constraint object perception was effectively a two-dimensional problem without the attendant complexities of three-dimensional scene analysis.

The object perception problem in this research was basically one of object-background image segmentation; an image pixel had to be classified as being either an element of an object region or an element of a nonobject region, the background. Many techniques are available for such simple image segmentation. These utilize a variety of approaches which typically depend on the regularity of the image characteristics. A good example is template matching. However the images encountered in this research exhibited many irregularities. The surface gray level and the shape of a wheat kernel were always unpredictable. The gray level of the background also had a certain amount of unpredictability due to the effects of shadow. As a result most simple segmentation techniques were inappropriate.

The object perception techniques, which were developed, relied on the one simple and regular characteristic of the wheat grading images used in this research: the object contour edge was always the most dominant type of edge. Consequently every technique employed a step edge detector at some stage of its operation to find object contour edges. Each of these edge detectors estimated the gradient of the gray level surface in order to gauge the probability of an edge at any particular location.

All of the computerized image analysis, other than the elementary image preparation performed using IMP, was executed on a Data General Eclipse MV/8000 Model 9300 mainframe computer. This image analysis included the object perception procedures described in this chapter and the procedures described in Chapter IV and Chapter V. Every routine was implemented in FORTRAN77 [Data General, 1983]. The Data General computer was the host computer referred to in Chapter II as being connected to the digital image acquisition system.

Two images were used in this chapter as subjects for demonstrating the capabilities of the object perception techniques. These images, JUMBLE1 and JUMBLE2, are shown in Figure 3.2 and Figure 3.3 respectively beginning on page 72. Both of these images were intended to present a substantial challenge to any object perception algorithm while at the same time remaining within the limitations of this research.

JUMBLE1 was the most formidable of the two images because it exhibited a wider range in wheat kernel sizes than normally encountered and because the contact point between two of the touching wheat kernels in JUMBLE1 was totally obscured by shadow.

Two methods were used to present images in this chapter and in Chapter IV and Chapter V. Unprocessed images, such as JUMBLE1 and JUMBLE2, were photographed directly from the digital imaging system's image display monitor and displayed on 8½ inch by 11 inch prints. The digital images shown in Chapter II were also prepared this way. Representations of processed images that were the result of some form of analysis, such as edge detection, were produced using the Symbolics laser printer [Symbolics, 1982]. This printer was a peripheral device of the Data General computer. With this method an image pixel was represented by a 5 by 5 square of dots, each of which could be either "on", producing a black dot on the paper, or "off", leaving a vacant white dot on the paper. Thus 26 gray levels could be represented with this method without the use of any intermediate photographic processing. Unfortunately the laser printer deposited ink with slight inconsistencies which caused the obliteration of subtle detail such as that seen on the surface of a wheat kernel. Consequently image representations of unprocessed digital images were not satisfactory when produced with this method.

The following four sections, 3.2 to 3.5, present four different object perception techniques in the chronological order in which they were developed during this research. In effect the first three techniques represent the evolution of the final superior technique, called the elliptical-object detector. However each of these three can itself satisfactorily perform the object perception function provided the limitations of the technique are recognized. The elliptical-object detector is the technique endorsed by this research. Much of its underlying theory is presented in Section 3.3.

Section 3.6 presents the shape description technique that was employed to generate shape features of an object contour provided by an object perception technique.

Figure 3.2: JUMBLE1

Figure 3.3: JUMBLE2

3.2 CLOSED-REGION DETECTOR

The first object perception technique developed in this research utilized very little a priori knowledge of the images to be expected. As explained in Section 3.1, a gradient based step edge detector was used to find possible locations of the object-background border. The operation of edge detection however only created an "intrinsic" image which, along with noise caused by the presence of secondary edges that were not of interest, indicated possible contour locations. Before the leap to the "segmented" image could be made, however, some technique based on the attributes of the imaged scene had to be used to discard the false edges and "string together" the real edge segments. With the formation of a segmented image in this fashion, the machine would, with the simple object-background scenes encountered in this research, be able to declare objects found and thereby accomplish basic object perception.

The technique used to find object contours in an edge image relied on a simple characteristic of object-background images. If the objects were not touching and the background was entirely homogeneous (free of edges) then an object contour would be a simple closed curve that would contain any edges present on the object's surface. Thus only an edge which formed a simple closed curve and which was not contained by another such closed edge would be declared to be an object contour. This technique was implemented in the

program CLOSED. This program inputted a binary edge image and outputted a binary region image and a file of Freeman chain codes describing each detected object. In the binary edge image a pixel value of 0 indicated no edge and one of 255 indicated that an edge was detected. In the binary region image a pixel value of 0 indicated background and one of 255 indicated an element of a declared object region.

The edge detector chosen was the Haralick [1982] zero crossing of second directional derivative edge operator. This operator is one of several advanced edge operators that represent a vast improvement, in terms of accuracy in edge definition and performance in noisy images, over the classical gradient based edge operators such as the Roberts, Kirsch and Sobel operators.

The classical edge operators perform poorly in noisy images for two reasons. First, each operator uses only a small window of image pixels in estimating the local gradient. Second, each operator finds edges by merely thresholding the gradient magnitude: when the magnitude is greater than a preset threshold an edge is declared: otherwise no edge is declared. This method gives rise to a wide declared edge that poorly defines the location of what was a well defined edge in the original image, and to a missed edge where an obscure edge existed in the original image.

For noisy images, the common solution is to preaverage the original image before applying the gradient operator. This approach can alleviate the noise problem so that noisy edges are not overlooked and spurious edges are not generated, but only at the expense of further widening the declared edges. In the images used in this research, the object contour edge was often obscured by detail on the object surface and shadow on the background. A more forgiving edge detector had to be used.

The Haralick edge operator combines two concepts to improve its performance. First, for an edge pixel to be declared not only must the local gradient magnitude be above some threshold value, but also a zero crossing of the second directional derivative must occur nearby. The latter criterion ascertains that the gradient has reached a local maximum in the pixel's vicinity. Thus the declared edge will be only one or two pixels thick and will provide a more accurate estimate of the real edge's location. The effectiveness of this approach is illustrated in Figure 3.4 where a one dimensional image of a step edge was used.

Haralick's edge operator is similar to the Marr and Hildreth [1980] Laplacian of a Gaussian edge operator since both operators use a form of the second derivative. The Marr-Hildreth edge operator is well-known for its accurate edge detection, although it suffers from susceptibility to noise. Before an edge pixel can be declared, the Marr-Hil-

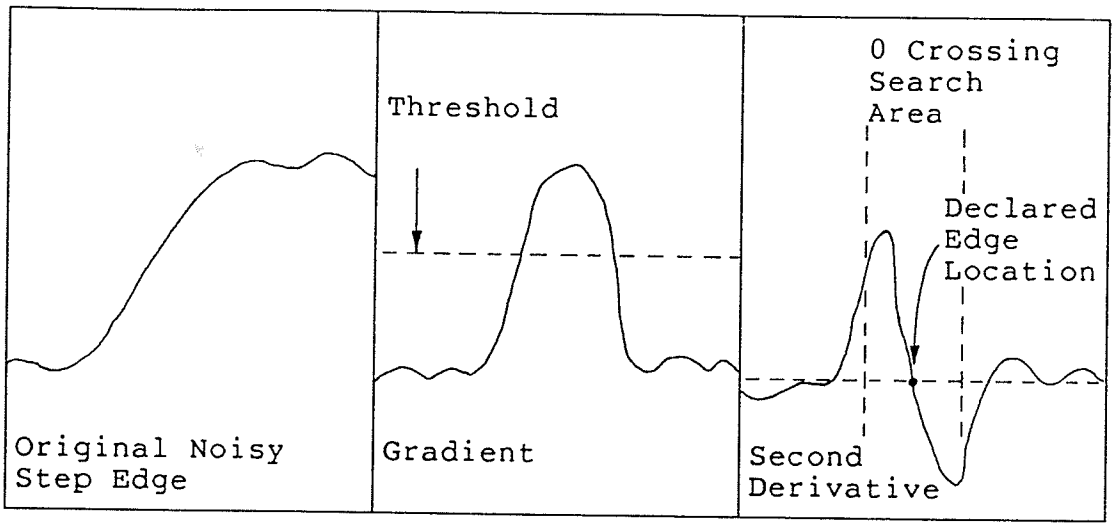


Figure 3.4: Zero Crossing at a Step Edge

dreth edge operator requires that a zero crossing of the Laplacian, a type of second derivative, must occur nearby. Since the Laplacian is a nondirectional operator, the Marr-Hildreth edge operator is not sensitive to direction. However, Haralick's edge operator requires that a zero crossing of the component of the second directional derivative that is in the same direction as the local gradient must occur nearby. Consequently the Haralick edge operator is sensitive to direction and therefore can more effectively differentiate between spurious and genuine edges.

The second concept that gives the Haralick edge operator its improved performance is the assumption that an image is a noisy version of some underlying mathematical model. Haralick assumes a two-dimensional cubic model. This assumption does not by itself improve performance. Rather, it has

provoked the use of an estimate of the underlying model for each square neighborhood of pixels. Thus an analytical model is generated which can be directly manipulated to find the gradient at and second directional derivative near the centre pixel. This model tends to ignore noise in a fashion similar to preaveraging. Yet it will not cause the smearing of an authentic edge to the extent of that caused by preaveraging. Thus it provides more accuracy in the definition of an edge.

The method of estimating the two-dimensional cubic model, which underlied each pixel neighborhood, was altered for the closed-region detector. Haralick employs a nine member orthogonal polynomial basis set to estimate the model. In this research a least squares estimate of the model was calculated directly. A pixel value in neighborhood row r and neighborhood column c was modelled as:

$$f(r,c) = k_1+k_2r+k_3c+k_4r^2+k_5rc+k_6c^2+k_7r^3+k_8r^2c+k_9rc^2+k_{10}c^3 \ .$$

Neighborhoods were square and had an odd number of rows and columns so that a single centre pixel was definable. This centre pixel occupied the origin position where $r = 0$ and $c = 0$. For an M by M square neighborhood the equations which estimated the values of the constituent pixels could have been written in matrix form as:

$$M^2 \begin{bmatrix} 1 & r & c & r^2 & rc & c^2 & r^3 & r^2c & rc^2 & c^3 \\ & & & & \text{------} & & & & & \\ & & & & \text{------} & & & & & \\ & & & & \text{------} & & & & & \\ & & & & \text{------} & & & & & \end{bmatrix} \begin{bmatrix} k_1 \\ k_2 \\ \cdot \\ k_{10} \end{bmatrix} = \begin{bmatrix} f(r,c) \\ \cdot \\ \cdot \\ \cdot \\ \cdot \end{bmatrix}$$

10

or: $$\mathbf{AX} = \mathbf{B}.$$

M^2 was larger than 10. Therefore the least squares solution to this problem, such that $||\mathbf{AX}-\mathbf{B}||^2$ is minimized, is given by:

$$\mathbf{X} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{B}$$

$$= \mathbf{A}^+\mathbf{B}$$

where $\mathbf{A}^+$ is the Moore-Penrose inverse of **A**. Evidently each estimated model coefficient k_i was a linear combination of the pixel values in the neighborhood, or in other words each coefficient was the dot product of the neighborhood and an M by M mask. The mask values for each coefficient were calculated once and later permanently incorporated into the program which executed the Haralick edge operator.

Another modification was made to the Haralick edge operator to allow several different pixel neighborhood sizes. Haralick advocates the use of an 11 by 11 window. In this research the neighborhood sizes used also included 5 by 5, 7 by 7 and 9 by 9. The smaller sizes, 5 by 5 and 7 by 7, were found to be a better compromise between noise rejection

and the ability to accurately define a sharply curved edge, such as the pointed end of a wheat kernel. In addition the smaller sizes required less computation. The 7 by 7 size was standard.

The Haralick edge operator was implemented with the previously described modifications in the program ZEROX. At each pixel location the coefficients k_2 and k_3 were calculated. Then the estimated gradient at that pixel's location, which was the centre of the neighborhood, was found using the equation:

$$|\nabla f(0,0)| = \sqrt{k_2{}^2+k_3{}^2} \ .$$

If this estimated gradient was above a previously entered threshold then processing would continue on this pixel. First, the remaining coefficients k_4 to k_{10} were calculated. These were then used to calculate the distance in the "chessboard" metric to the nearest zero crossing of the second directional derivative. Only the component of the second directional derivative in the same direction as the gradient was considered. The distance was determined by the expression:

$$d\ (\min) = \frac{|F|}{|E|+|D|}$$

where $D = k_2{}^2k_8+2k_2k_3k_9+3k_3{}^2k_{10}$

$E = 3k_2{}^2k_7+2k_2k_3k_8+k_3{}^2k_9$

$F = k_2{}^2k_4+k_2k_3k_5+k_3{}^2k_6$

This expression was relatively simple because the locus of the zero crossing was the line:

$$Dc+Er = -F \ .$$

If the pixel passed the gradient threshold test and a zero crossing was within a set distance of the pixel, usually 0.5 of the interpixel distance, then the pixel would be declared to be an edge pixel.

Before a binary edge image outputted by ZEROX was analyzed by CLOSED for the presence of object regions, it was first transformed into another binary image. This new binary image was a "thin" version of the edge image, in other words a line drawing of it. Thinning of the edge image was necessary because by definition in the continuous plane a contour, which may define a region with thickness, should not by itself have thickness. In the discrete plane the minimum line thickness is 1. Therefore all contours in a binary edge image should have a thickness of 1. The thinning algorithm chosen was developed by Pavlidis [1981] and implemented in the program THIN. Other than thinning, this algorithm did not destroy the integrity of the original binary image since any region of 1's in the input image that was a connected set would be left connected by the algorithm. Valid edges found and declared by ZEROX were not greatly affected by THIN since these edges were usually only 1 or 2 pixels thick.

In summary, the first object perception technique developed in this research, the closed-region detector, consisted of three steps. First, the Haralick edge operator was employed in the program ZEROX to create an intrinsic binary edge image from an unprocessed wheat grading image. Second, the edge image was thinned using the Pavlidis thinning algorithm implemented in the program THIN. Finally, all closed and uncontained contours in the thinned image were found and specified in Freeman chain code by the program CLOSED. The last step also produced a segmented or region image. With the specification of object regions provided in the last step, a shaky form of primitive machine image understanding was achieved.

The following three figures show the intermediate and final results of the closed-region detector operating on JUMBLE2. Figure 3.5 shows the edge image produced by ZEROX. A gradient threshold of 5, one-half of the standard value of 10, was chosen. This lower threshold was selected since more sensitivity was desired for detecting the edges obscured by shadows. These excessive shadows were a result of the close object spacing. The standard neighborhood size, 7 by 7, and the standard minimum zero crossing distance, 0.5, were used. Figure 3.6 shows the thinned edge image produced by THIN. Finally, Figure 3.7 shows the region image produced by CLOSED. The solid black connected regions were "objects" declared by CLOSED. The Freeman chain code of each region's contour was stored in the output file.

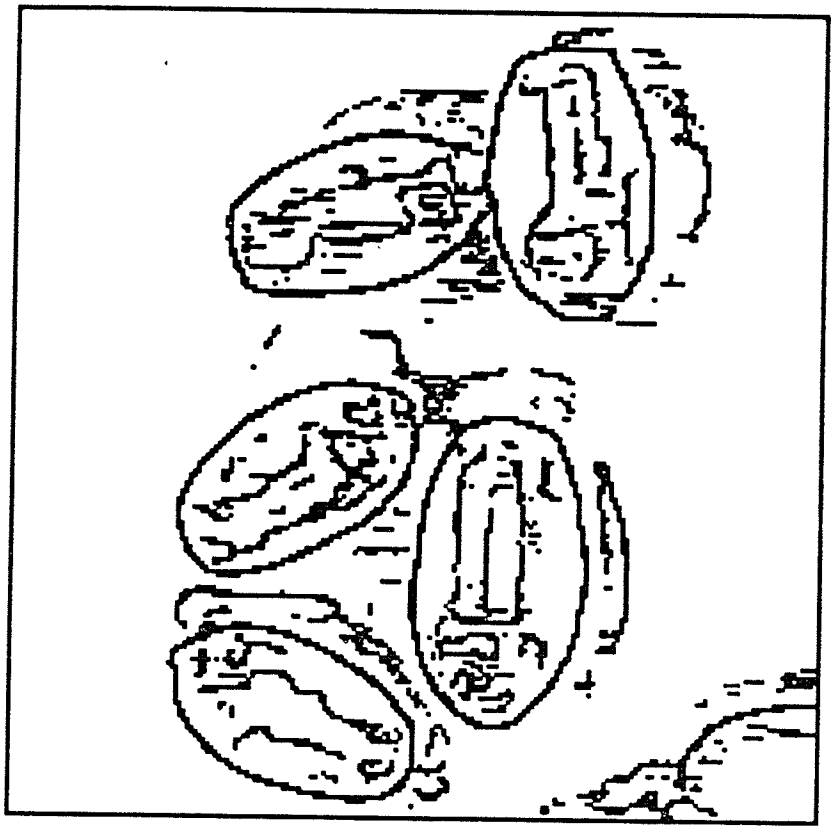


Figure 3.5: Edge Image from ZEROX

The previous figures demonstrate some shortcomings of the closed-region detector. Although the Haralick edge operator produced superb results, it sometimes would leave a small break in an object's contour thus voiding the object's detection. In scenes of closely spaced objects it was impossible to set a gradient threshold that would allow both the detection of object contour edges which were partially obscured by shadow and the rejection of spurious edges which were caused by regions of shadow on the background. Scenes of touching objects just could not be dealt with. In short, this detector was unreliable, requiring human intervention

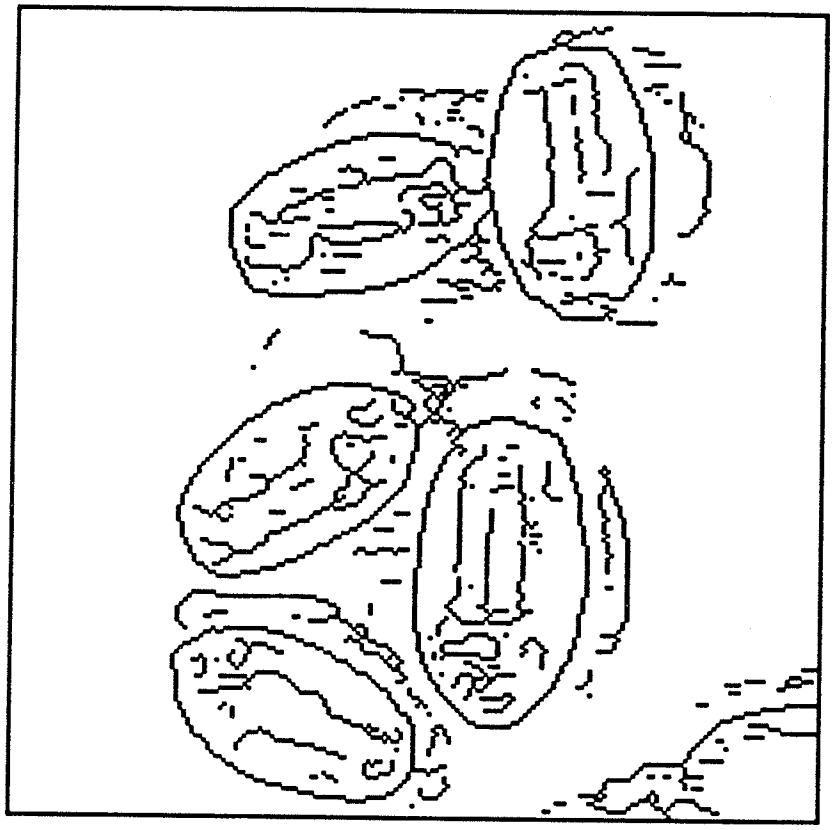


Figure 3.6: Thinned Edge Image from THIN

to ensure that only real objects had been found, and restricted to use on well behaved images containing widely spaced objects with sharply defined contours.

The previous figures do not demonstrate a less serious shortcoming of the closed-region detector. The Haralick edge operator demanded massive amounts of computation to produce the superb results essential for the closed-region detector. On the standard 200 by 200 image used in this research, applying this edge operator with the standard neighborhood size of 7 by 7 required an absolute minimum of 3.92

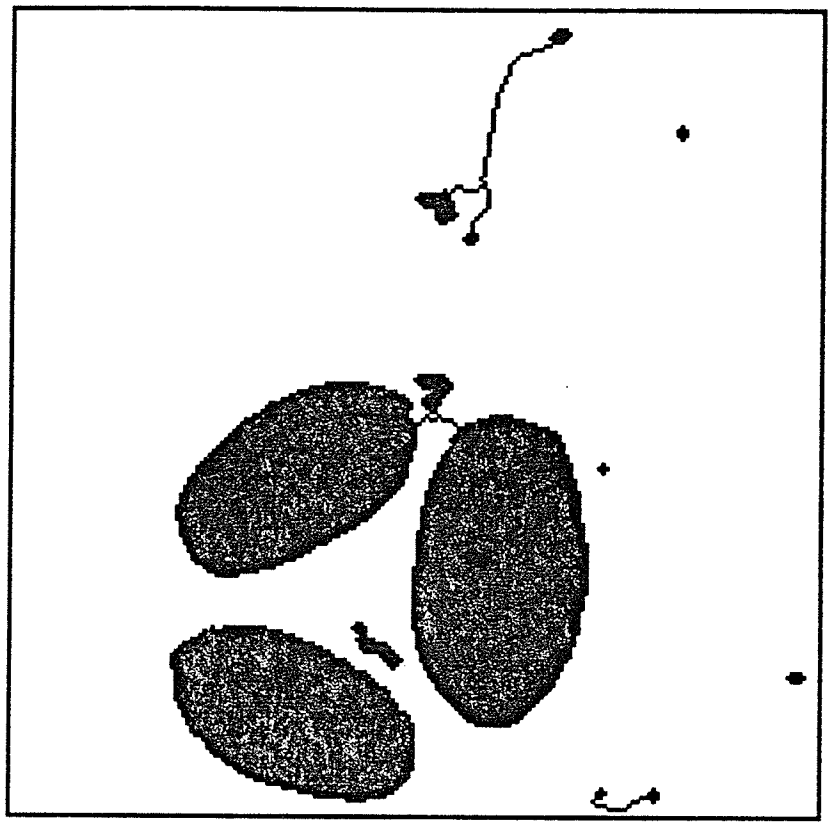


Figure 3.7: Region Image from CLOSED

million multiplications alone. Even for the flexible speed of execution constraints for research, this was an undesirably large amount of computation. Unfortunately most of this computation was a waste since many edge pixels declared by ZEROX would later be discarded as noise.

For the automated grading process pursued in this research a more dynamic and self-reliant object perception technique was required. Certainly the closed-region detector could have been improved. The routine CLOSED could have been modified to ignore closed regions that were below a

preset size or that had parts with a thickness of only 1. These modifications would have assisted CLOSED in discriminating against false edges caused by shadow. Further, CLOSED could have been modified so that an attempt would have been made to complete, rather than ignore, broken contours which satisfied certain size and shape constraints. All of these alterations would have been an attempt to introduce additional image contextual information and a priori knowledge about grain grading images into the object perception algorithm. However, each would have involved making improvements based on the contents of only an intrinsic image, the thinned edge image. This image would have lacked much of the original image information, notably gradient magnitude and direction, important in determining image context. Consequently two powerful and well-known techniques were combined to create a new object perception algorithm which used contextual image information and a priori knowledge to detect objects while utilizing the original image directly. Several versions of this more successful approach were developed, the first of which is described in the next section.

3.3 REGULAR-SIZED-OBJECT DETECTOR

The regular-sized-object detector was a radically different approach to object perception. The first major difference was the exploitation of global image context obtained in part by the use of local gradient magnitude and direction. In contrast, the closed-region-detector was limited in its use of context to the stringing together of declared edges without the knowledge of each edge's gradient components. The second major difference was the use of the a priori knowledge that objects in a grain grading image have similar sizes. It was assumed that images would be taken with a known or constant magnification so that the object region size in an image could be approximately known beforehand. This assumption was expressed in this research by having the operator enter the approximate object radius. For a practical system the expected size would be calculated by the machine using the known magnification. With these two fundamental changes the regular-sized-object detector was much more dynamic and capable of use in an automated system while at the same time being much more efficient.

The first of the two stages comprising the regular-sized-object detector consisted of roughly determining the location of each object in an image. This operation focussed the computational power of the remainder of the detection procedure to only those parts of the image known to contain an object. Focussing computational power like this was one

way that efficiency was improved as compared to the closed-region detector in which the Haralick edge operator was applied to each and every pixel location. This stage of the detector's operation was accomplished with a Hough transform designed to find roughly circular shapes. This Hough transform utilized both gradient magnitude and direction information from all parts of the image and thereby found prospective object locations which satisfied the global gradient context of the image.

The second stage consisted of finding and specifying the contour of each object previously located. As stated in Section 3.1 the contour of an object was the most obvious edge to be found in a wheat grading image. Thus the task of finding the contour at each object location amounted to finding the single most consistent well defined closed curve edge which circumscribed the central point of the region and which satisfied the object size constraints. This curve would satisfy the local context of the object's region of the image. By treating the object region as a graph, this task was transformed into a heuristic graph search problem.

Both stages of the regular-sized-object detector utilized the same method, the Sobel edge operator, for approximating the gradient. This is one of the classical edge operators. It was chosen because of its computational simplicity and its good performance in noisy images as compared to the other classical edge operators. The Sobel operator employs two

3 by 3 masks to approximate the gradient components at the centre pixel of a 3 by 3 neighborhood. Figure 3.8 shows these masks. The dot product of mask 1 and a pixel neigh-

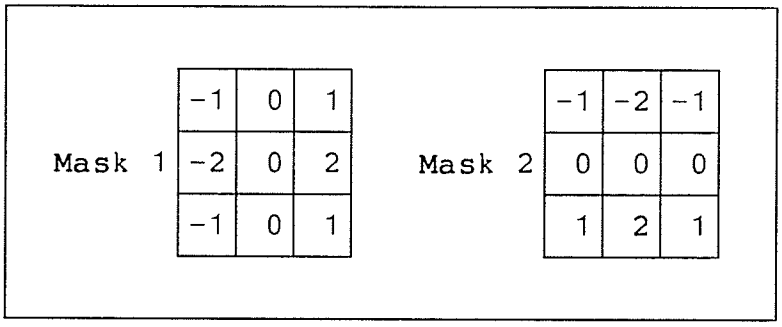


Figure 3.8: Sobel Operator

borhood was f_1, the horizontal component, and that of mask 2 and the neighborhood was f_2, the vertical component. The estimated magnitude and direction of the gradient were then:

$$|\nabla f| = \sqrt{f_1^2+f_2^2} \quad \text{and} \quad \emptyset = \arctan(f_2,f_1) \ .$$

The computational simplicity of the Sobel operator provided a further improvement in efficiency for the regular-sized-object detector.

3.3.1 The Hough Transform

The Hough transform engaged in the regular-sized-object detector used both gradient magnitude and direction to detect roughly circular curves. Ballard [1981] presents an excel-

lent overview of the development of the Hough transform through to its present form and presents examples which fortuitously cover most of the theory of the two configurations of the transform involved in this research.

Originally the Hough transform was designed to detect simple parametric curves in an image by using only edge magnitude information gleaned from the image. In effect it transforms the image into a parameter space in which each dimension represents an alterable parameter of the desired curve. The value of each point in this space is proportional to some estimate of the likelihood of the existence of a curve possessing the parameters denoted by the point's position. This parameter space is dealt with in a discrete form called an accumulator array. The Hough transform has the advantage that detection of even a severely broken curve in a noisy image is possible where it would not be in many other strategies.

Since its inception the transform has been improved in several ways. One improvement was the use of edge orientation information as was done for the regular-sized-object detector. This enhancement substantially reduced the computation time since the dimensionality of the locus of curves implied by a single edge element was reduced by one. This enhancement also increased the accuracy of detected curves. Other innovations allowed the detection of generalized nonanalytic shapes and of composite shapes composed of several

simpler shapes. These latter innovations were not employed in this research.

The Hough transform generated the transformed version of an image by first determining the subset of the desired class of curves implied by each suitable edge element in the image, and then adding a suitable contribution to the regions of the accumulator array possessing parameters which corresponded to the implied subset. Some criterion had to be determined for discriminating between suitable and unsuitable edge elements. For the regular-sized-object detector the criterion was simply that the 3 by 3 neighborhood of which a pixel was the centre had to exhibit a Sobel gradient magnitude greater than some threshold. This threshold value was usually set to 125 since this value provided good discrimination between edges caused by object contour and those caused by interior object detail. Some strategy also had to be determined for making an edge element's suitable contribution to the regions of the accumulator array implied by the edge element. For the regular-sized-object detector, the implied accumulator array entries were simply incremented by one. This incrementation strategy and the edge element suitability criterion previously described biased the Hough transform in this research towards finding objects whose contours were largely well defined.

The form of the accumulator array and the method of calculating the curve parameters implied by each edge element

were determined by the class of curves to be detected. The class of curves that were of interest for the regular-sized-object detector were circles of only a single radius R, as entered by the operator, but at any location in the image. The circle was selected as a shape because of its analytic simplicity, its symmetry and because only the rough circularity of objects in the image was being assumed. For a circle of radius R whose centre was at row b and column a, the row r and column c of a point on the circle were given by:

$$(c-a)^2+(r-b)^2 = R^2 \ .$$

The accumulator array was two-dimensional since two dimensions were required to represent the circle's parameters a and b. This array was in registration with the image so that each of its positions represented the location of the centre of a prospective circular object. The above equation used only the location of the edge element at (r,c) to define the locus of the centres of possible circles on which it may have lain. In parameter space this locus was itself a circle of radius R.

To introduce the use of edge orientation in determining the parameters of an indicated curve, the equation for the curve was also given in terms of its slope. This slope was perpendicular to the gradient of an edge represented by the curve. For the above equation of a circle differentiated

with respect to column, the slope, or equivalently, orientation of an edge element was given by:

$$\frac{dr}{dc} = \tan(\emptyset \pm \frac{\pi}{2}) = -\frac{(c-a)}{(r-b)}$$

where $\emptyset$ was the gradient direction and $\emptyset \pm \frac{\pi}{2}$ were the two interpretations of edge orientation. By combining this expression for edge orientation with the previous expression for edge location, the locus of the centres of possible circular objects implied by an edge element at (r,c) exhibiting a gradient direction $\emptyset$ was given by:

$$a = c - R \cdot \cos \emptyset$$

$$b = r - R \cdot \sin \emptyset$$

when the objects were dark on a light surface. Obviously by including the use of edge orientation an edge element had been made to specify the location of a single circular object of radius R. Figure 3.9 shows the geometry of this method.

Unfortunately the objects which the regular-sized-object detector was intended to detect were never perfectly circular in shape but rather were elliptical, oval or at least irregularly shaped. This problem of noncircular shape was a major source of error in the accumulator array since the calculation of the location (b,a) of the object centre implied by an edge element rested upon the premise that the object was circular. This error was interpreted as result-

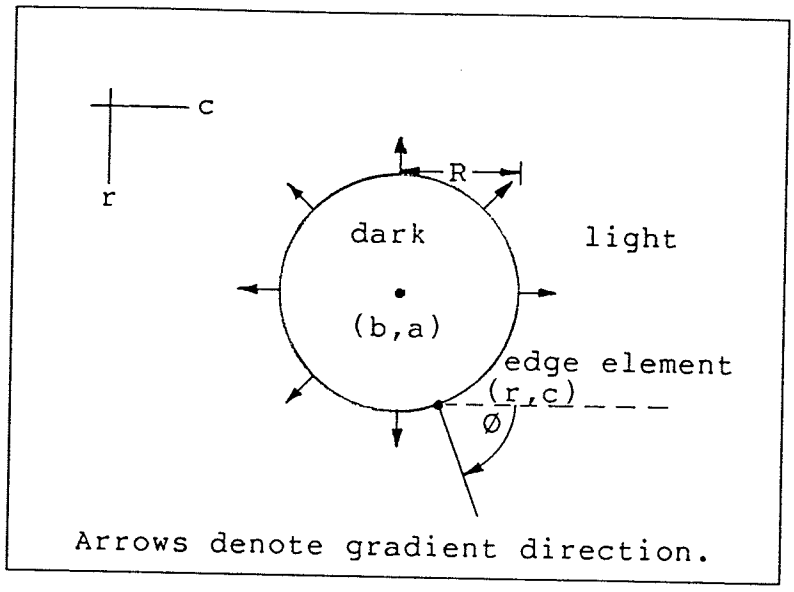


Figure 3.9: Geometry of Circle Detection by H. T.

ing from uncertainties in R and the estimated Ø such that:

$$Ø-\Delta Ø < Ø' < Ø+\Delta Ø$$

$$R-\Delta R < R' < R+\Delta R$$

where Ø' and R' were the actual values. An error compensating convolution template, as described by Ballard, was the device used to deal with this problem. Instead of only incrementing a single array position (b,a) implied by an edge element, a group of array positions in a region centred on (b,a) and similar in size to the area of the expected uncertainties in R and Ø was incremented. For simplicity the template was a square region composed entirely of 1's. A template size of 0.7 R by 0.7 R was chosen based on general observations about the actual shape of the objects expected

in wheat grading images. The large size of the error template was a manifestation of the fact that this Hough transform technique was meant to detect the approximate location of irregularly shaped objects rather than the exact location of perfectly circular objects.

In summary, the Hough transform algorithm used for the regular-sized-object detector was as follows:

1. Set each entry in the accumulator array to 0.
2. For every 3 by 3 neighborhood centred at (r,c) in the image, do:
   a) Calculate Sobel gradient direction Ø and magnitude.
   b) If the gradient magnitude is less than threshold, start next neighborhood; otherwise continue.
   c) Calculate centre (b,a) of implied circle using
      $a = c - R \cdot \cos \emptyset$
      $b = r - R \cdot \sin \emptyset$.
   d) Increment all accumulator array positions within the 0.7 R by 0.7 R square centred at (b,a).
3. Determine position of the next object by finding the location of the largest entry in the accumulator array.

The regular-sized-object detector performed this algorithm only at the beginning of its operation and in this fashion found the first object. After the detector found each object's contour, it set the accumulator array entries within the contour to 0. Thus the detector found each subsequent object by again finding the location of the largest entry in the array.

3.3.2 The Heuristic Edge Search

The method developed to find the most likely contour edge in an object region was an extension of the popular graph search approach to the more general problem of locating a single edge between a start position and an end position in a digital image. With this approach the problem was conveniently expressed as a search in a graph for the least cost path between a start node and a goal node. In this graph each node corresponded to an image pixel. An edge was represented by a path comprised of nodes and interconnecting arcs in the graph. Each arc had an associated nonnegative cost c which a path incurred if the arc was on the path. With a properly designed cost function the least cost path represented some form of the best edge. In general the cost function had to generate a cost inversely related to the likelihood of the presence of an edge.

The advantages of this interpretation of the edge finding problem were that the characteristics of the desired edge could be easily changed by changing only the cost function and that several powerful techniques already existed for searching graphs. Some of these techniques employ heuristic information to increase the speed of a search. The fruitful idea of applying heuristic graph search methods to general edge detection was first proposed by Martelli [1972].

Much of the methodology for the heuristic edge search strategy in this research was derived from a paper by Les-

ter et al. [1978]. The object of their work was to complete the contour and thus separate the region of several touching white blood cells in a micrograph image. Part of the contour of a cell was determined by some other means prior to the application of an edge search. The major extension to their work by this research was to determine an entire closed contour of an object with only the general location of the object being known beforehand. They did extend their work to tracing an entire cellular contour, but a starting point had to be known before application of the edge search. Other differences include the form of the cost function and the method used to incorporate circularity in the algorithm.

The graph used for the regular-sized-object detector was shaped like a circular annulus having the object location estimated from the Hough transform situated at its centre. The assumption that the objects to be imaged were roughly circular in shape was the basis for the choice of the graph's form. The centre point of the graph was called the pivot point. The constraints on the size and shape of the object contours to be found were made very loose by having a large outer radius and a small inner radius for the graph. These radii were 3R and $\frac{1}{3}$R respectively, where R was the estimated object radius entered by the operator. The start/goal nodes, each of which performed the dual duties of both start and goal node, were placed along a vertical line emanating from the pivot point to the top of the graph. A typ-

ical search of the graph would begin on the left of the start/goal nodes, proceed in a counterclockwise direction around the pivot point (hence its name), and terminate after approaching the start/goal nodes on their right. Figure 3.10 shows the form of the circular-annulus graph.

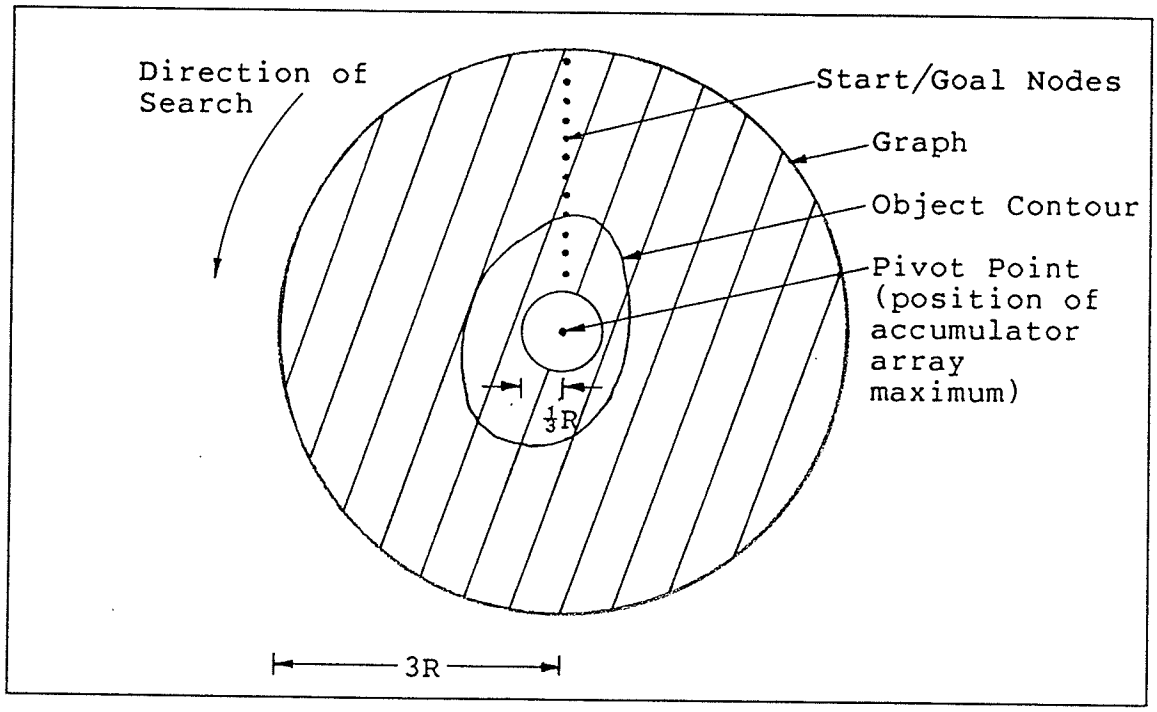


Figure 3.10: Circular-Annulus Graph

The expected and desired features of the contour edge determined which arcs were allowed in the graph. The first feature was that a contour would be 8-connected so that contiguous points on the contour could be either horizontal, vertical or diagonal neighbors. Therefore an arc was al-

lowed to be in one of eight possible directions. These arc directions were enumerated with the same Freeman chain code used in the rest of this research. The second feature was that a contour could proceed in a counterclockwise rotary direction around the pivot point and in a radial direction toward or away from the pivot point. However the contour would not be allowed to reverse its rotary direction and proceed clockwise since the expected objects were not sufficiently irregular in shape to exhibit this rotary backtracking. Therefore the number of possible directions of an arc emanating from any node was restricted to five by the node's position relative to the pivot point. This group of five contiguous directions was centred on the preferred direction which was an angle of $\frac{\pi}{2}$ greater than the direction of the pivot point from the node. From any node in the graph the preferred direction corresponded to the direction of counterclockwise rotary motion. The final feature was that sharp turns, angles greater than $\frac{\pi}{4}$, would not be allowed on a contour. This requirement was based on the observation that sharp turns did not occur on valid contour edges: only on spurious edges resulting from noise. Therefore each time a path was extended by adding a new arc to join the current node to some neighboring successor node, the new arc would only be allowed if its direction was different by an angle of either $\frac{\pi}{4}$ or 0 from the direction of the arc joining the previous node to the current node. This feature also ensured that the contours generated were thin in the

8-connected region sense thus satisfying the thinness property desired for contours. Figure 3.11 shows the arc selection process for a typical current node neighborhood.

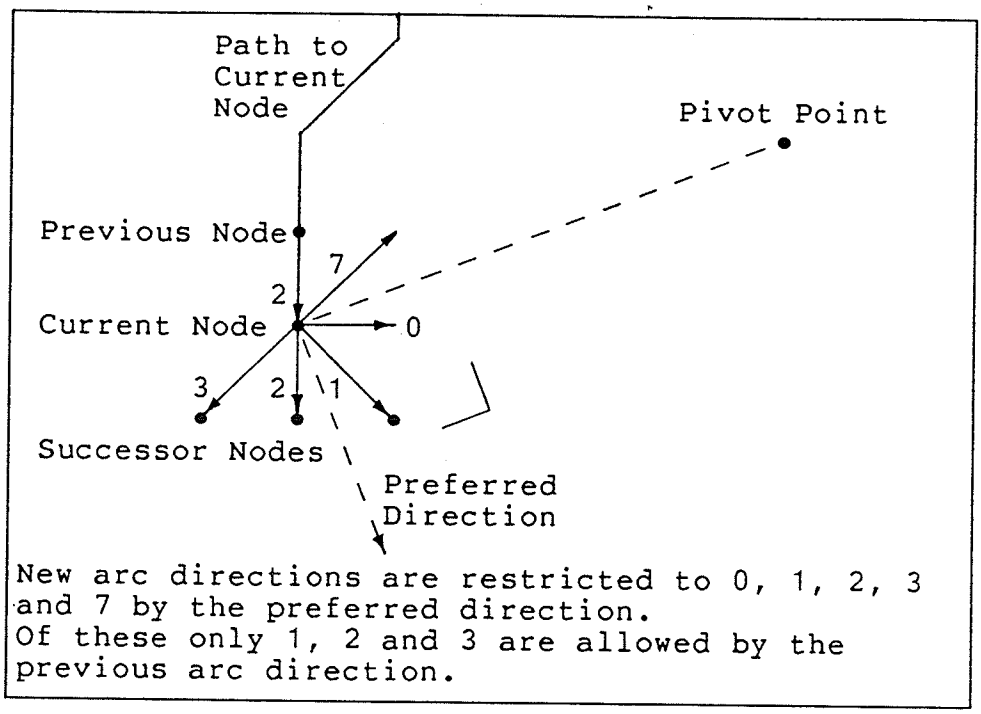


Figure 3.11: Arc Selection Process

The Algorithm A*, a powerful heuristic graph search technique formulated by Nilsson [1971], was the tool employed to find the least cost path. In the terminology of Nilsson, the function **f** which evaluates the suitability of a node n in a path is:

$$\mathbf{f}(n) = \mathbf{g}(n)+\mathbf{h}(n) \ .$$

g(n) is the cumulative cost of the least expensive path found between a start node and n. **h**(n), a type of heuristic information, is an estimate of the cost h(n) of the least expensive path from n to a goal node. When **h**(n) is a lower bound of h(n), the solution path found by Algorithm A* has least cost. The closer **h**(n) is to h(n) the more efficient the search will be. In this research **h**(n) was set equal to 0 because, while the optimal path was desired, h(n) was difficult to estimate. Thus the search strategy used in this research was a special form of the Algorithm A* called a "uniform cost algorithm", in which **h**(n) is unavailable.

Algorithm A* develops paths in a graph through the iteration of a process called node expansion. Expansion consists of replacing the lead node of a path, called an opened node, with successor nodes in a file generally called OPEN. Each successor node is reached by traversing an arc emanating from the old lead node. The extension to the path is recorded by directing a pointer associated with each successor node back to the old node. The old node is closed by placing it in a file generally called CLOSED, since it is no longer the lead node of a path. Before each expansion the node possessing the least **f** is selected from OPEN as the next node to be expanded. Thus only the currently most attractive path is lengthened.

Two characteristics of Algorithm A*, other than its use of heuristic information, contribute to its efficiency.

First, Algorithm A* is a dynamic programming algorithm: at any time during a search it records only one path to each encountered node, and this is the least cost path between the node and any start node. A result of this characteristic is that, of the vast number of paths between a start and a goal node, only a small number are ever considered during a graph search. Second, because this algorithm is a sequential search, it will not consider any more nodes than are actually required to find the least cost path. For an edge search, if the edge is well defined, the operation of the algorithm can become equivalent to a simple edge following technique which considers only those nodes actually on the edge. For a worse edge the search will spread out and consider more nodes.

Algorithm A* was modified in two ways to improve its efficiency. First, each node had two flags, one labelled "marked" and one labelled "opened", associated with it. These indicated whether the node had been encountered and if so whether it was currently open or closed. As a result a search determined a successor node's status simply by inspecting the node's flags rather than searching both OPEN and CLOSED for it. Second, the file CLOSED was not installed since a node's flags indicated the node's closed status. However the file OPEN was retained since its use drastically diminished the time required to find the next node to be expanded. A search quickly determined the next

node by searching this file, which contained only the opened nodes, for the lowest value of **f**, instead of searching the entire graph for the opened node with the smallest **f**.

The above two flags together with the node pointer and three other flags of each node were stored in the array GRAPH. This array was in registration with the image. Figure 3.12 shows the bit definitions of GRAPH.

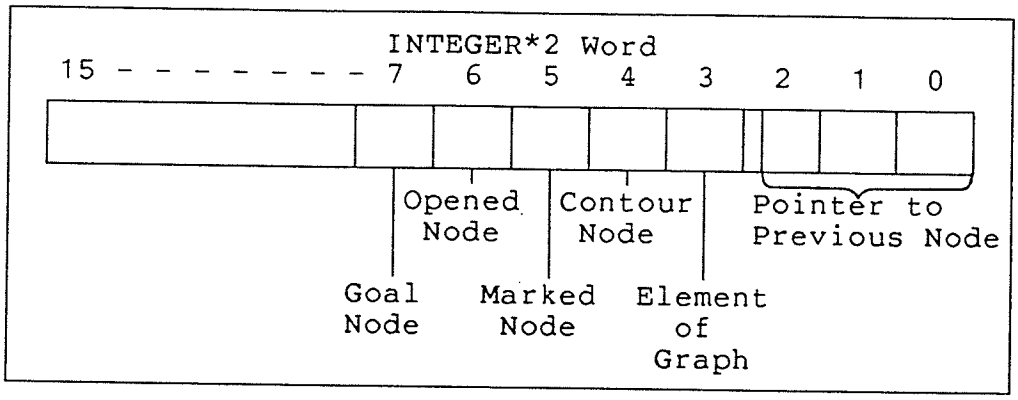


Figure 3.12: Bit Definitions of GRAPH

The graph search used a form of heuristic information to improve its efficiency. The search measured the progress of any path by the angular displacement about the pivot point between the path's lead node and the column of start/goal nodes. It was assumed that if this angle, called $\Omega(n)$, for the path's lead node n was less than the maximum Ω of all opened nodes minus some angular back step limit, then the path represented by n could be discarded as a contender for the least cost path. Thus when the search selected the next

opened node from OPEN, the node was closed and not expanded if it failed the back step limit test. This back step limit was set to $\frac{\pi}{2}$. In general the graph search was not optimal with the use of this heuristic. However many tests were conducted to compare the suboptimal with the optimal search results for typical wheat grading images; these tests demonstrated that even with this heuristic a search almost always produced the least cost path.

The form of the Algorithm A* used for the regular-sized-object detector, including modifications, was as follows:

1. For each start/goal node, set its goal-node bit, place its coordinates in OPEN and set its cost **f**=0. However leave it unmarked so that it may fulfill the goal node function.

2. Find the node n in OPEN with the lowest value of **f** and mark it closed by resetting its opened-node bit; if OPEN is empty abort the algorithm with failure.

3. If n is a marked (already encountered) start/goal node then an optimal path has been found so exit the algorithm; otherwise continue.

4. Calculate $\Omega(n)$ and if it is less than $\Omega(max)$-back step limit, go to 2.; otherwise continue.

5. Expand n by generating its successors, each of which must:

   a) be an element of the graph,

   b) be reachable by one of the five arcs centred on the preferred direction, and

   c) satisfy the no sharp turns rule.

6. For each successor node not already marked calculate **f**, place its coordinates in OPEN, set its opened-node and marked-node bits and calculate its Ω. If its Ω is greater than $\Omega(max)$, then replace $\Omega(max)$ with the new value.

7. For each successor node already marked calculate its new value of **f** but store only a temporary version of it. If the new value of **f** is not less than the old value, then omit this successor. Otherwise replace the old value of **f** with its new value, redirect the successor's pointer to n and if it is currently marked closed, mark it open and place its coordinates in OPEN.

8. Go to 2.

The quality of the cost function used to calculate c(n) was of paramount importance since the quality of the detected edge depended directly upon it. The cost function for the regular-sized-object detector was composed of three parts as follows:

$$c(n) = \text{DADJUST}(n) \cdot \text{RADJUST}(n) \cdot \text{GRADCOST}(n) \ .$$

The cost of the gradient, GRADCOST(n), utilized the gradient magnitude and direction to gauge the suitability of the pixel neighborhood around n for having an edge passing through n in the same direction as the arc leading to n. The equation of the gradient cost was as follows:

$$\text{GRADCOST}(n) = \begin{cases} \text{GRADCOST}', & \text{GRADCOST}' \geq 0 \\ 0, & \text{otherwise} \end{cases}$$

where $\text{GRADCOST}' = \text{BASECOST} - s(n) \cdot \cos\{2[\text{arc direction}(n) - \phi(n) + \frac{\pi}{2}]\}$,
$s(n)$ was Sobel gradient magnitude at n,
$\phi(n)$ was Sobel gradient direction at n, and
arc direction was angle of the arc leading to n.

GRADCOST never produced a negative cost since Algorithm A* can provide an optimum path only if every cost incurred is

nonnegative. BASECOST was the cost incurred if the neighborhood contained no detail indicating an edge, in which case s(n) = 0. The value of BASECOST was normally 250. If the arc direction was within an angle of $\frac{\pi}{4}$ of the edge direction, $\emptyset(n)-\frac{\pi}{2}$, implied by the gradient direction then GRADCOST would be less than BASECOST. Otherwise it would be greater than BASECOST. The magnitude of this divergence about BASECOST was proportional to the gradient magnitude. Figure 3.13 shows these relationships.

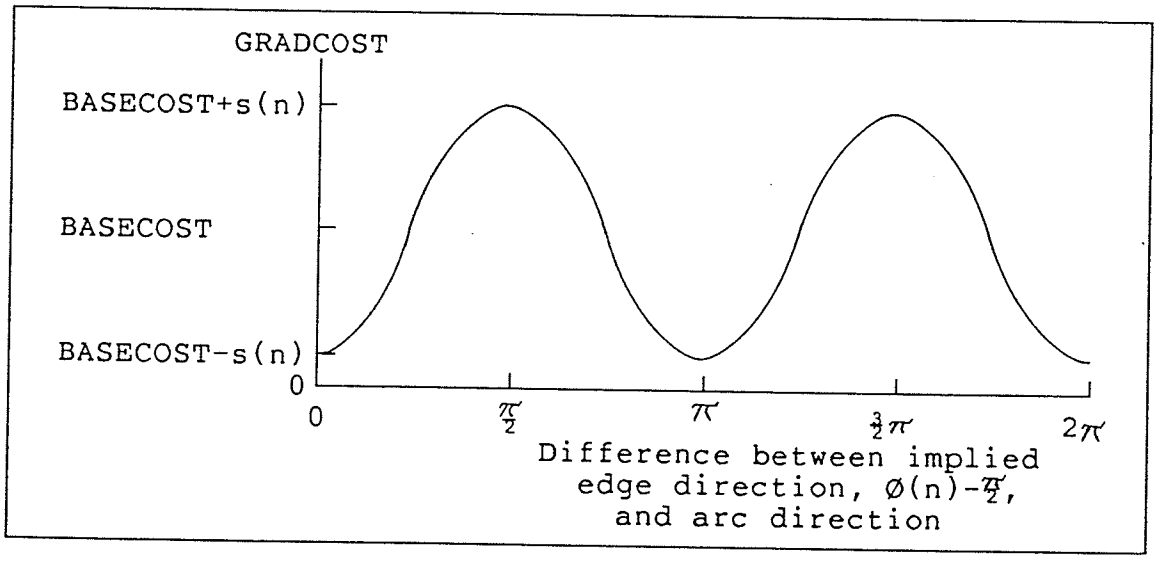


Figure 3.13: Operation of GRADCOST

A drawback of GRADCOST was that it was insensitive to the relative gray level of the regions being divided by an edge. In other words it could respond the same to either an edge dividing a dark region on the left and a light region on the right, or one dividing a light region on the left and a dark

region on the right. This property was a result of the two-fold symmetry of GRADCOST from 0 to 2π.

The radial adjustment, RADJUST(n), neutralized the preference for shorter small-radius paths near the pivot point and against longer large-radius paths far from the pivot point. The effect of RADJUST was equivalent to mapping the circular-annulus graph nodes onto a rectangular unfolded version of the original graph. The expression for RADJUST was simply:

$$\text{RADJUST}(n) = \frac{1}{d(n,\text{pivot point})}$$

where d was the Euclidean distance.

The diagonal adjustment, DADJUST(n), was identical to that used by Lester et al. This adjustment neutralized the preference for diagonal paths. This preference was caused by the smaller requirement for nodes in a diagonal path covering the same distance as a vertical or horizontal path. DADJUST was given by:

$$\text{DADJUST}(n) = \begin{cases} \sqrt{2}, & \text{if the arc leading to n was diagonal} \\ 1, & \text{otherwise.} \end{cases}$$

The "closing" of a solution path was a significant problem in designing the regular-sized-object detector. The optimal version of Algorithm A* searching the circular-annulus

graph finds the least cost path which begins at a start/goal node, circumscribes the pivot point and ends at a start/goal node. However the start/goal node at the beginning of the path is not guaranteed to be the same start/goal node at the end of the path; sometimes they are indeed different and the resulting path is not closed. If the path is not closed it cannot represent a complete object contour.

It is enlightening to consider how the least cost closed path solution can always be found. A closed path must begin and end at the same start/goal node. Therefore one method to find the least cost closed path in an annular graph containing t start/goal nodes would be to conduct t searches, each having only one start/goal node which would be one of the original t start/goal nodes. Then the least expensive of the t paths found by the t searches would be the solution path. Of course this method would be computationally very expensive. A more efficient method would make use of a three-dimensional graph. Each of the t levels of this graph would have only one of the t start/goal nodes and would not have any interconnection with any other level. The search would be conducted simultaneously on all levels of the graph and its single solution path would be the least cost closed path about the pivot point. While this method would be more efficient, the amount of computer memory required would be colossal for the standard object region size.

Due to the difficulty in finding the least cost closed path, a simple but suboptimal approach to this problem was taken. The Algorithm A* search operated on the original circular-annulus graph to yield the least cost, but possibly open, path. If this path was found to be open then the search continued, further extending the path, until finally the first start/goal node of the path and the last start/goal node of the path were one and the same node. In most circumstances the resulting path was optimal; in a few it was not. However the resulting closed path was accepted as the object contour because a "good" and not necessarily optimal solution was considered to be sufficient. Effects of this suboptimality were rare and when present were small. Figure 3.14 shows a situation in which the optimal solution has not been found with this approach.

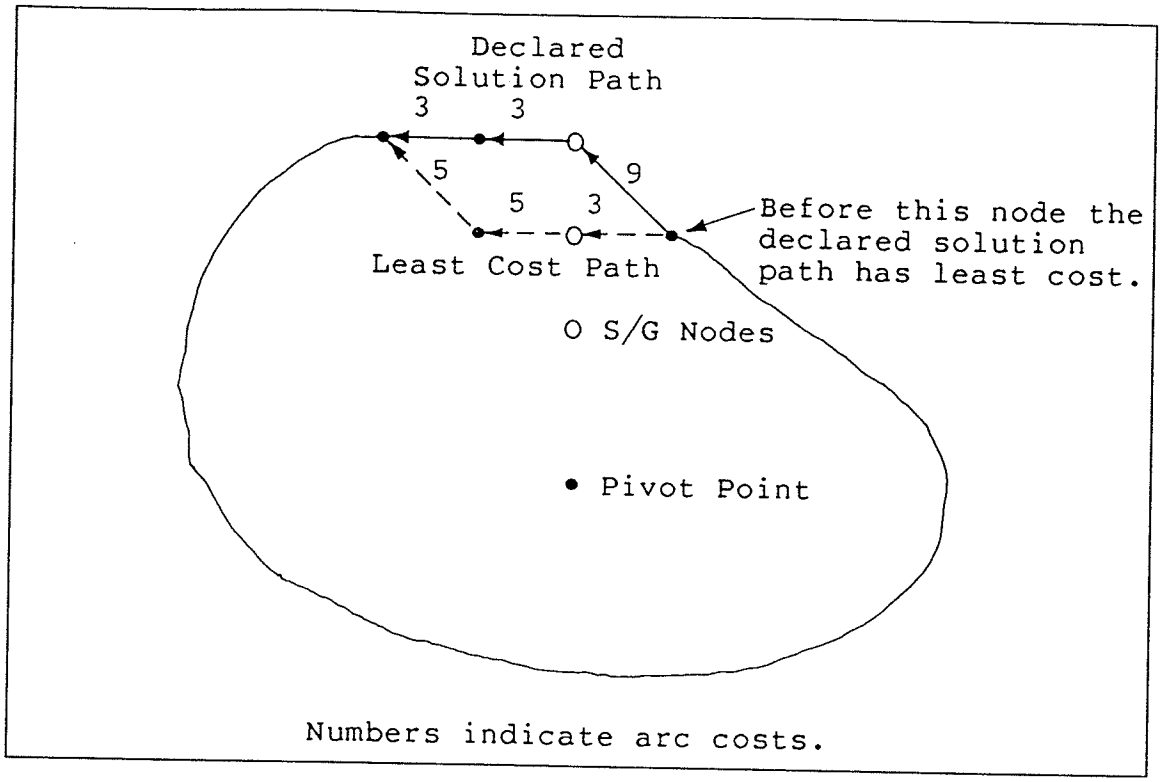


Figure 3.14: Suboptimality of Path Closing Method

3.3.3 Performance

Figure 3.15 and Figure 3.16 show the results of the regular-sized-object detector operating on JUMBLE1 and JUMBLE2 respectively. Each closed curve represents the contour of a declared object. For both runs of the detector the entered value of R was 20. Hence the outer diameter of the circular-annulus graph was set to 120. This was an ample size since the object lengths ranged from 65 to 89.

The performance of the regular-sized-object detector operating on JUMBLE2 was a drastic improvement over that of the closed-region detector operating on the same image.

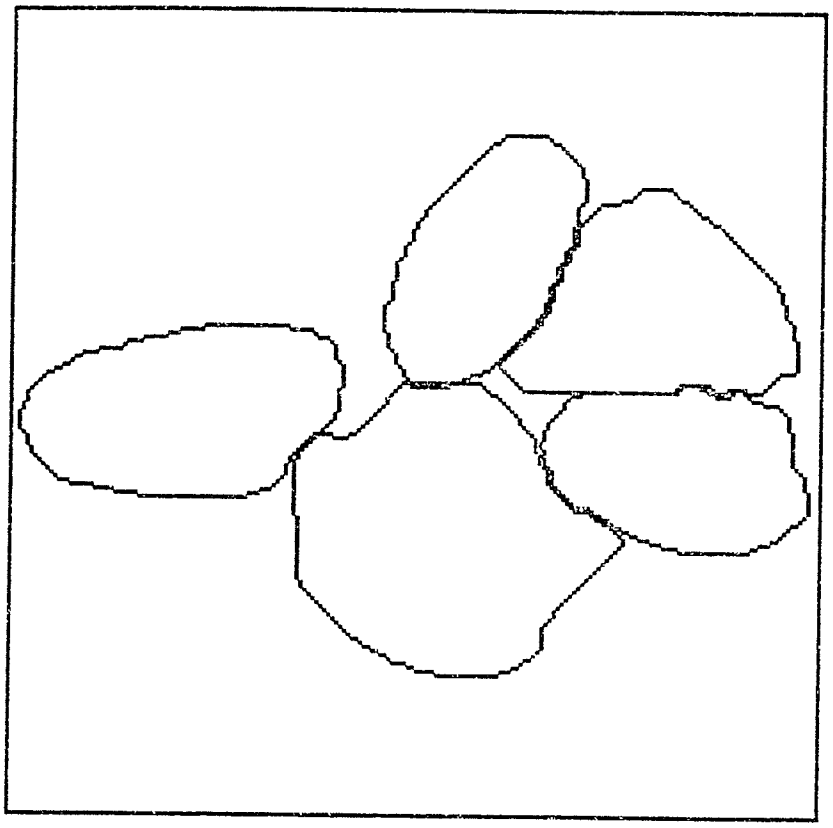


Figure 3.15: Contour Image of JUMBLE1 from R-S-O Detector

Each of the wholly visible objects in the image was detected as being a single object and specified by a declared contour closely corresponding to the actual contour. As desired the Hough transform ignored the partially visible object in the lower right corner since too little of the object region was available in the image to allow the formation of a circular-annulus graph. In short the detector produced almost perfect results with this image.

Only two flaws were evident and these were minor. First, the shadow which obscured the contact point between the two

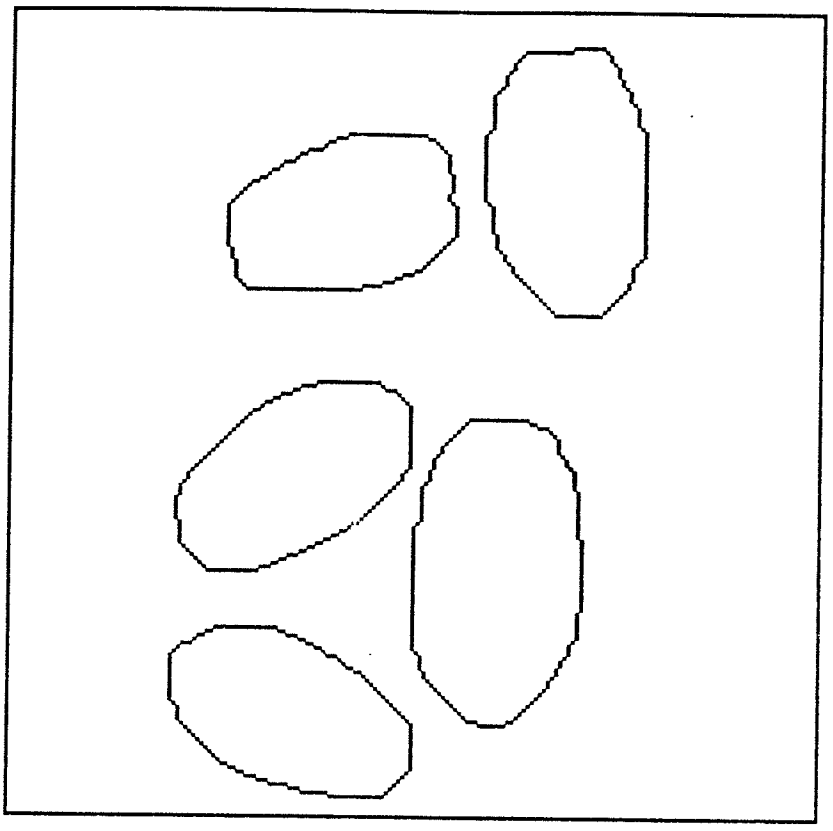


Figure 3.16: Contour Image of JUMBLE2 from R-S-O Detector

touching objects deceived the edge search. As a result the search partly omitted the tip of one of these objects from the declared contour. Second, the edge search gave portions of some contours a "fragmented" quality. In comparison these same contours did not receive this quality from the Haralick edge operator of the closed-region detector (Figure 3.5). This quality, a side effect of image equalization, indicated an over sensitivity of the edge search to noise.

The performance of the detector on JUMBLE1 was much worse. On one side of two separate objects the more dominant contour edges of neighboring objects distracted the edge search. Due to the large size of the circular-annulus graph, these diverted searches were able to cover large areas while erroneously following the neighboring edges. These failures were partly a result of the edge search cost function's inability to distinguish between object on left/background on right edges and object on right/background on left edges. Another defect was the exclusion of a small portion of one end of the largest object from its contour. This end could not be detected because it protruded slightly outside the graph. This was a consequence of two problems: the graph pivot point was displaced by an approximate distance of 20 left of the actual object centre and the object was abnormally large. In total only one object was satisfactorily detected. This, the upper middle object, suffered least from surrounding shadow. As occurred with JUMBLE2, some edges retained a fragmented quality.

The number of node expansions performed in each edge search gives a rough measure of the efficiency of the search. For the ten edge searches conducted on these two images the average number of expansions was 4718. The minimum was 2570 for the upper middle object in JUMBLE1 and the maximum was 6091 for the lower middle object in the same image. The wide range in values implied that for some search-

es a lot of unnecessary work was being done in mistakenly following the contour edges of neighboring objects.

3.4 HYBRID DETECTOR

The hybrid detector was designed to improve on the rejection of noise during an edge search by taking advantage of the excellent performance of the Haralick edge operator. The fragmented quality of some of the contours declared by the regular-sized-object detector motivated the development of this detector.

The hybrid detector was identical to the regular-sized-object detector except for its edge search cost function. Instead of the Sobel edge operator, this detector approximated the gradient with the Haralick edge operator employing a 5 by 5 neighborhood size. If the neighborhood satisfied the distance to the zero crossing of the second directional derivative test, then a cost function similar to that of the regular-sized-object detector determined the cost. The only difference was the value of BASECOST. However, if the neighborhood did not pass this test the cost assigned was 10 times BASECOST. This large penalty cost effectively constrained the declared edge to only those pixels satisfying the second directional derivative test. The only exceptions were the rare breaks sometimes produced by the Haralick edge operator in an object contour. An edge search had to bridge these breaks by incurring the penalty cost.

Figure 3.17 and Figure 3.18 show the contour edge images produced by the hybrid detector operating on JUMBLE1 and JUMBLE2 respectively. The value of R entered by the operator was the same as that used to produce the results of the

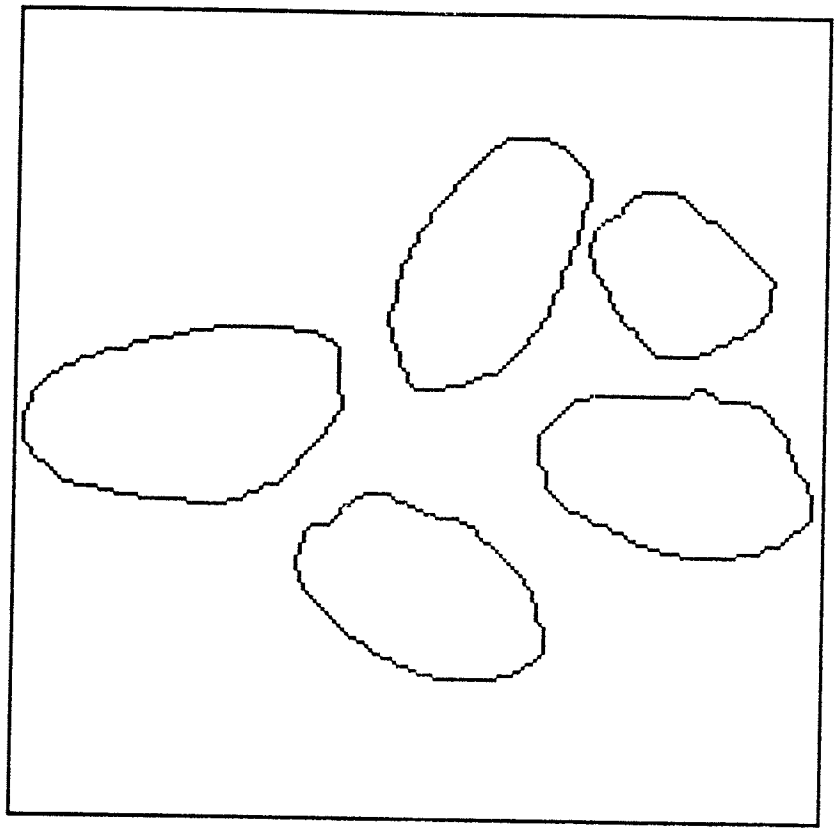


Figure 3.17: Contour Image of JUMBLE1 from Hybrid Detector

regular-sized-object detector presented in Subsection 3.3.3.

Both images exhibited improvements over the contour edge images generated by the regular-sized-object detector. The fragmented quality of most edges was eliminated. Unexpectedly none of the edge searches conducted on JUMBLE1 were de-

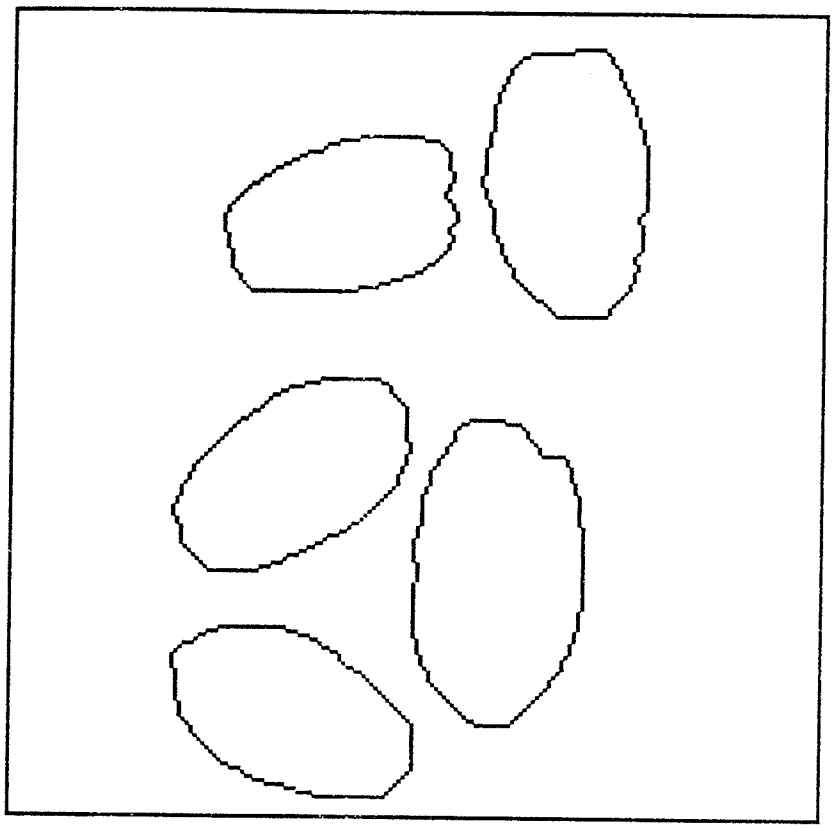


Figure 3.18: Contour Image of JUMBLE2 from Hybrid Detector

ceived by contours of neighboring objects, as occurred with the regular-sized-object detector. This was likely due to the influence of the second directional derivative test. For each object, the closed curve of the contour edge pixels which passed this test were typically bounded on the inside and outside by a margin of pixels not passing the test. The high penalty cost of crossing this margin restrained a search from diverging over the background to a neighboring object.

Three defects were evident. The same two objects, which had a fraction of one end ignored by the regular-sized-object detector, again had almost the same areas ignored by the hybrid detector. The causes were also the same: shadow partially obscured the object's end near the contact point in JUMBLE2, and in JUMBLE1 the object's end exceeded the limit of its graph. A third and more serious flaw occurred in JUMBLE1. Almost one-half of the upper right object, one of the pair of touching objects, was omitted from its declared contour. This was evidently caused by the shadow near the contact point which severely obscured the contour of the ignored end. Surprisingly the declared contour of the other object in this pair was a good representation of its actual contour.

The number of node expansions required by the hybrid detector was generally less than that for the regular-sized-object detector. The average number of expansions performed in the ten edge searches conducted on JUMBLE1 and JUMBLE2 was 3106. The minimum was 1769 for the upper middle object in JUMBLE1 and the maximum was 3994 for the upper right object in JUMBLE2. However the amount of computation required for each opened node was much greater than it was for the regular-sized-object detector. When a node was first opened the application of the Haralick edge operator required 225 multiplications and 225 additions just to derive the cubic model. Thus the overall computation was substantially greater than for the regular-sized-object detector.

The hybrid detector was a classic example of diminishing returns. The edge quality had been marginally improved and the edge search had been made less prone to divergence from the actual contour; but these refinements were made at the cost of shouldering the heavy computational burden of the Haralick edge operator. Even with the application of this computational brute force drastic errors in a defined contour edge could still occur, as was demonstrated in JUMBLE1. Consequently the use of the Haralick edge operator was not pursued further as an alternative to the Sobel edge operator in the edge search.

3.5 ELLIPTICAL-OBJECT DETECTOR

The elliptical-object detector was the most superior object perception technique, in terms of reliability, edge quality and efficiency, developed during this research. As such it is the technique advocated by this research. Essentially it was a modified version of the regular-sized-object detector. Two general refinements were made to the regular-sized-object detector to create the elliptical-object detector. These were inspired by some of the deficiencies of the regular-sized-object detector observed during its operation on test images.

The first refinement was to constrain the edge search to a graph of much less area while ensuring that the entire actual object contour remained within the graph. By limiting

the search area the number of unrelated edge elements, from the object's surface detail or from neighboring objects, was substantially reduced. Thus the chance of an edge search being diverted to an erroneous edge was diminished and the reliability of the detector, particularly in images of closely spaced objects, was increased. In addition, the detector's efficiency was improved since a graph with fewer nodes required less search effort.

The shape and size of the graph were chosen through consideration of the expected object features. As pointed out in Subsection 1.3.2 the objects could be expected to be of similar size due to prior cleaning of the grain sample. For the elliptical-object detector the objects were also expected to be roughly elliptical in shape. This was of course true for wheat kernels and foreign grain kernels and this could be a relatively safe assumption for other foreign objects if the size limits of the graph were sufficiently unrestrained.

The shape selected for the graph was an elliptical annulus. The inner and outer borders of this graph were ellipses centred on the same point and aligned in the same orientation. The inner ellipse was 40 percent smaller than the expected elliptical object size and the outer ellipse was 40 percent larger than the expected size. The span of the resulting graph, which determined the acceptable object sizes, was intended to enable detection of any object likely to ap-

pear. The aspect ratio of the graph, b/a, was set to 0.565 based on the typical wheat kernel shape. Eight orientations of the graph were allowed, ranging from an angle of 0 to $\frac{7}{8}\pi$ in $\frac{1}{8}\pi$ angular increments. This number of orientations was a compromise between having sufficient fineness of orientation adjustment and having a manageable accumulator array size for the Hough transform. Figure 3.19 shows a typical elliptical-annulus graph.

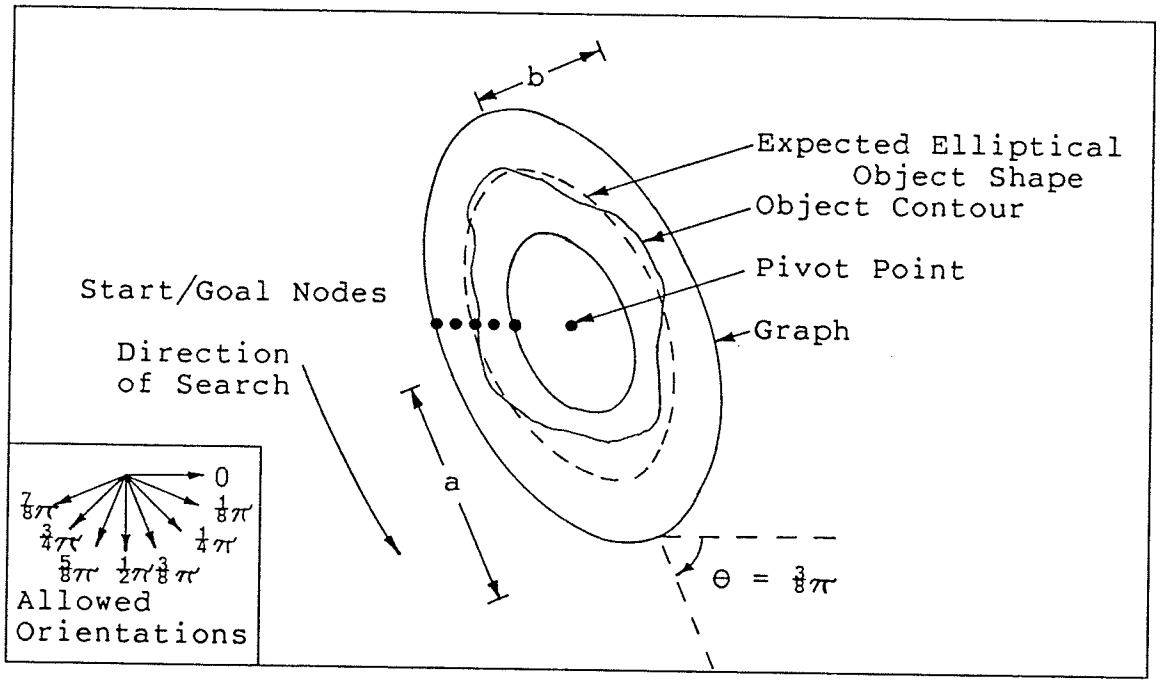


Figure 3.19: Elliptical-Annulus Graph

Closing the solution path was a problem with this detector as it was with the regular-sized-object detector. The elliptical-object detector used the same method to close the solution path and therefore the optimality of the solution

path was again not assured. To minimize the effects of this problem the start/goal nodes were placed along either a vertical, horizontal or diagonal ray which emanated from the pivot point on or near the minor axis of the graph. In this location the start/goal node line usually crossed the object contour orthogonally, the contour was usually well defined and fewer start/goal nodes were required to cross the graph. As a result the danger of confusing the edge search near the start/goal nodes was minimized. This reduced the risk of arriving at a suboptimal solution path.

The Hough transform was modified to detect ellipses of uniform size, but of any orientation, since this was now the object shape of interest. The attributes of the elliptical shape to be detected were the same as those of the expected object shape: the aspect ratio, b/a, was 0.565 and the expected object length, 2a, was entered by the operator. Ballard [1981] presents the ellipse detection technique as an example.

This technique was developed in the same manner as that for circle detection. It utilized both the gradient magnitude and direction of an edge element to imply a set of ellipses, each being at a different orientation. The r'-axis and the c'-axis were a coordinate set in which an ellipse

was oriented with its major axis parallel to the c'-axis. An edge element on this ellipse indicated the ellipse's centre (h',k') as:

$$h' = r' + Sr\frac{b}{\sqrt{1+\frac{a^2\xi'^2}{b^2}}}, \quad k' = c' + Sc\frac{a}{\sqrt{1+\frac{b^2}{a^2\xi'^2}}}$$

where (r',c') was the edge element's location,
a and b were the major and minor axis of the ellipse (In this case b/a = 0.565 and the operator entered the value of a.),
$\xi = \tan(\emptyset' - \frac{\pi}{2})$,
$\emptyset'$ was the gradient direction, and
Sr and Sc were ±1 depending on the quadrant of (r',c') with respect to (h',k').

The sign functions, Sr and Sc, were defined as:

Sr	Background	
	White	Black
$0 < \emptyset' < \pi$	−1	+1
$\pi < \emptyset' < 2\pi$	+1	−1

Sc	Background	
	White	Black
$-\frac{\pi}{2} < \emptyset' < \frac{\pi}{2}$	−1	+1
$\frac{\pi}{2} < \emptyset' < \frac{3}{2}\pi$	+1	−1

Rotations of the ellipse were accommodated by regarding the r' and c' coordinate set as being a rotated version of the r and c coordinate set. With Θ being the angle of rotation, the transformed coordinates of an edge element (r,c) were:

$$r' = -c \cdot \sin\Theta + r \cdot \cos\Theta$$

$$c' = c \cdot \cos\Theta + r \cdot \sin\Theta$$

and, once the ellipse centre (h',k') implied by (r',c') had

been determined, the reverse transformed coordinates of (h',k') were:

$$h = k' \cdot \sin \Theta + h' \cdot \cos \Theta$$

$$k = k' \cdot \cos \Theta - h' \cdot \sin \Theta \ .$$

Figure 3.20 shows the geometry of this Hough transform.

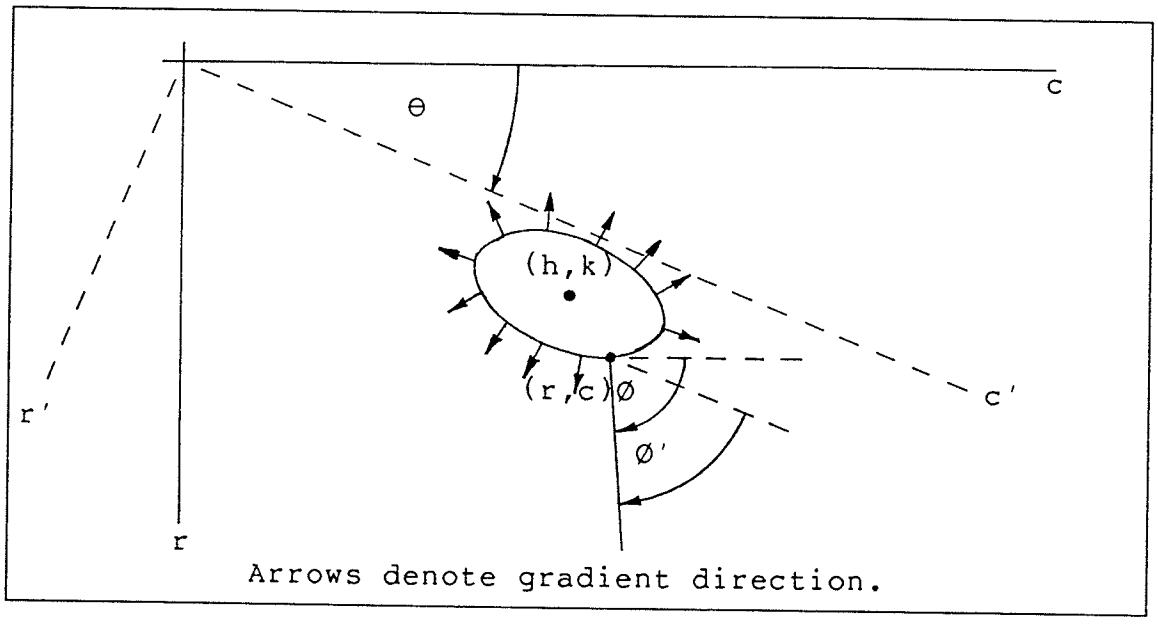


Figure 3.20: Geometry of Ellipse Detection by H. T.

The accumulator array required three dimensions to handle the three parameters h, k and Θ. The eight Θ levels of the array represented the eight possible graph orientations. Each level was in registration with the image. However each level only represented the even rows and even columns of the image so that computer memory was conserved.

As with the regular-sized-object detector an error compensating convolution template was incorporated in the accumulator array incrementation strategy. However less error was expected in the accumulator array since the actual shape of an object was normally closer to the expected object shape. Therefore a much smaller template, 0.2·a on a side, was employed.

In summary, the Hough transform algorithm of the elliptical-object detector was as follows:

1. Set each accumulator array entry to 0.
2. For every 3 by 3 neighborhood centred at (r,c) in the image, do:
   a) Calculate Sobel gradient direction Ø and magnitude.
   b) If the gradient magnitude is less than threshold, start next neighborhood; otherwise continue.
   c) For each of the eight values of Θ, do:
      i) Calculate the transformed edge element location, (r',c'), and gradient direction, Ø'.
      ii) Calculate the implied ellipse centre, (h',k').
      iii) Calculate the reverse transformed ellipse centre, (h,k).
      iv) Increment all accumulator array positions within the 0.2a by 0.2a square centred at (h,k) in the level corresponding to Θ.
3. Determine position and orientation of the next object by finding the location of the largest accumulator array entry.

This Hough transform produced surprisingly accurate predictions of an object's location and orientation. Most often the predicted location and orientation of an object were the same as those later determined by the shape analysis of the object's contour found by the edge search.

The second refinement incorporated in the elliptical-object detector was an improved cost function for the edge search. The improvement was a new version of GRADCOST, similar to the old version, but which could distinguish between object on left/background on right edges and object on right/background on left edges. The validity of this new cost function depended on the assumption that an object pixel would always be darker than a neighboring background pixel if the background was white (vice versa for a black background). Only in extremely rare instances was this assumption incorrect. This new function ensured that an edge search would never be deceived into following the contour edge of a neighboring object. The new equation for GRADCOST was as follows:

$$\text{GRADCOST}(n) = \begin{cases} \text{GRADCOST}', & \text{GRADCOST}' \geq 0 \\ 0, & \text{otherwise} \end{cases}$$

where $\text{GRADCOST}' = \underset{\text{white background}}{\text{BASECOST}-s(n)\cdot\cos[\text{arc direction}(n)-\emptyset(n)+\frac{\pi}{2}]}$,

$\underset{\text{black background}}{\text{BASECOST}-s(n)\cdot\cos[\text{arc direction}(n)-\emptyset(n)-\frac{\pi}{2}]}$,

$s(n)$ was Sobel gradient magnitude at n,
$\emptyset(n)$ was Sobel gradient direction at n, and
"arc direction" was the angle of the arc leading to n.

Figure 3.21 shows the operation of this new GRADCOST.

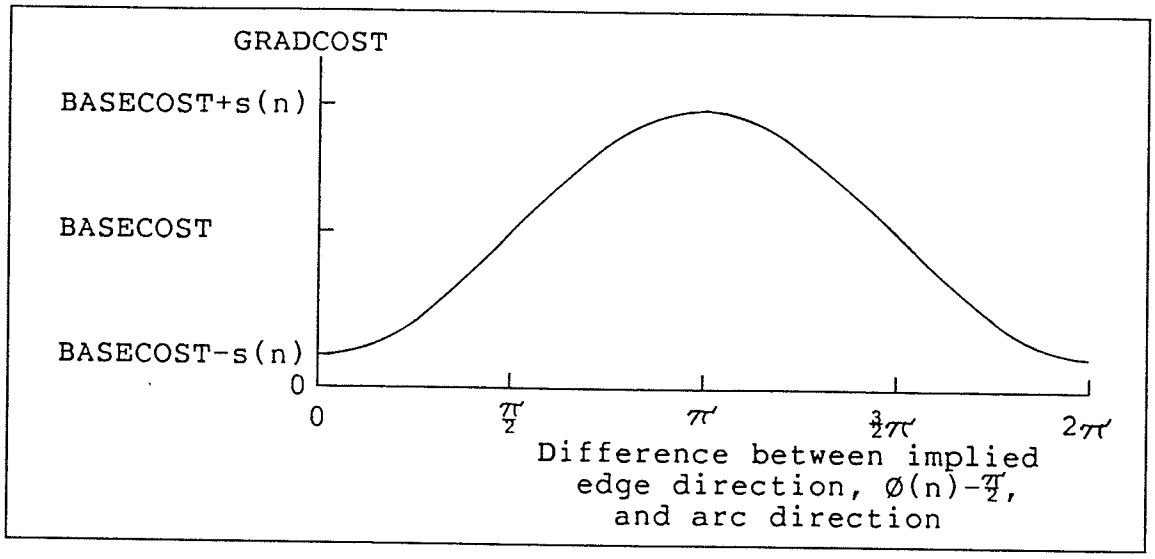


Figure 3.21: Operation of Improved GRADCOST

Figure 3.22 and Figure 3.23 show the contour edge images produced by the elliptical-object detector operating on JUMBLE1 and JUMBLE2 respectively. The expected object length entered by the operator was 75 for JUMBLE1 and 70 for JUMBLE2.

Both contour images demonstrated the superiority of this detector over any of the previous object perception techniques. The analysis of JUMBLE1 produced only one notable flaw. Of the pair of touching objects whose contact point was totally obscured by shadow, part of the end of one was ignored while the contour of the other was slightly mangled near the contact point. This was not unexpected since even a human would have difficulty accurately drawing the contours of these two objects near their contact point. Of the

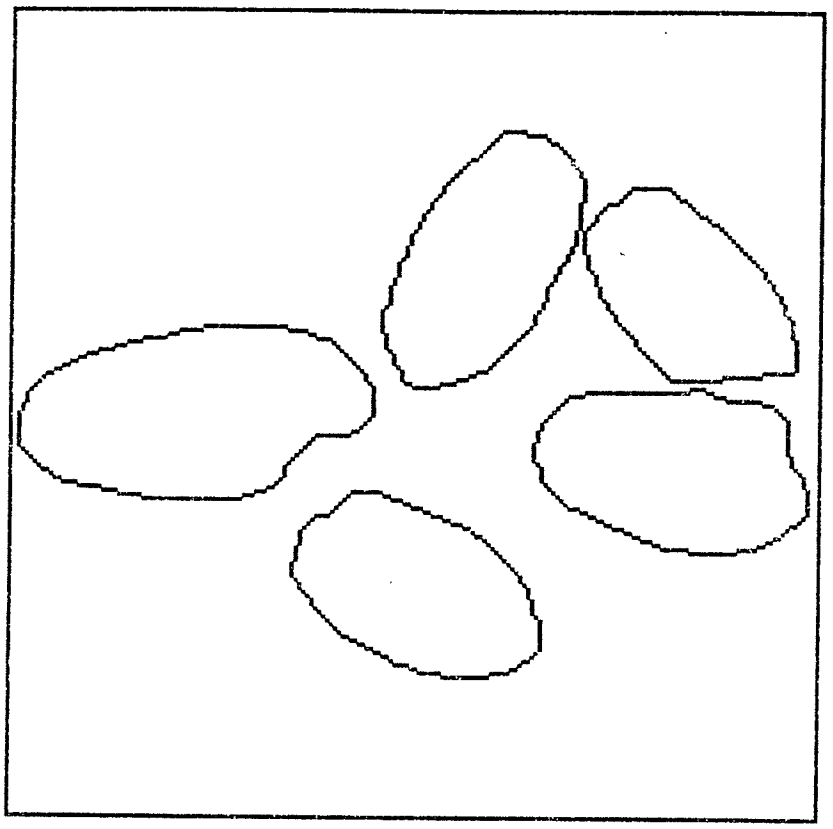


Figure 3.22: Contour Image of JUMBLE1 from E-O Detector

five objects in this image, three were satisfactorily detected and two had slight errors in their declared contours. On JUMBLE2 no notable flaws were produced; each object was satisfactorily detected. Surprisingly both contour images displayed less of the fragmented edge quality that had been typical of the regular-sized-object detector's declared contour edges. This was perhaps due to the improved cost function.

The number of node expansions required by the elliptical-object detector indicated its more efficient and consistent

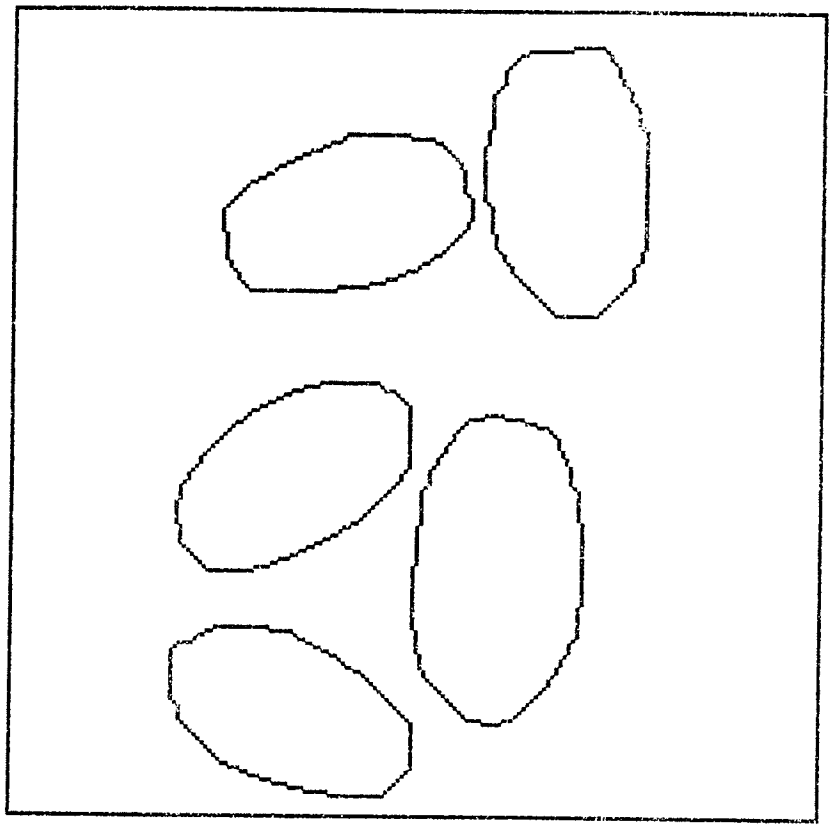


Figure 3.23: Contour Image of JUMBLE2 from E-O Detector

edge search. The average number of expansions performed in each edge search conducted on JUMBLE1 and JUMBLE2 was 2427. The minimum was 1709 for the upper left object in JUMBLE2 and the maximum was 3213 for the lower right object in the same image. These values were lower than for the regular-sized-object and hybrid detectors. This implied less search effort had been required. Their reduced variability suggested less search diversion by false edges.

In conclusion the use of contextual image information and of a priori knowledge of the specific object perception

problem were maximized in the elliptical-object detector to produce a dynamic, reliable and efficient object perception technique. Indeed further constraint of the technique would probably have had a detrimental effect on its performance. This detector produced satisfactory results with all of the wheat grading images encountered during this research. However if a wider range of object types had been of interest, for instance wheat and rape seed in the same image, then a less specific approach, such as the regular-sized-object detector, would have been required.

The reliability of the elliptical-object detector made it suitable for use in an automated system. For images satisfying the limitations of this research the detector did not require human interaction to verify its output; the detector always produced good results even with scenes of closely spaced or almost touching objects.

However this detector was not a final solution for scenes of touching objects (although for such images it did prevent drastic errors from being made). For more complicated scenes, perhaps also involving occluded objects, an even more intelligent approach would be required. Such an approach, similar to the human visual system, would likely treat an image as a two-dimensional representation of a three-dimensional world, match regions in the image to three-dimensional models of known objects and from such insights knowledgeably infer object identity, contours and orientation.

3.6 SHAPE DESCRIPTION

The first and most important group of features extracted from a wheat grading image are those that describe the size and shape of an object. As discussed in Section 1.4 shape features of an object outline are necessary for distinguishing between wheat kernels and other cereal grains or other foreign objects, they are useful in detecting surface defects such as breakage and shrivelling, and they are important in determining variety. In addition, the shape can indicate an object's orientation and thereby direct succeeding routines which must analyze only a particular subset of an object's visible surface.

The method chosen for describing the shape of an object outline was moments. For a binary discrete image, the p,q moment is defined as:

$$M_{pq} = \sum_{(m,n)\in R} m^p n^q$$

where (m,n) is a point in the object region R. Once one of the object perception programs had determined the contour of each object in an image, the program MOMENTS could then determine shape features based on the moments of each detected object. This program used as input the object's contour specified in Freeman chain code.

The moments method of shape analysis offered several advantages. Moments provided a unique description of a shape

and provided more accuracy in that description when more of them were considered. They were simple to calculate. They were easily made to be invariant to translation, rotation and size. The invariance calculations involved only moment values and not the original data.

The discrete Green's theorem, as developed by Tang [1981], was used to calculate the Mpq moments of a region. This theorem enabled the calculation of a moment by summing over an object's contour instead of the object's region. Consequently the number of computations was proportional to the length of the region's boundary instead of to its area.

In the discrete Green's theorem, R is a discrete 8-connected region without holes in the subspace:

$$S' = \{(h,k),\ h \geq 0,\ k \geq 0,\ h \text{ and } k \text{ are integers}\}.$$

R has more than one lattice point. The sequential boundary of R is B represented by the Freeman chain code as $((r_0,c_0),a_0,a_1, \ldots ,a_{\ell-1})$ such that the region is seen on the right as one moves along the boundary. The moment of R is then:

$$M_{pq} = \sum_{i=0}^{l-1} F_c(r_i,c_i)\, D(a_{r_{i-1}},a_i) + f(r_i,c_i)\, C(a_{r_{i-1}},a_i)$$

where (r_i, c_i) is a sequential boundary point,

$$F_c(m,n) = \sum_{i=0}^{n} i^q m^p,$$

$$f(i,n) = i^p n^q,$$

and D_r and C_r are defined as:

$D_r(a_{i-1}, a_i)$

a_{i-1} \ a_i	0	1	2	3	4	5	6	7
0	0	1	1	1	1	0	0	0
1	0	1	1	1	1	0	0	0
2	0	1	1	1	1	0	0	0
3	0	1	1	1	1	0	0	0
4	-1	0	0	0	0	-1	-1	-1
5	-1	0	0	0	0	-1	-1	-1
6	-1	0	0	0	0	-1	-1	-1
7	-1	0	0	0	0	-1	-1	-1

$C_r(a_{i-1}, a_i)$

a_{i-1} \ a_i	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	1	0	0
2	0	0	0	0	0	1	1	0
3	0	0	0	0	0	1	1	1
4	1	0	0	0	0	1	1	1
5	1	1	0	0	0	1	1	1
6	1	1	1	0	0	1	1	1
7	1	1	1	1	0	1	1	1

Once the moments of a region R had been determined with the discrete Green's theorem, MOMENTS calculated and printed several preliminary results before the normalization of the moments was undertaken. First, MOMENTS calculated the centroid of R as $(M_{10}/M_{00}, M_{01}/M_{00})$. The "principal centroidal moments of inertia", which were the second order moments about the principal axes of the object, were then calculated as:

$$\max(\alpha,\beta) \quad \text{and} \quad \min(\alpha,\beta)$$

where α and β are eigenvalues of the matrix:

$$\begin{bmatrix} C_{02} & C_{11} \\ C_{11} & C_{20} \end{bmatrix}$$

where $$C_{20} = \frac{M_{20}}{M_{00}} - \frac{M_{10}^2}{M_{00}^2} ,$$

$$C_{02} = \frac{M_{02}}{M_{00}} - \frac{M_{01}^2}{M_{00}^2} ,$$

and $$C_{11} = \frac{M_{11}}{M_{00}} - \frac{M_{01}M_{10}}{M_{00}M_{00}} .$$

The aspect ratio of these moments:

$$A = \frac{\max(\alpha,\beta)}{\min(\alpha,\beta)}$$

was then printed. These moments were a means of describing the size and aspect ratio of any general object shape although they did not provide the real length, width and aspect ratio. Last, the area of R, M_{00}, was printed.

MOMENTS normalized the original moments with respect to translation, rotation and size using methods described by Reeves and Rostampour [1981]. To normalize with respect to translation MOMENTS calculated the central moments from the original moments by:

$$\mu_{pq} = \sum_{u=0}^{p} \sum_{v=0}^{q} C_u^p C_v^q (-\bar{r})^u (-\bar{c})^v M_{p-u,q-v}$$

where $$C_u^p = \frac{p!}{u!(p-u)!} ,$$

$$\bar{r} = \frac{M_{10}}{M_{00}} ,$$

and $$\bar{c} = \frac{M_{01}}{M_{00}} .$$

Then the angle Θ of the principal axes was calculated by:

$$\tan 2\Theta = \frac{2\mu_{11}}{\mu_{02}-\mu_{20}} .$$

This equation specified four possible values of Θ. The value which satisfied $\emptyset_{02} > \emptyset_{20}$ and $\emptyset_{30} > 0$ was chosen. The rotation normalized moments were then calculated as:

$$\emptyset_{pq} = \sum_{u=0}^{p} \sum_{v=0}^{q} (-1)^{p-u} C_u^p C_v^q (\cos\Theta)^{q-u+v} (\sin\Theta)^{p-v+u} \mu_{u+v,q-u+p-v}$$

At this stage the orientation and position of the principal axes were known, so MOMENTS determined and printed the coordinates of the two object contour points nearest the principal major axis. The locations of these two points later guided the search for the kernel crease by a succeeding program. Figure 3.24 shows the geometry of rotation normaliza-

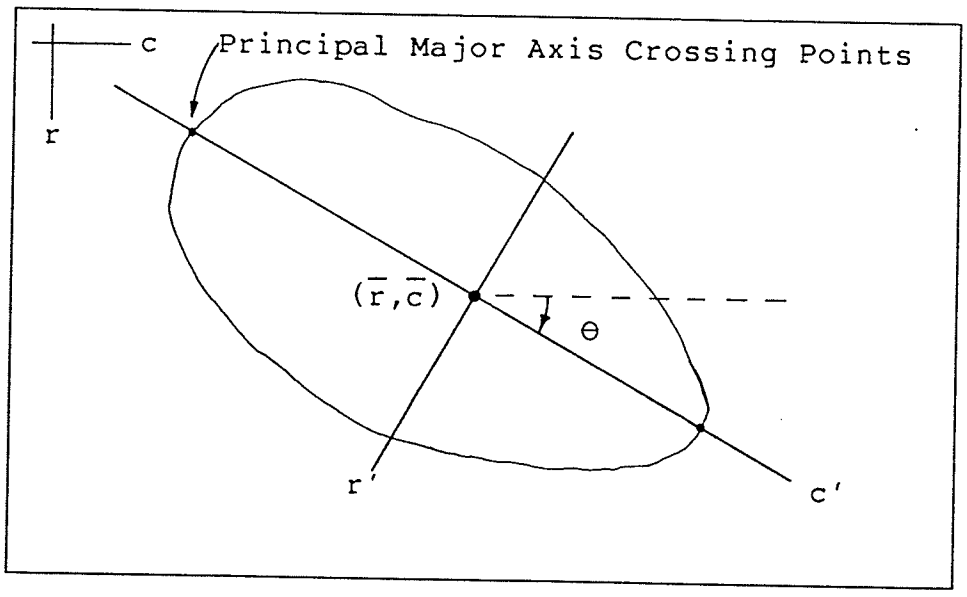


Figure 3.24: Rotation Normalization

tion. Finally MOMENTS normalized the moments with respect to size by using:

$$N_{pq} = \emptyset_{pq} / \emptyset_{00}^{\gamma}$$

where $\gamma = \frac{1}{2}(p+q)+1$.

This normalized set of moments, called "standard moments", was characterized by the following values:

$N_{00} = 1$ (normalized size),

$N_{01} = N_{10} = 0$ (normalized location),

and $N_{11} = 0$ (normalized orientation).

MOMENTS printed only the second and third order standard moments and excluded the above four values. Higher order moments could easily have been calculated with the above procedures. However the value of high order moments is limited since the higher the order of a moment, the more it is susceptible to digital sampling errors near the edge of an object.

The final calculations performed by MOMENTS on each object estimated its length and width as:

$$L = 4\sqrt{N_{02}M_{00}} \quad \text{and} \quad W = \frac{1}{\pi}\sqrt{M_{00}/N_{02}} \ .$$

These estimates were correct if the object contour was elliptical. Since wheat kernels are generally elliptical in shape, these estimates were normally very close to the actual values. These estimates had the additional advantage

that they ignored the small irregularities in a kernel contour caused by a rough surface and digital sampling error. The aspect ratio, L/W, and eccentricity, $\sqrt{1-W^2/L^2}$ were also calculated and printed.

The output from MOMENTS into the output data file included only the object region centroid, the two principal major axis crossing points, the object orientation, Θ, and the Freeman chain code of the object contour. This data was later used by the crease detection program for initially estimating the location of a crease. More of the results, such as the standard moments, could have been included in the output file. However they were not required since the task of classifying objects based on the generated shape features was not pursued in this research.

Figure 3.25 on pages 138 and 139 shows the printed output produced by MOMENTS processing the object contours found in JUMBLE2 by the elliptical-object detector. All of the calculated values of length and width assuming an elliptical object were very close to the actual values measured in the original image.

```
                                    MOMENTS PROGRAM OUTPUT


*** INPUT IMAGE NAME ***

JUMBLE2


*** OBJECT NUMBER    1 ***

CENTROID 175.295,  69.512

PRINCIPAL CENTROIDAL MOMENTS OF INERTIA
          MAXIMUM        275.2
          MINIMUM         83.5
          ASPECT RATIO   3.295

AREA    1901.
ORIENTATION OF PRINCIPAL AXES    24.321 DEGREES
PRINCIPAL MAJOR AXIS CROSSING POINTS    162, 40    AND    189,100

TRANSLATION, ROTATION AND SIZE NORMALIZED MOMENTS (2ND AND 3RD ORDER)
          N(2,0)    .14474E+00
          N(0,2)    .43923E 01
          N(3,0)    .25933E 02
          N(2,1)    .74673E-05
          N(1,2)    .70923E 03
          N(0,3)    .21147E 03

SIZE, ASSUMING AN ELLIPTICAL OBJECT
          LENGTH          66.4
          WIDTH           36.5
          ECCENTRICITY   .8353
          ASPECT RATIO   1.819

*** OBJECT NUMBER    2 ***

CENTROID 139.038,120.975

PRINCIPAL CENTROIDAL MOMENTS OF INERTIA
          MAXIMUM        386.2
          MINIMUM        119.6
          ASPECT RATIO   3.230

AREA    2696.
ORIENTATION OF PRINCIPAL AXES   273.439 DEGREES
PRINCIPAL MAJOR AXIS CROSSING POINTS    178,119    AND    102,123

TRANSLATION, ROTATION AND SIZE NORMALIZED MOMENTS (2ND AND 3RD ORDER)
          N(2,0)    .14325E+00
          N(0,2)    .44348E-01
          N(3,0)    .23480E-03
          N(2,1)   -.51781E-03
          N(1,2)   -.67550E-04
          N(0,3)    .22507E-03

SIZE, ASSUMING AN ELLIPTICAL OBJECT
          LENGTH          78.6
          WIDTH           43.7
          ECCENTRICITY   .8315
          ASPECT RATIO   1.800

*** OBJECT NUMBER    3 ***

CENTROID  51.184,  83.109

PRINCIPAL CENTROIDAL MOMENTS OF INERTIA
          MAXIMUM        268.0
          MINIMUM         89.0
          ASPECT RATIO   3.009

AREA    1934.
ORIENTATION OF PRINCIPAL AXES   164.328 DEGREES
PRINCIPAL MAJOR AXIS CROSSING POINTS     43,112    AND     60, 53

TRANSLATION, ROTATION AND SIZE NORMALIZED MOMENTS (2ND AND 3RD ORDER)
          N(2,0)    .13857E+00
          N(0,2)    .46043E-01
          N(3,0)    .23210E-02
          N(2,1)   -.40229E-03
          N(1,2)   -.77910E-03
          N(0,3)    .12639E-03

SIZE, ASSUMING AN ELLIPTICAL OBJECT
          LENGTH          65.5
          WIDTH           37.6
          ECCENTRICITY   .8187
          ASPECT RATIO   1.741
```

Figure 3.25: Output from MOMENTS Processing JUMBLE2

```
*** OBJECT NUMBER    4 ***

CENTROID  41.769,137.658

PRINCIPAL CENTROIDAL MOMENTS OF INERTIA
          MAXIMUM        303.5
          MINIMUM        108.4
          ASPECT RATIO   2.799

AREA    2271.
ORIENTATION OF PRINCIPAL AXES    85.149 DEGREES
PRINCIPAL MAJOR AXIS CROSSING POINTS     10,135    AND     76,141

TRANSLATION, ROTATION AND SIZE NORMALIZED MOMENTS (2ND AND 3RD ORDER)
          N(2,0)    .13363E+00
          N(0,2)    .47745E 01
          N(3,0)    .11253E 02
          N(2,1)   -.10661E 02
          N(1,2)    .59680E 03
          N(0,3)    .81515E-03

SIZE, ASSUMING AN ELLIPTICAL OBJECT
          LENGTH          69.7
          WIDTH           41.5
          ECCENTRICITY   .8033
          ASPECT RATIO   1.679

*** OBJECT NUMBER    5 ***

CENTROID 116.065, 70.289

PRINCIPAL CENTROIDAL MOMENTS OF INERTIA
          MAXIMUM        279.4
          MINIMUM         89.8
          ASPECT RATIO   3.112

AREA    1988.
ORIENTATION OF PRINCIPAL AXES   147.875 DEGREES
PRINCIPAL MAJOR AXIS CROSSING POINTS     98, 98    AND    134, 43

TRANSLATION, ROTATION AND SIZE NORMALIZED MOMENTS (2ND AND 3RD ORDER)
          N(2,0)    .14055E+00
          N(0,2)    .45174E-01
          N(3,0)    .49141E 03
          N(2,1)   -.71222E-03
          N(1,2)    .21482E 03
          N(0,3)    .25805E 03

SIZE, ASSUMING AN ELLIPTICAL OBJECT
          LENGTH          66.9
          WIDTH           37.9
          ECCENTRICITY   .8243
          ASPECT RATIO   1.766
```

Figure 3.25 Continued

Chapter IV

KERNEL ANATOMY DETECTION

4.1 INTRODUCTION

After the computerized image analysis system has perceived all the objects in a wheat grading image and obtained shape features for each of them, the machine is left only with data describing the location and shape of each object's contour. It was assumed in this research that some form of pattern classification analysis will then utilize the shape data to decide which object regions correspond to unbroken wheat kernels. Only these objects would be scrutinized for the grading factors, variety, vitreousness and soundness, since each of the remaining objects would be classed as foreign material. However, before the machine can look for these factors on each kernel, it must first determine which side of each kernel is visible (dorsal or ventral) and locate the anatomical parts present on that side which are useful for grading. This task was accomplished by attempting to detect only two major parts of the kernel: the crease and the germ.

Sections 1.3 and 1.4 presented the reasons for wishing to detect the crease and the germ. The detection of either of these parts is not by itself useful for grading. Rather,

the knowledge of the existence of a crease on a kernel and of the location of either a crease or germ can guide texture or shape analysis of the visible parts of the kernel that are important for grading. The crease and germ are themselves important areas in which to find cues for the condition of the kernel. They also imply the location of other important parts, especially the cheeks and the brush (see Figure 1.2).

The anatomy detection analysis was performed by two routines. The first attempted to detect a crease on each object in the image. The second routine attempted to find a germ on only those objects on which a crease was not found. Crease detection was performed first since the crease is the most distinctive anatomical part which appears on only one side of the kernel, the ventral side. Its presence or absence on the visible side indicates the kernel's orientation, ventral or dorsal side up respectively. Germ detection was only attempted on those kernels not displaying a crease since the germ is visible only on the dorsal side. In addition this routine searched for the germ only on the bulbous end of the kernel because this is usually the end on which the germ is located (see Figure 1.1). The brush is located opposite the germ on the more pointed end of the kernel.

Both kernel anatomy detection routines specified the contour of each detected anatomical part with a version of the Freeman chain code. This code was described in Section 3.1.

Two unprocessed images acted as subjects in this chapter with which to demonstrate the performance of the anatomy detection routines. These images, CREASES1 and CREASES3, appear in Figure 4.1 and Figure 4.2 respectively. CREASES1 contained four wheat kernels, all of which were ventral side up. The top left kernel was shrivelled. The left two kernels had distinct creases while the right two had poorly defined creases. This image was intended to demonstrate crease detection. CREASES3 also contained four kernels. However the left two kernels were ventral side up while the right two were dorsal side up. This image was intended to demonstrate both crease and germ detection.

Section 4.2 presents the crease detection technique and Section 4.3 presents the germ detection technique. Much of the underlying theory for both of these detectors was presented in Subsection 3.3.2.

Figure 4.1: CREASES1

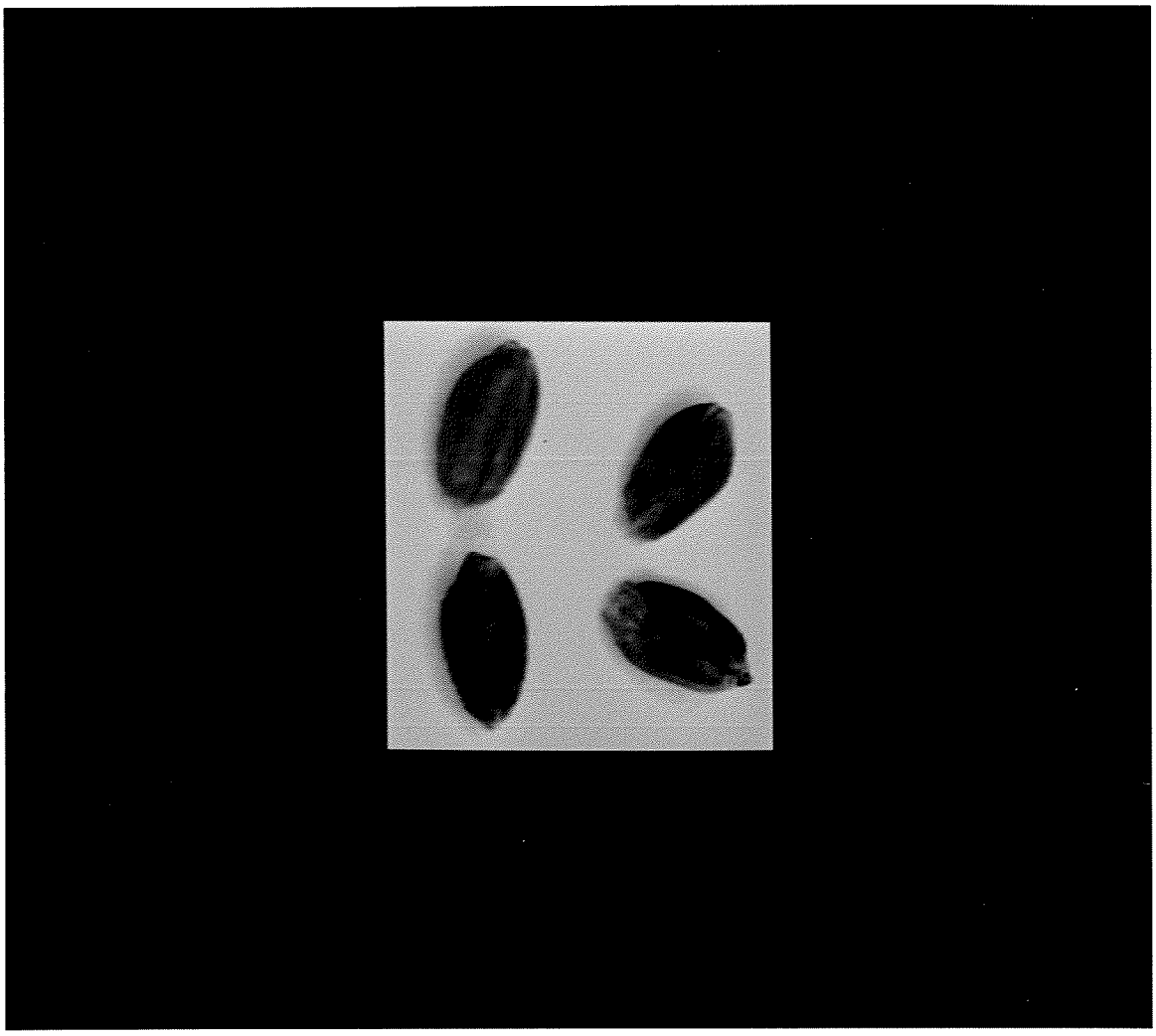
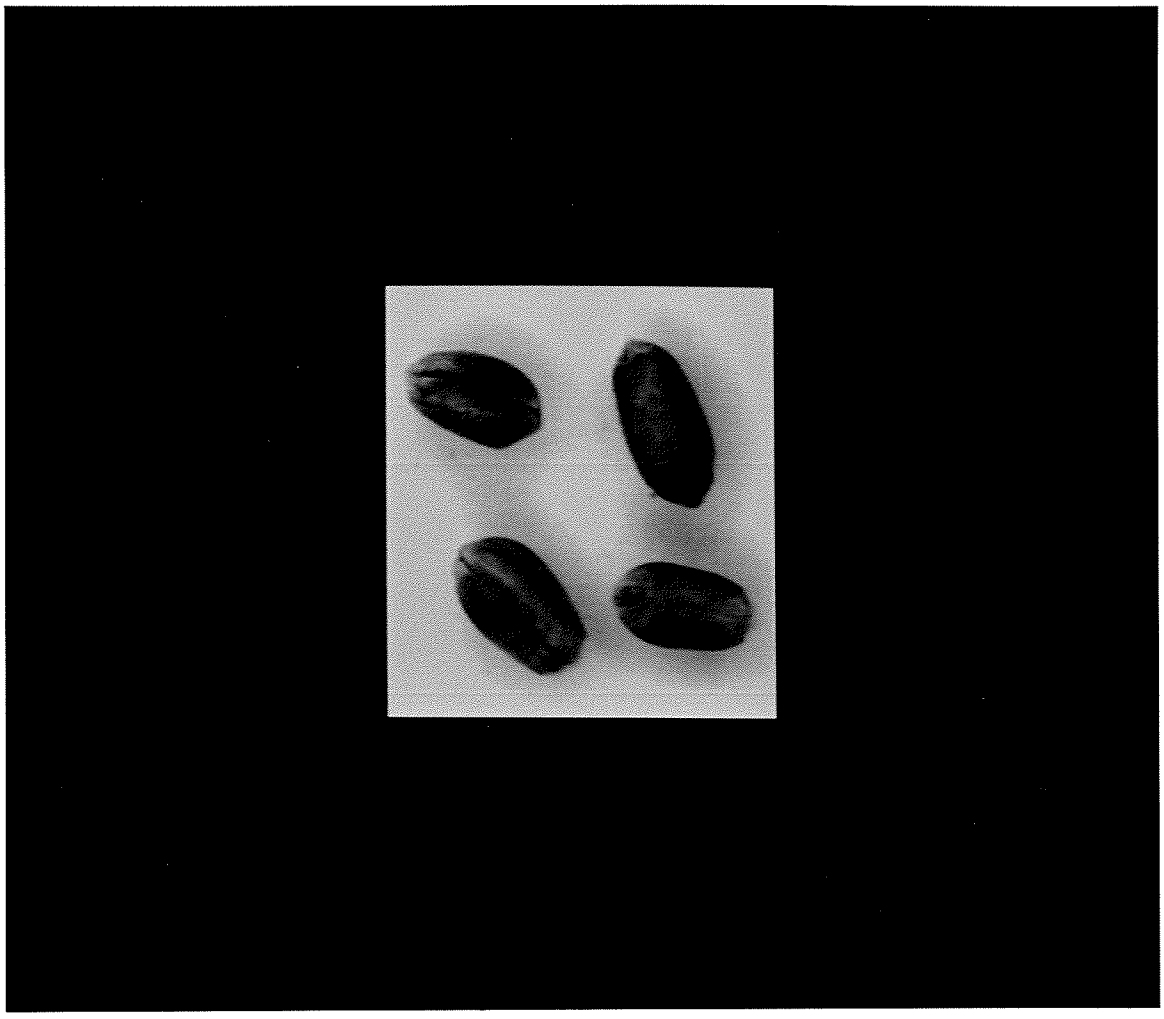


Figure 4.2: CREASES3



4.2 CREASE DETECTOR

The design of the crease detector was based on three characteristics of wheat kernel creases. First, the typical crease is darker than the surrounding kernel surface. Since it is an indentation, it therefore reflects less of the illuminating light. Second, this change in reflectivity near the crease often causes a large change in gray level perpendicular to the direction of the crease. Thus the crease frequently has the characteristics of a step edge. This is understandable since the crease is an edge separating the kernel cheeks. Usually the crease is the most prominent edge feature on the kernel surface although the effects of shadow sometimes cause false edges on the kernel which are just as prominent. Finally, the crease runs the full length of the kernel along its longitudinal, or in other words, principal major axis.

4.2.1 Principles of Operation

The crease detector operated by finding and specifying the single best edge running longitudinally through each kernel image region. This problem was transformed into a graph search by treating the kernel image region as a graph in a way similar to that of the regular-sized-object detector. This graph was in registration with the object region of the image so that each pixel within or on the kernel contour corresponded to a graph node. An edge was represented by a path comprised of nodes and interconnecting arcs in the

graph. The idea of the "best" edge was quantitatively defined by a node evaluation or cost function like that of the regular-sized-object detector.

The cost function of the crease detector embodied the expected characteristics of the crease. This function, which determined the cost for a path to incorporate the node n, was composed of two subordinate costs as follows:

$$c(n) = \begin{cases} c'(n), & c'(n) \geq 0 \\ 0, & \text{otherwise} \end{cases}$$

where $c'(n) = \text{DADJUST}(n) \cdot [\alpha \cdot \text{GRADCOST}(n) + \beta \cdot \text{PIXCOST}(n)]$.

The coefficients α and β were entered by the operator to control the influence of GRADCOST(n) and PIXCOST(n). Normally they were both set to 1. The diagonal adjustment, DADJUST(n), was the same as that in the regular-sized-object detector.

The cost of the gradient, GRADCOST(n), evaluated the likelihood that n represented a pixel on an edge running through n in the direction of the longitudinal axis of the kernel and in the same direction as the arc leading to n. Thus GRADCOST(n) measured the edge strength at n and gauged the correspondence of the edge direction implied by the gradient direction with the principal major axis direction and

with the arc direction. The expression for the gradient cost was:

$$\text{GRADCOST}(n) = \text{BASECOST} - s(n) \cdot \left|\cos\left[\phi(n) - \tfrac{\pi}{2} - \theta\right]\right| \cdot \cos\left\{2\left[\phi(n) - \tfrac{\pi}{2} - \text{arc direction}\right]\right\}$$

where $s(n)$ was the Sobel gradient magnitude at n,
$\phi(n)$ was the Sobel gradient direction at n,
"arc direction" was the angle of the arc leading to n,
and θ was the orientation of the principal major axis.

The normal value of BASECOST was 250. This function was the same as the GRADCOST'(n) of the regular-sized-object detector except for the inclusion of the term comparing $\phi(n)$ with θ. This term could only reduce the cost if the edge direction implied by the gradient, $\phi(n)-\frac{\pi}{2}$, was similar to the kernel orientation, θ. Figure 4.3 shows the effect of this term on GRADCOST(n).

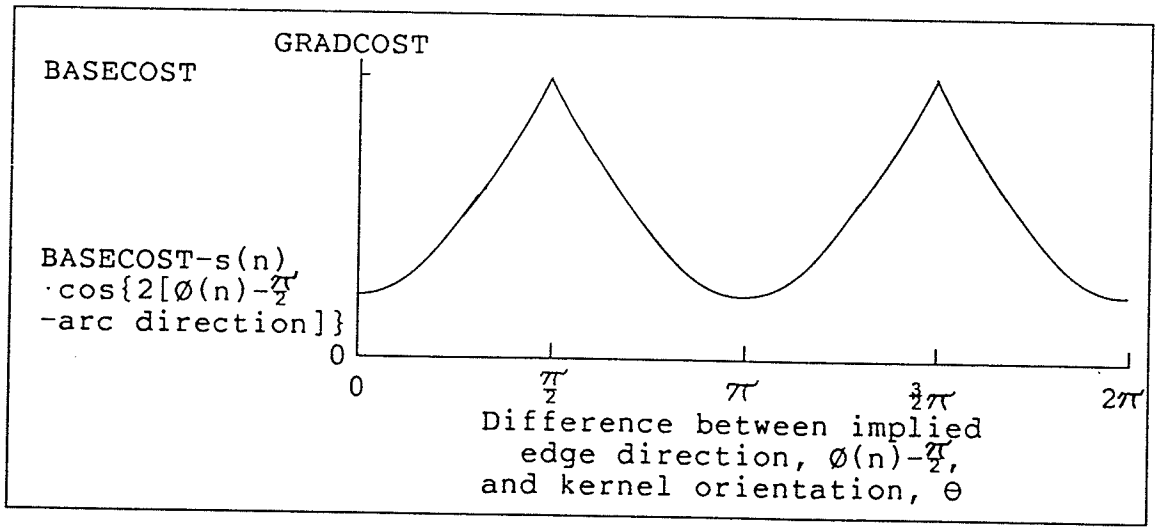


Figure 4.3: Effect of Kernel Orientation Term on GRADCOST

The cost of the pixel value, PIXCOST(n), favoured dark pixels. The expression for the pixel value cost was simply:

$$PIXCOST(n) = (BASECOST/128) \cdot [p(n)-MINPIX]$$

where p(n) was the pixel value at n and
MINPIX was the minimum pixel value within the kernel region.

The coefficient, BASECOST/128, normalized PIXCOST(n) with respect to GRADCOST(n) so that each cost had the same effect when α equalled β.

Only three arc directions were allowed in the graph used for the crease detector. These three contiguous directions were centred on the "major lattice direction". This direction was that Freeman chain code direction which was closest to the kernel's principal major axis orientation, θ. The set of allowed arc directions ensured that the minimum angle available on either side of the major axis direction was $\frac{\pi}{8}$ and that the start and goal nodes were connected. In addition they were the only arc directions which could possibly be on a path representing an authentic crease.

The graph had only a single start node and a single goal node. These nodes corresponded to the principal major axis crossing points found by the program MOMENTS on the kernel contour. These locations made the edge search progress along the longitudinal axis of the kernel region where the crease was expected to run. Figure 4.4 shows the form of a typical graph used in the crease detector.

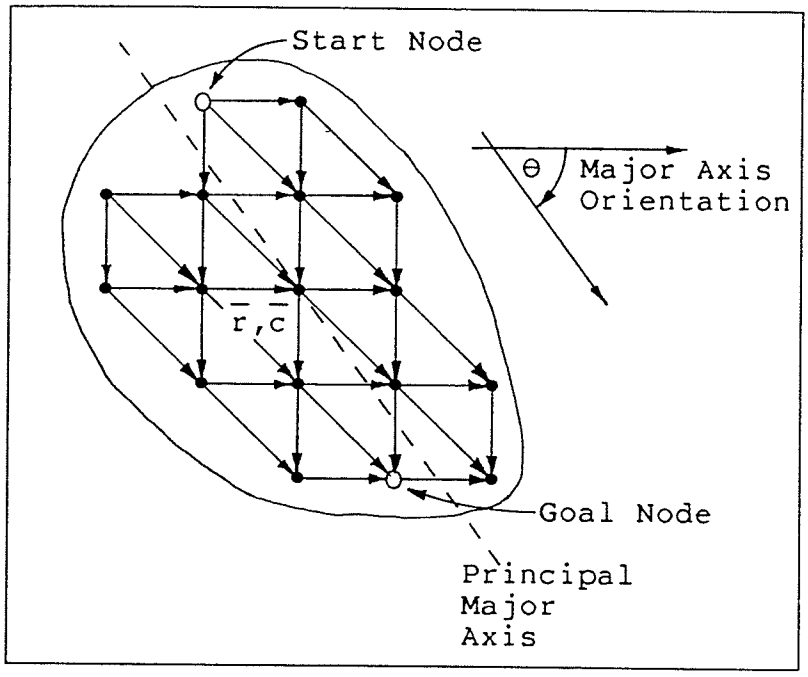


Figure 4.4: Typical Crease Detector Graph

The array GRAPH represented the crease detector graph. The bit definitions in GRAPH were the same as those used in the regular-sized-object detector except that a flag bit for the goal node was neither required nor defined. Figure 4.5 shows the bit definitions of GRAPH.

The graph search algorithm employed in the crease detector to find the least cost path had two major differences from the graph search algorithm of the regular-sized-object detector. First, it did not have the mechanisms installed for handling combined start/goal nodes since the start and goal nodes were separate nodes in the crease detector. Second, it did not use heuristic information to improve the

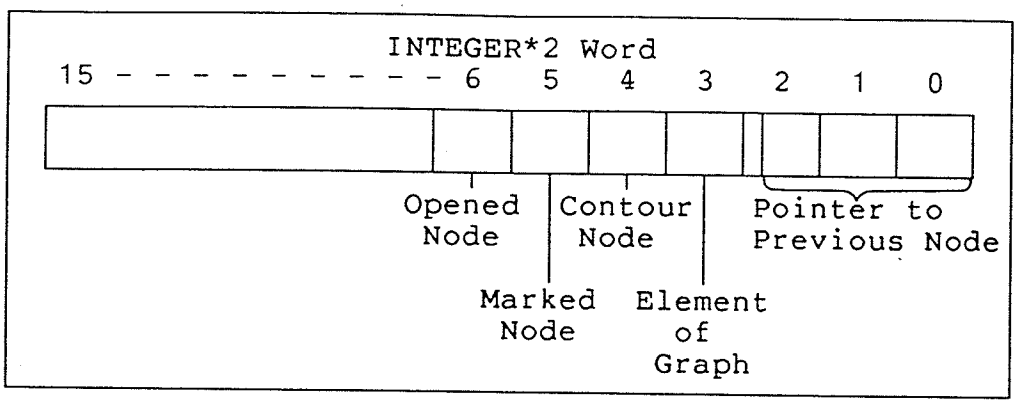


Figure 4.5: Bit Definitions of GRAPH

speed of the search. A typical graph for the crease detector was much smaller than its counterpart for the regular-sized-object detector so that speed was not as important. Also, it was found that obtaining the single least cost solution was important for obtaining acceptable results. Therefore a heuristic could not be used since it would have introduced suboptimality into the search. Indeed the combination of no heuristic and the allowed arc directions guaranteed not only that solely the least cost solution was found, but also that the search would never end in failure. Sharp turns again were not allowed in the solution path.

The form of the Algorithm A* used for the crease detector was as follows:

1. Place the start node's coordinates in OPEN and set its cost **f**=0.
2. Find the node n in OPEN with the lowest value of **f** and mark it closed by resetting its opened-node bit.

3. If n is the goal node then the least cost path has been found so exit the algorithm; otherwise continue.

4. Expand n by generating its successors, each of which must:

   a) be an element of the graph,

   b) be reachable by one of the three arcs centred on the major lattice direction, and

   c) satisfy the no sharp turns rule.

5. For each successor node not already marked calculate **f**, set its opened-node and marked-node bits and place its coordinates in OPEN.

6. For each successor node already marked calculate its new value of **f** but store only a temporary version of it. If the new value of **f** is not less than the old value, then omit this successor. Otherwise replace the old value of **f** with its new value, redirect the successor's pointer to n and if it is currently marked closed, mark it open and place its coordinates in OPEN.

7. Go to 2.

A major difference between the crease detector and the regular-sized-object detector was that not only did the crease detector have to find the least cost path in a graph, as did the regular-sized-object detector, but also it had to decide whether this path represented an actual edge, the crease. This decision was based on two conditions.

The first condition was that the number of solution path nodes following the kernel contour had to be below a threshold entered by the operator. If a fairly well defined crease existed in a kernel region then the solution path would follow it. In contrast, following the contour from one end of the kernel to the other was more expensive since a longer

path was required and pixels on the contour generally had higher gray levels. If a well defined crease did not exist, such as for a dorsal side view, the solution path often followed part of the kernel contour since it was the only well defined edge. The standard setting for this threshold was 5.

The second condition was that the average cost per node on the solution path had to be below a threshold entered by the operator. A solution path that followed a well defined crease had less average cost per node than one following a poorly defined or nonexistent crease. The standard setting for this threshold was about 200.

4.2.2 Performance

Figure 4.6, Figure 4.7 and Figure 4.8 show the crease images outputted by the crease detector operating on CREASES1. The kernel outlines were previously determined by the elliptical-object detector. Pixels on a contour were set to 255. The defined crease paths were then found by the crease detector. Pixels on a defined crease were set to 128. Thus points representing contours and creases in these figures can be easily differentiated since those representing defined creases are smaller than those representing kernel contours. The thresholds entered by the operator, for crease nodes on contour and average cost per node, were set in effect to infinity so that a crease was always declared and

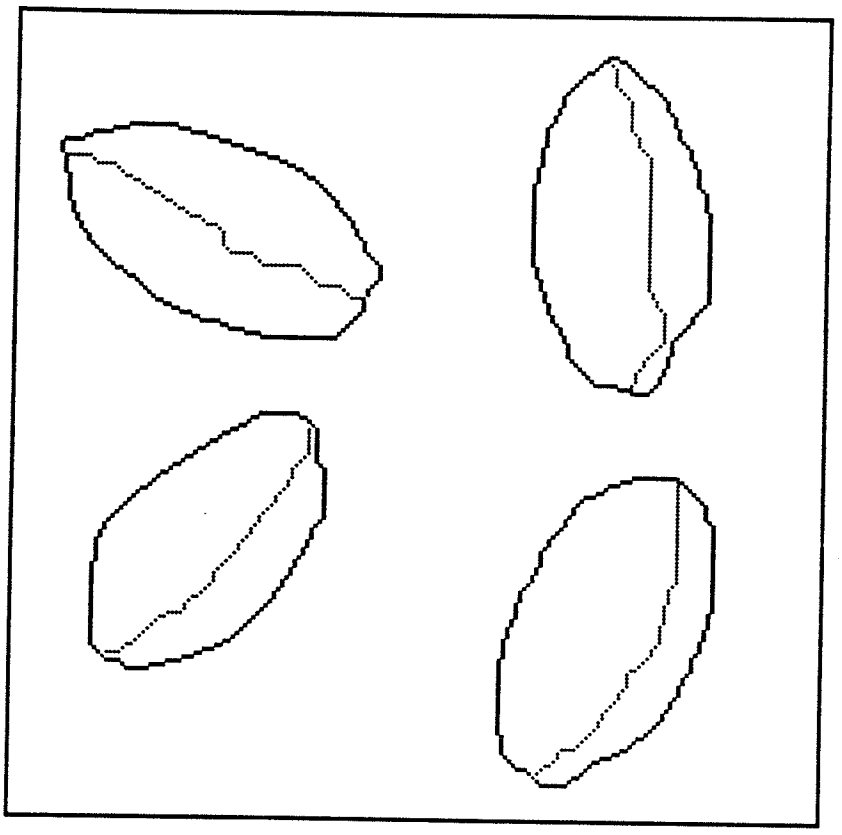


Figure 4.6: Crease Image of CREASES1 With Pixel Value Cost Only

therefore the solution path was always displayed in the figure.

Figure 4.6 and Figure 4.7 demonstrate the results produced by using only the pixel value cost and only the gradient cost respectively. With the pixel value cost only, each edge search found a solution path that traversed just the darker areas of the kernel region. This path was just as likely to follow shadows on the kernel surface as it was to follow the crease. With the gradient cost only, each solution path followed, within limits imposed by the allowed arc

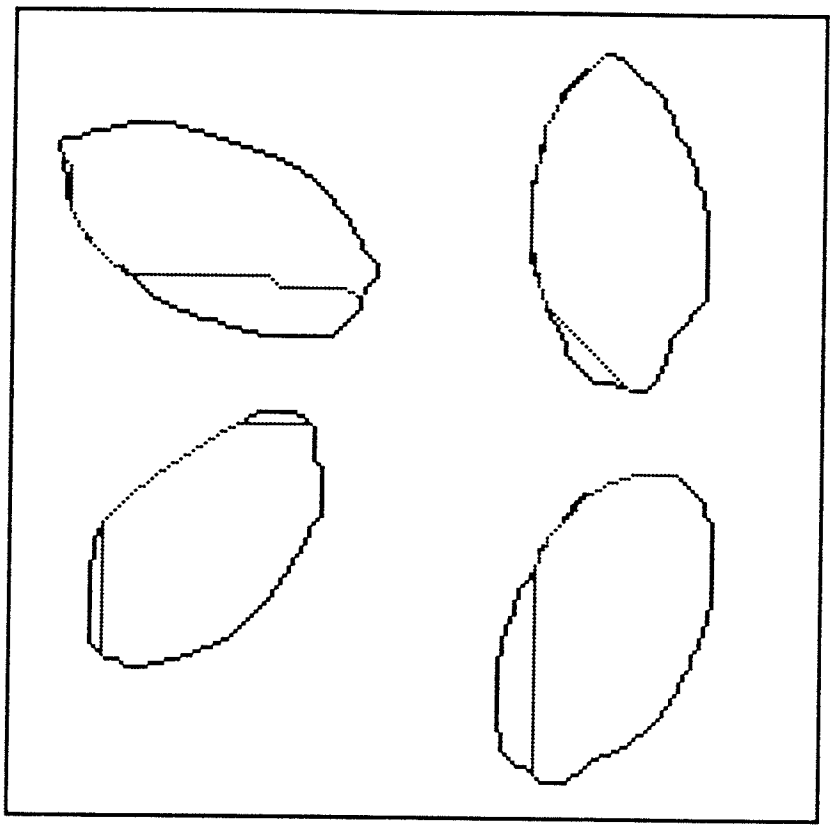


Figure 4.7: Crease Image of CREASES1 With Gradient Cost Only

directions, the most well defined edge in the kernel image region: namely the kernel contour.

Figure 4.8 shows the results produced by using the standard settings of α and β so that the pixel value cost and gradient cost had equal influence on the solution path. The crease detector accurately specified the well defined creases of the two kernels on the left side. The average cost per node on the two solution paths was 168 for the upper kernel and 200 for the lower kernel. However the detector incorrectly specified the poorly defined creases of the two

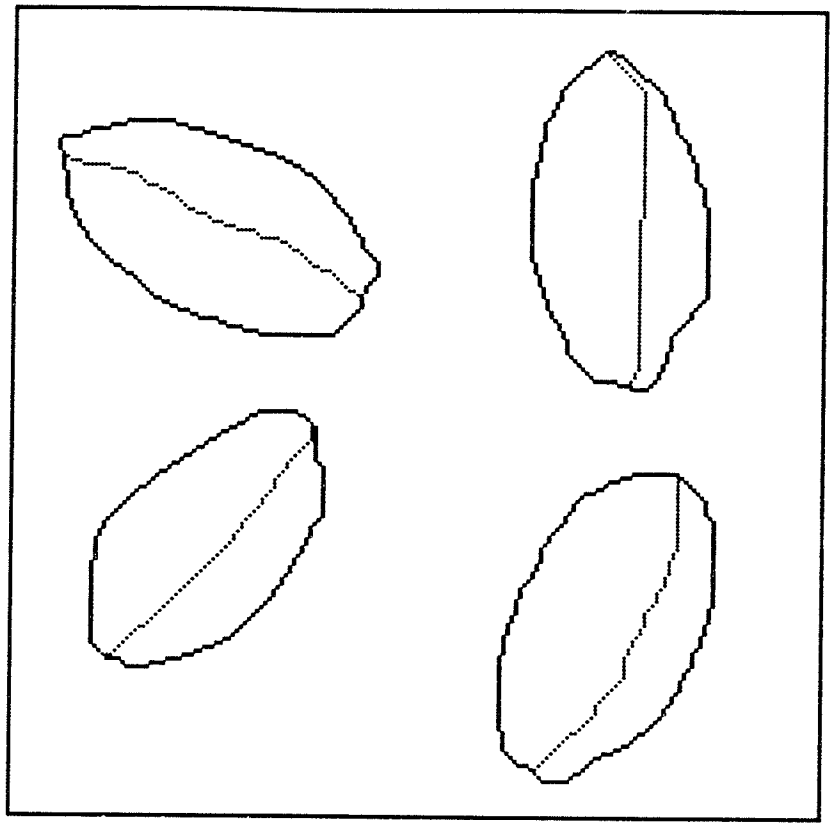


Figure 4.8: Crease Image of CREASES1 With Equal Weights

kernels on the right side. Instead the solution paths followed false edges created by shadow. The average cost per node was 225 for the upper kernel and 233 for the lower kernel. On average 91 percent of the total graph nodes were expanded in each graph search. This high percentage illustrated the difficulty encountered by the typical search in estimating the location of the crease.

Figure 4.9 and Figure 4.10 show the crease images produced by the crease detector operating on CREASES3 with equal values of α and β. For Figure 4.9 the thresholds en-

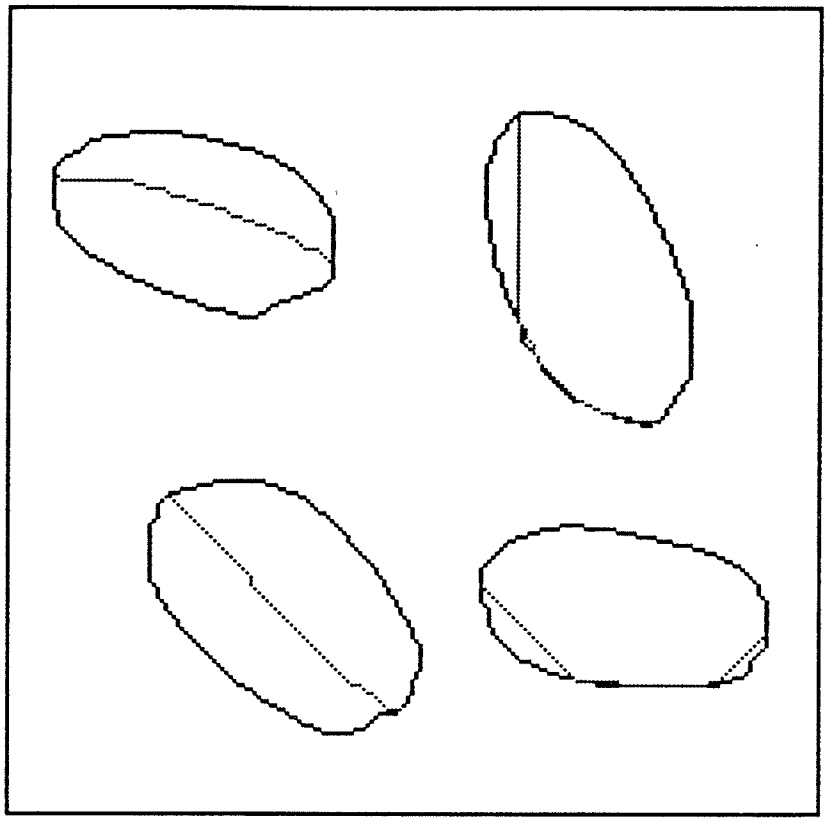


Figure 4.9: Crease Image of CREASES3 Using Infinite Thresholds

tered by the operator were effectively infinite while for Figure 4.10 the thresholds were the standard values. The detector accurately specified the crease of the upper left kernel. But it made a small error specifying the location of the crease on the lower left kernel. The upper half of the solution path for this kernel was somewhat above the actual crease position. The average cost per node for each of these two solution paths was 167 and 187 respectively. Since each of the right two kernel regions in CREASES3 did not have a crease, the edge search in that region followed a

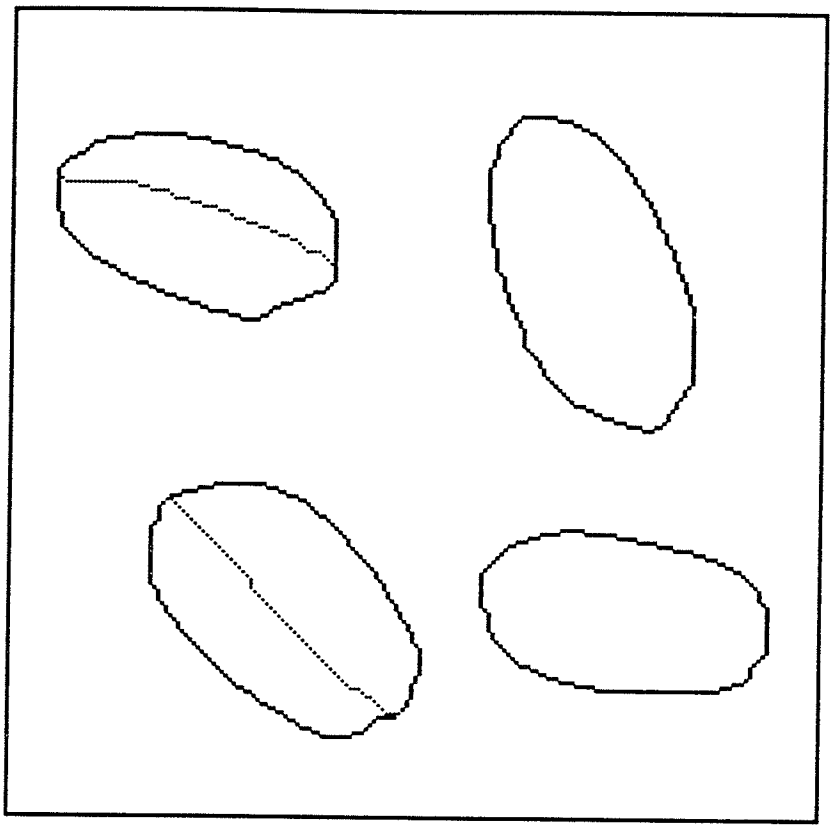


Figure 4.10: Crease Image of CREASES3 Using Standard Thresholds

considerable length of the kernel contour. Thus the solution path for each of these kernels exceeded the crease nodes on contour threshold in the second run of the crease detector. Hence Figure 4.10 does not show defined crease locations in the right two kernel regions.

The above five figures demonstrate the shortcomings of the crease detector. Although the detector was usually able to detect and specify well defined creases, it was rarely able to detect poorly defined creases. When a crease was poorly defined, the detector was easily deceived by the ef-

fects of shadow. Another fault with the detector was its questionable ability to differentiate between a solution path representing a poorly defined crease and one representing a false edge. For the limited number of test images employed in this research the crease detector had modest success at distinguishing between valid and invalid solution paths. However the small change in the average cost per node encountered between typical valid and invalid solution paths made clear the uncertainty of the crease decision method.

The mediocre performance of the crease detector suggested that a better detector was desirable. In fact many modifications of the crease detector were tested before the current version was arrived at. These included a different set of allowed arc directions, several different crease decision methods, multiple start and goal nodes, artificial limitation of the edge search away from the kernel contour and six different cost functions. It was deduced that the essential problem with the crease detector was that it was a two-dimensional solution to a three-dimensional problem. Indeed considering this handicap, the performance of the crease detector was surprisingly good.

It is instructive to consider the human vision system when assessing the performance of the crease detector. The human vision system can easily detect the kernel creases in CREASES1 and CREASES3 because it employs the knowledge that

these images are representations of three-dimensional solids and that the crease is a manifestation of the kernel's surface topography. A machine that would improve on the crease detector by taking advantage of this approach would probably use local surface gray level to estimate local surface orientation while generating a three-dimensional model of each kernel and finally determine a crease's presence and location based on this model, not on the image itself.

In conclusion the crease detector applied three known traits of the appearance of kernel creases in a graphical edge search to detect and specify the kernel crease. The performance of this detector was satisfactory for well defined creases. However it could not reliably detect poorly defined creases or discriminate between a solution path representing a poorly defined crease and one not representing a crease at all.

4.3 GERM DETECTOR

Several characteristics of wheat kernel germs guided the design of the germ detector. The germ appears only on the dorsal side of the kernel where the crease is not visible. Thus the germ detector did not attempt to find a germ on a kernel on which the crease detector had previously found a crease. The germ is usually located on the bulbous end of the kernel. Therefore the germ detector analyzed only the bulbous end of a kernel for the presence of a germ. This

end was previously determined by the program MOMENTS. The germ has an albedo and surface orientation which differ from those of the surrounding kernel surface. These changes occur abruptly at the periphery of the germ and thereby cause the appearance of a weak step edge which delineates the germ. The germ can appear either darker or lighter than the surrounding kernel surface. Finally, the typical germ outline roughly describes part of a circle whose centre is the major axis crossing point of the bulbous end of the kernel contour. The radius of this circle is about one-quarter of the length of the kernel.

Evidently the germ detector had an objective similar to that of the crease detector. The germ detector first had to find and specify the single best edge which possibly represented an anatomical feature on part of the visible kernel surface. Then it had to decide whether the detected edge actually did represent the desired feature. Consequently the germ detector employed a very similar technique to that of the crease detector. The only major differences between the germ detector and crease detector were the graph location, the cost function, the allowed arc directions and the number of start and goal nodes.

The following subsection discusses, for the sake of brevity, only the significant differences between the germ detector and crease detector.

4.3.1 Principles of Operation

The graph of the germ detector represented and was in registration with the bulbous half of the kernel image region. This graph was demarcated by the principal minor axis and half of the kernel contour. Figure 4.11 shows the form of the germ detector graph.

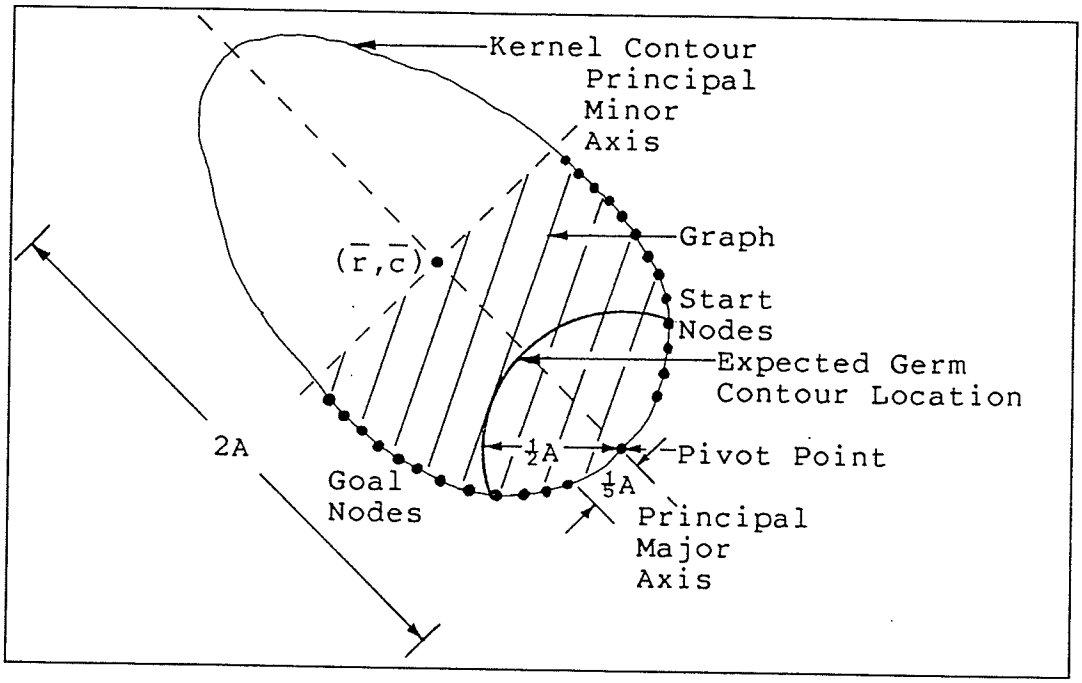


Figure 4.11: Typical Germ Detector Graph

The graph had multiple start and goal nodes. The start nodes were positioned on the kernel contour on the right side of the principal major axis crossing point. The goal nodes were positioned on the kernel contour on the left side of the principal major axis crossing point. Thus the edge search was rotary in nature about the principal major axis

crossing point. This point was referred to as the pivot point. The start nodes and the goal nodes were not positioned closer than $\frac{1}{5}A$ to the pivot point, where 2A was the kernel length. This restriction ensured that the solution path did not begin or end nearer to the pivot point than expected for the typical germ outline.

The array GRAPH represented the germ detector graph. The bit definitions in GRAPH were equivalent to those of the regular-sized-object detector. Figure 3.12 shows these bit definitions.

The arcs allowed in the graph were determined in exactly the same fashion as those in the regular-sized-object detector since the desired features of the germ contour were similar to those of the kernel contour. Each arc had to be one of the eight Freeman chain code elements, have a direction which was one of the five contiguous directions centred on the preferred direction and satisfy the no sharp turns rule. Figure 3.11 shows the arc selection process.

The cost function of the germ detector was composed of three parts as follows:

$$c(n) = \text{DADJUST}(n) \cdot \text{RADJUST}(n) \cdot \text{GRADCOST}(n) \ .$$

The diagonal adjustment, DADJUST(n), was identical to that of the regular-sized-object detector.

The cost of the gradient, GRADCOST(n), was the same in every way to that of the regular-sized-object detector except that BASECOST had the lower value of 75. This reduced BASECOST provided more sensitivity for the edge search with which to detect the weak step edge that delineated the typical germ.

The radial adjustment, RADJUST(n), neutralized the low-cost advantage of short paths near the pivot point. This adjustment performed the same function as that of the regular-sized-object detector, but its form was considerably different. The derivation of RADJUST(n) was based on the elliptical model of the kernel contour shape and the circular arc model of the germ contour shape. For a circle of radius R centred on a vertex of an ellipse with an aspect ratio of 2 and length of 2A, the length of the arc of the circle within the ellipse is:

$$L = 2R \cdot \arccos\left[-\tfrac{1}{3}(A/R)+\tfrac{1}{3}\sqrt{(A/R)^2+12}\right] .$$

A close approximation to L for $R/A < 1.3$ is given by:

$$\hat{L}/A = \begin{cases} a(R/A)^2+b(R/A), & (R/A) < 0.8 \\ 1, & \text{otherwise} \end{cases}$$

where $a = -1.71$, and

$b = 2.58$.

This approximation, which is normalized with respect to the size of the ellipse, required considerably less computation

to calculate than the exact expression for L. The radial adjustment was then:

$$\text{RADJUST}(n) = \hat{(L/A)}^{-1} .$$

Figure 4.12 shows the geometry of the derivation of RADJUST(n).

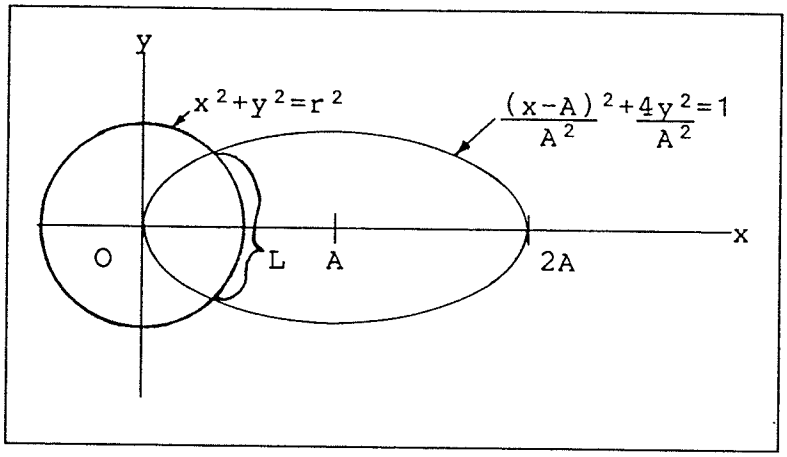


Figure 4.12: Derivation of RADJUST(n)

4.3.2 Performance

Figure 4.13 shows the germ image created by the germ detector program when it processed CREASES3. The kernel outlines and crease paths had been previously defined by the elliptical-object detector and crease detector respectively. Pixels on kernel contours were assigned the value 255 and those on crease paths and germ contours were assigned the value 128. This image demonstrated the germ detector's remarkable ability to detect and follow the very subtle edge which out-

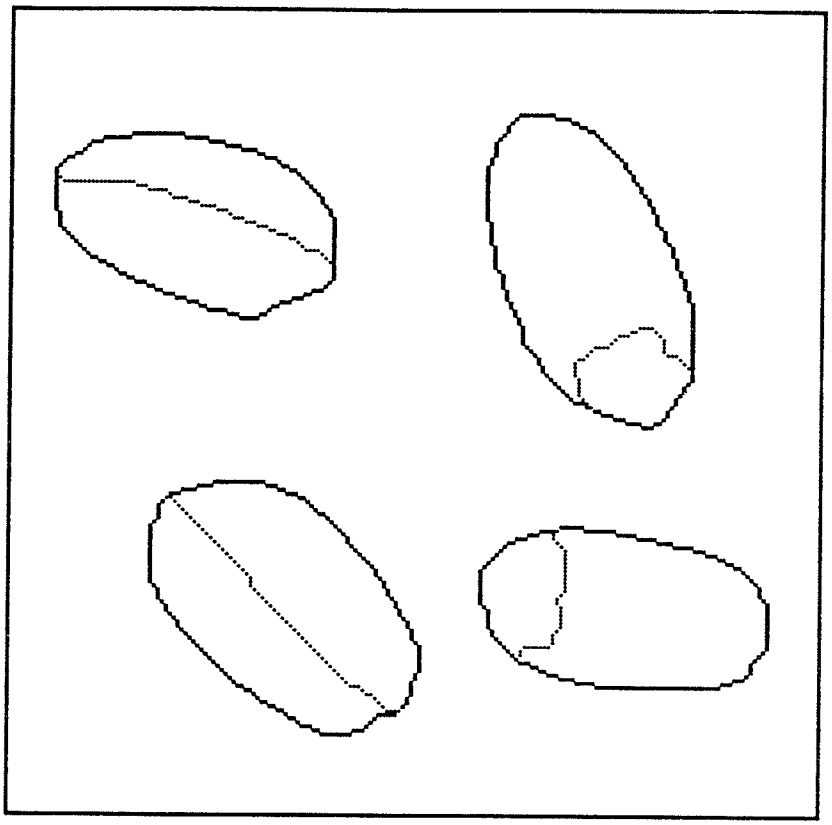


Figure 4.13: Germ Contour Image of CREASES3

lined the germ in each of the right two kernel image regions.

A definitive statement concerning the performance of the germ detector can not be made because the detector was not tested on a sufficient number of images. However, since both detectors utilized similar techniques, the germ detector likely had problems similar to those of the crease detector. Consequently the germ detector probably suffered from the rigidity of its two-dimensional approach to the germ detection problem, thereby being truly successful only on images of at least moderately well defined germs.

Chapter V

TEXTURE ANALYSIS

5.1 INTRODUCTION

As discussed in Section 1.4, surface texture is a broad class of features which is important for wheat grading. Texture is the major characteristic in describing the two grading factors, vitreousness and soundness. It is also useful for determining variety which is another grading factor. Indeed texture is second only to shape in importance for wheat grading.

The areas of interest on the visible kernel surface for which texture features are valuable include not only the entire visible kernel surface, but also several subsets of this region: specifically the germ and the crease. The envisioned computerized grading system would obtain texture features for each of these regions of interest in each wheat kernel image region after it had first identified the location of each whole wheat kernel and its visible anatomical parts.

In this research a texture analysis technique was implemented and employed just for obtaining texture features of the overall image region of a wheat kernel. The implementa-

tion of a technique to analyze specific subsets of the kernel region was left for future research.

The choice of the texture analysis technique was based on several desired and tractable capabilities of the technique. The technique had to distinguish between micro-texture (fine) and macro-texture (coarse) components of a texture. Since these distinctions correspond to spatial frequency, then some form of frequency transformation was required. The technique had to characterize the amount of energy in a texture or, in other words, the gray tone variation between pixels. Finally, a desired, but not essential, capability was texture synthesis. This facility would allow visualization of a texture represented by texture features and indicate that in some sense the analysis technique could capture everything about a texture.

The selected texture analysis technique, two-dimensional (2-D) autoregression (AR) modelling, was chosen from the wide variety of approaches that are applied to the texture analysis problem today. Haralick [1979] gives a broad accounting of most of these. The advantages of 2-D AR modelling which motivated its use are as follows:

- AR modelling is sensitive to spatial pixel dependence while at the same time being able to account for the stochastic nature of a random texture image.
- The AR model parameter equations are linear and therefore have a straight forward solution.
- The estimated AR model can be easily manipulated to generate its power density spectrum (PDS).

- AR modelling is a superior technique for resolving spectrum peaks in one-dimensional (1-D) signal analysis and so the same can probably be said for 2-D AR image analysis. This superiority is related to the fact that AR modelling is, as pointed out by Makhoul [1975], identical to the very successful maximum entropy spectral estimation technique.
- AR modelling has a complimentary texture image synthesis facility.

AR modelling provided two types of features with which to describe a texture. The first, the AR model parameters, characterized the spatial dependence of pixel values. A stepwise linear discriminant analysis was applied to the AR parameters of images of three classes of wheat kernel surfaces in order to gauge the discriminant power of these parameters. The results were disheartening and are described in Subsection 5.6.2. The second type, samples from the AR model's power density spectrum which approximated that of the texture, however held far more promise for successful discrimination. The approximate texture spectrum provided a quantitative assessment of the importance of the spatial frequency components of a texture in a range from coarse to fine texture. This characterization corresponded more closely to the visual description of the surface degrading factors of wheat kernels and, in addition, was much easier for the experimenter to relate to his own visual experience.

A problem arose in applying AR modelling to texture synthesis. Although some authors have used only a 1-D AR model, most authors agree, Haralick for instance, that a 2-D

AR model is necessary to fully represent a texture image. But, 2-D AR models do not have easily calculated stability criteria. Therefore, when a 2-D AR model is used for texture image synthesis, the experimenter, Gambotto [1980] for example, must go to great lengths to ensure that the model remains in a stable domain. This usually implies that model complexity is extremely limited.

The texture synthesis technique chosen for this research is due to Kashyap [1980] who has found a solution to stability problems in texture synthesis with 2-D AR models. Kashyap assumes that the image is folded into a torus. A torus image is folded so that its top edge joins its bottom edge and its left edge joins its right edge. Under this assumption AR texture image synthesis becomes a problem in the simultaneous solution of a set of linear equations. Thus, AR model instability becomes very unlikely, occurring only when the solution matrix is singular. If a Moore-Penrose inverse is used then the problem of model stability is totally eliminated. Chellappa and Kashyap [1981] refer to this approach as simultaneous autoregressive modelling or SAR.

Section 5.2 presents the technique employed to normalize the texture analysis with respect to rotation. Rotation normalization was necessary to remove the dependence of the texture analysis on wheat kernel orientation.

Sections 5.3, 5.4 and 5.5 present the AR model estimation, the PDS generation and the texture synthesis techniques respectively.

Section 5.6 demonstrates preliminary experimental results. Section 5.7 presents recommendations based on these results for the direction of future research into the use of the texture analysis technique for wheat grading.

5.2 ROTATION NORMALIZATION

The normalization of the estimated AR model with respect to rotation minimized the dependence of the model on the orientation of the kernel being analyzed. Thus the AR models of images of the same kernel rotated at different angles about an axis parallel to the camera axis would be almost identical provided that the same magnification and lighting were used. The results of a test like this are presented in Subsection 5.6.1.

The normalization was performed by copying the kernel region of the original image onto a transformed image whose column axis was parallel to the principal major axis of the kernel. The principal major axis had been previously determined by the program MOMENTS. Then the AR model was estimated using the transformed image so that the direction dependent model parameters always had the same orientation relative to the kernel orientation. This procedure was repeated for each kernel region in the original image.

Each pixel, P', in the transformed image, I', was assigned the value of the nearest pixel, P, in the original image, I. The centre of the transformed image corresponded to the centroid of the kernel image region. The size of the transformed image was 128 rows by 128 columns which easily accommodated typical kernel images at the standard magnification. The expression that defined each transformed image pixel value was then:

$$P'(i',j') = P(i,j)$$

such that $i = (j'-64)\cdot\sin\Theta + (i'-64)\cdot\cos\Theta + \bar{r}$

and $j = (j'-64)\cdot\cos\Theta - (i'-64)\cdot\sin\Theta + \bar{c}$

where (i,j) was the pixel location in I,
(i',j') was the pixel location in I',
(r,c) was the centroid of the kernel region, and
Θ was the orientation of the kernel.

Figure 5.1 shows the geometry of the rotation normalization method.

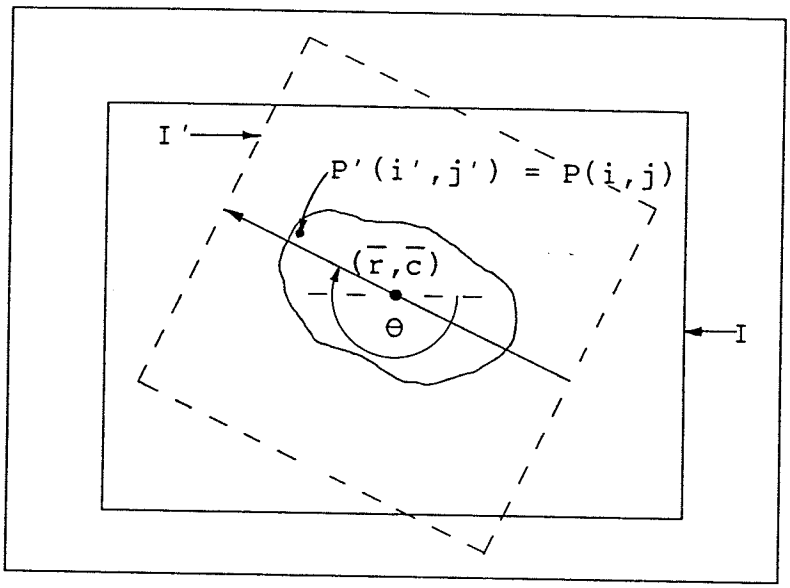


Figure 5.1: Geometry of Rotation Normalization

5.3 TEXTURE MODEL ESTIMATION

The texture of an arbitrarily shaped kernel image region was modelled as a 2-D AR system driven by the uncorrelated, zero mean, nonstationary and unity variance random process w(i,j) such that:

$$Y(i,j) = \sqrt{\rho}\, w(i,j) + \sum_{p,q \in N} \theta(p,q)\, Y(i+p,j+q)$$

where $Y(i,j) = P(i,j) - a$ is the zero mean pixel value,
$P(i,j)$ is the actual pixel value,
a is the average pixel value in the region,
ρ is the spatial input noise variance,
$\theta(p,q)$ is an AR coefficient, and
N is the influential neighborhood.

N was the set of coordinates of the estimated AR coefficients. N was called the influential neighborhood and Y(i,j) the dependent pixel of that neighborhood since Y(i,j)

was related to a linear combination of the other pixel values in the neighborhood. N was, for convenience, defined to be rectangular and lower-right causal such that:

$$N = \{i,j \mid i,j \neq 0,0;\ 0 \leq i \leq N_1-1;\ 0 \leq j \leq N_2-1\}.$$

Figure 5.2 illustrates the AR texture model.

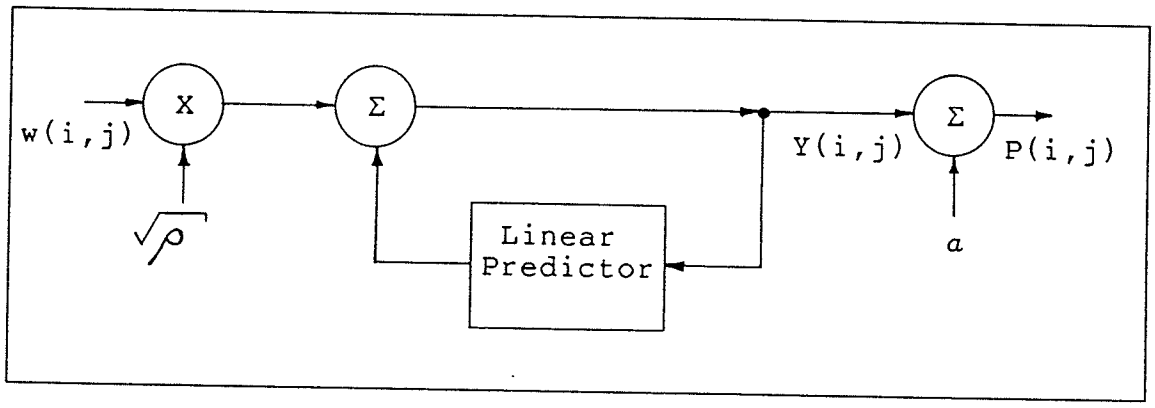


Figure 5.2: AR Texture Model

The method of solution for the AR coefficients is the well known method of least squares. The predicted value of the zero mean pixel value Y(i,j) was:

$$\hat{Y}(i,j) = \sum_{p,q \in N} \theta(p,q)\ Y(i+p,j+q)$$

and the resulting total squared error was:

$$E = \sum_{i,j \in R'} [Y(i,j)-\hat{Y}(i,j)]^2 = \rho \sum_{i,j \in R'} w^2(i,j)$$

where $R' = \{i,j \mid (i,j) \in R;\ (i+p,j+q) \in R \text{ for all } (p,q) \in N\}$, and R was the kernel image region.

R′ was the largest subset of R for which the influential neighborhood of each Y(i,j) was within R. By minimizing E with respect to θ(p,q), the "normal equations" which defined θ(p,q) were:

$$\sum_{p,q\in N} \theta(p,q) \sum_{i,j\in R'} Y(i+p,j+q)\, Y(i+u,j+v) = \sum_{i,j\in R'} Y(i,j)\, Y(i+u,j+v)$$

$$\text{for all } u,v\in N,$$

and the spatial input noise variance was:

$$\rho = \{ \sum_{i,j\in R'} [Y^2(i,j) - \sum_{p,q\in N} \theta(p,q)\, Y(i,j)\, Y(i+p,j+q)]\}/n$$

where n was the number of influential neighborhoods in R′. Thus the AR coefficients were found by the solution of N_1N_2-1 simultaneous linear equations with N_1N_2-1 unknowns.

The above normal equations had several properties that are worthy of note. First, they did not rely on the assumption that the texture's random process was stationary. Hence the AR model's portrayal of a texture was more dependable than that of one which assumed stationarity. Second, the terms

$$\sum_{i,j\in R'} Y(i+p,j+q)\, Y(i+u,j+v)$$

formed an image correlation matrix which was symmetric. The program TEXTURE, which performed both the rotation normali-

zation and texture model estimation, solved these equations with a Cholesky decomposition which took advantage of this symmetry. Finally, an increase in the number of AR coefficients, or equivalently in N, improved the model "fit" and thereby decreased ρ at the expense of more parameters to describe a texture.

A concern with AR modelling is model order determination: in this case the determination of the optimal influential neighborhood size N to represent the typical wheat kernel surface texture. Several objective criteria exist for model order determination. One is the Final Prediction Error (FPE) of Akaike. Deguchi and Morishita [1978] have applied the FPE to 2-D AR texture modelling for images. However the wheat grading problem requires only a rudimentary form of texture characterization and therefore only a small N such as 2 by 2 or 3 by 3. A small N like this would probably be much smaller than that indicated by the FPE. Consequently a more suitable approach would be a statistical analysis to determine the number of parameters required to differentiate surface class. Being beyond the scope of this research, this analysis was left for future research.

5.4 POWER DENSITY SPECTRUM

The PDS of the 2-D AR model approximated the actual texture spectrum.

The AR predictor model again was:

$$\sqrt{\rho}\, w(i,j) = Y(i,j) - \sum_{p,q \in N} \theta(p,q)\, Y(i+p,j+q) \; .$$

For convenience this was rewritten as:

$$\sqrt{\rho}\, w(i,j) = \sum_{p=0}^{N_1-1} \sum_{q=0}^{N_2-1} a(p,q)\, Y(i+p,j+q)$$

where: $$a(p,q) = \begin{cases} 1, & p=0 \text{ and } q=0 \\ -\theta(p,q), & \text{otherwise} \end{cases}$$

and N is assumed to be rectangular.

Taking the Z transform:

$$\sqrt{\rho}\, W(z_1,z_2) = Y(z_1,z_2) \sum_{p=0}^{N_1-1} \sum_{q=0}^{N_2-1} a(p,q)\, z_1^{p}\, z_2^{q} \; .$$

The system function of the AR model then was:

$$H(z_1,z_2) = \frac{Y(z_1,z_2)}{\sqrt{\rho}\, W(z_1,z_2)} = \frac{1}{\sum_{p=0}^{N_1-1} \sum_{q=0}^{N_2-1} a(p,q)\, z_1^{p}\, z_2^{q}} \; .$$

By considering only

$$z_1 = e^{j\omega_1} \quad \text{and} \quad z_2 = e^{j\omega_2}$$

the PDS of the AR model was:

$$\hat{P}(\omega_1,\omega_2) = \left| H(e^{-j\omega_1}, e^{-j\omega_2}) \right| = \frac{1}{\left| \sum_{p=0}^{N_1-1} \sum_{q=0}^{N_2-1} a(p,q)\, e^{-jp\omega_1}\, e^{-jq\omega_2} \right|^2}$$

The following discrete version of the above expression was used with equal summation limits to generate the square digital image representing the texture PDS:

$$\hat{P}(k,l) = \frac{1}{\left| \sum_{p=0}^{M-1} \sum_{q=0}^{M-1} a(p,q)\, e^{-j\frac{2\pi pk}{M}}\, e^{-j\frac{2\pi ql}{M}} \right|^2}$$

where $M \geq N_1$ and $M \geq N_2$,

$$a(p,q) = \begin{cases} 1, & p=0 \text{ and } q=0 \\ -\theta(p,q), & p \leq N_1-1 \text{ and } q \leq N_2-1 \\ 0, & \text{otherwise,} \end{cases}$$

k and l are integers such that $0 \leq k,l \leq M-1$, and

$k = \frac{\omega_1 M}{2}$ and $l = \frac{\omega_2 M}{2}$.

The above spectral estimate was more simply expressed as:

$$\hat{P}(k,l) = \frac{1}{|A(k,l)|^2}$$

where $A(k,l)$ is the 2-D discrete Fourier transform (DFT) of $a(p,q)$. The PDS image becomes more detailed as the size of the influential neighborhood, and therefore the number of AR parameters, increases since, in the limit,

$$\hat{P}(k,l) = P(k,l) \text{ as the size of } N \longrightarrow \infty$$

where $P(k,l)$ is the actual texture PDS. The 1-D version of this is stated by Makhoul.

The program TXTRGEN, which calculated and created the PDS image, had several peculiarities. First, since this program also created a synthetic texture, it was more convenient for it to calculate the spectrum with the equivalent expression:

$$\hat{P}(k,l) = \left| \frac{1}{M^2 \cdot A^{-1}(k,l)} \right|^2$$

where $A^{-1}(k,l)$ was the inverse 2-D DFT of $a(p,q)$. The fraction

$$\frac{1}{M^2 \cdot A^{-1}(k,l)}$$

was also employed in the texture synthesis calculation. Second, the image width M was 128 so that a fast Fourier transform (FFT) algorithm could be used to calculate the inverse DFT. Third, the zero frequency pixel, where k=l=0, was placed in the centre of the image so that for the pixel value Y(i,j) the values of k and l were:

$$k = (i+64) \bmod 128 \quad \text{and} \quad l = (j+64) \bmod 128 \ .$$

Thus the distance from the centre of the image was proportional to the spatial frequency. Since in general the Fourier transform is conjugate symmetric, the PDS was also symmetric such that

$$\hat{P}(k,l) = \hat{P}(-k,-l)$$

so that the PDS image was symmetric about its centre or zero frequency point. Finally, to accommodate the wide dynamic

range of the detail in the PDS, TXTRGEN set each pixel to a value proportional to the logarithm of the corresponding PDS value.

5.5 TEXTURE SYNTHESIS

The synthesis of SAR texture images was simplified considerably by the torus image assumption. The zero mean pixel value in an M by M torus image obeying the 2-D AR model is given by:

$$Y(i,j) - \sum_{p,q \in N} \theta(p,q)\, Y[(i-p) \bmod M,\ (j-q) \bmod M] = \sqrt{\rho}\, w(i,j),$$

$$0 \le i,j \le M-1 \ .$$

Using the terminology of Chellappa and Kashyap, this set of linear equations can be written in matrix form as:

$$\mathbf{B}(\theta)\mathbf{Y} = \sqrt{\rho}\, \mathbf{w}$$

where $\mathbf{Y}$ and $\mathbf{w}$ are in lexicographic order. If $\mathbf{w}$ is in lexicographic order then:

$$\mathbf{w}^{T} = [\ w(0,0), w(0,1), \ . \ . \ . \ w(M-1,M-1)\ ] \ .$$

To avoid redundant influential neighbors, $\theta(i,j)$ is restricted to $0 \le i,j \le M-1$. This is called a one-sided or causal lower right neighborhood. Other one-sided neighborhoods are possible.

$\mathbf{B}(\theta)$ is obviously M^2 by M^2. If

$$\mathbf{B}(\theta) = [b(r,c)]_{M^2 \times M^2}$$

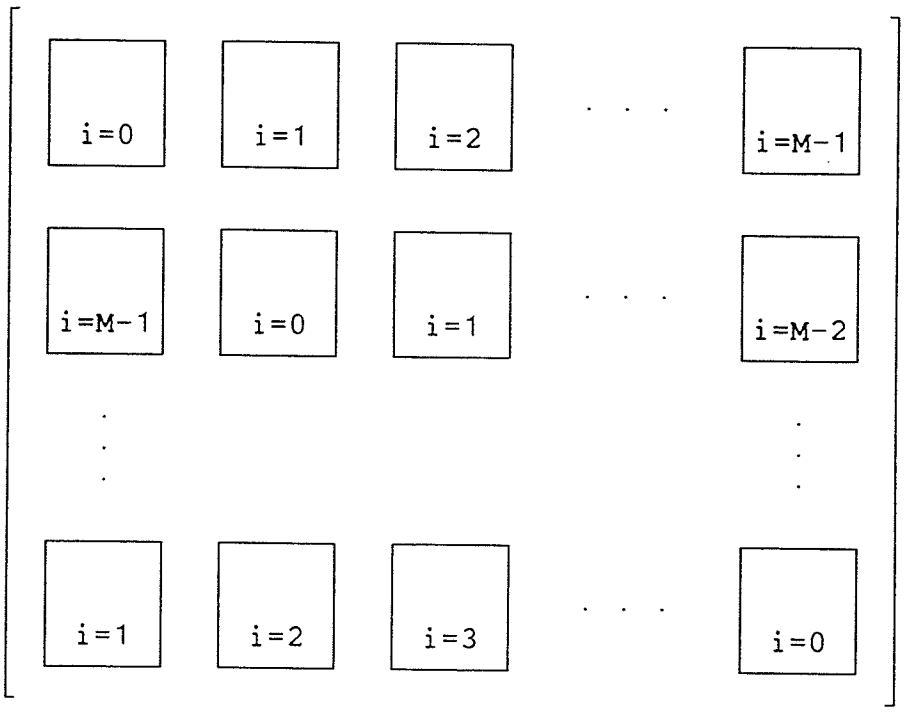
then it can be shown that:

$$b(r,c) = \begin{cases} -\theta(0,0)=1, & r=c \\ & \text{This is the dependent pixel of row r.} \\ -\theta(i,j), & (i,j) \in N \\ & \text{where} \\ & i = \{\text{Int}[(c-1)/M] - \text{Int}[(r-1)/M] + M\} \bmod M \\ & j = \{(c-1) \bmod M - (r-1) \bmod M + M\} \bmod M \\ 0, & (i,j) \notin N \end{cases}$$

These specifications for $\mathbf{B}(\theta)$ imply that it is an element of a unique set of matrices called "block circulant with circulant blocks" (BCCB), where each block of $\mathbf{B}(\theta)$ is M by M. The structure of each block is as follows:

$$\begin{bmatrix} \theta(i,0) & \theta(i,1) & \theta(i,2) & \cdots & \theta(i,M-1) \\ \theta(i,M-1) & \theta(i,0) & \theta(i,1) & \cdots & \theta(i,M-2) \\ \vdots & & & & \vdots \\ \theta(i,1) & \theta(i,2) & \theta(i,3) & \cdots & \theta(i,0) \end{bmatrix}$$

The arrangement of the blocks within $\mathbf{B}(\theta)$ is as follows:



The objective in synthesizing a torus AR texture is to solve the set of linear equations:

$$\mathbf{B}(\theta)\mathbf{Y} = \sqrt{\rho}\,\mathbf{w}$$

Davis [1978] has shown that all BCCB matrices are diagonalizable so that $\mathbf{B}(\theta)$ can be diagonalized as:

$$\mathbf{B}(\theta) = \mathbf{F}^{*}\Lambda\mathbf{F}$$

where

$$\mathbf{F} = \mathbf{F}_M \otimes \mathbf{F}_M ,$$

$\mathbf{F}_M$ is the M by M Fourier matrix,

$\otimes$ is the Kronecker product, and

Λ is the diagonal eigenvalue matrix.

Using this diagonalization Davis has shown that the Moore-Penrose inverse of $\mathbf{B}(\theta)$ is:

$$\mathbf{B}^{+}(\theta) = \mathbf{F}^{*\,+}\mathbf{F}$$

where

$$\Lambda^{+} = (\lambda^{+})_{M^2 \times M^2}$$

$$\lambda^{+} = \begin{cases} \frac{1}{\lambda}, & \lambda \neq 0 \\ 0, & \lambda = 0 \end{cases}$$

which presents the following solution to the synthesis problem:

$$\mathbf{Y} = \mathbf{B}^{+}(\theta)\mathbf{w} = \mathbf{F}^{*}\Lambda^{+}\mathbf{F}\mathbf{w} \ .$$

The eigenvalues of $\mathbf{B}(\theta)$ are given by:

$$\lambda(k,l) = \sum_{i=0}^{M-1} W_M^{-(k-1)i} \sum_{j=0}^{M-1} b(i+1,j+1)\, W_M^{-(l-1)j}$$

where $W_M = e^{-j(2\pi/M)}$,

$b(p,q)$ is element q of block p of $\mathbf{B}(\theta)$, and

$\lambda(k,l)$ is eigenvalue l of block k of $\mathbf{B}(\theta)$.

Substituting in for the elements of $\mathbf{B}(\theta)$ which are the SAR coefficients, the eigenvalue expressions become:

$$\lambda(k,l) = -\sum_{i=0}^{M-1} \sum_{j=0}^{M-1} W_M^{-(k-1)i-(l-1)j}\, \theta(i,j)$$

where $\theta(0,0) = -1$ as this corresponds to the dependent pix-

el. Obviously $\lambda(k,l)$ is expressed as a 2-D inverse DFT over the SAR coefficients $\theta(i,j)$.

By using the Moore-Penrose inverse of $\mathbf{B}(\theta)$ the difficulty of having a singular $\mathbf{B}(\theta)$ and therefore not having a standard inverse of $\mathbf{B}(\theta)$ is avoided. Thus the problem of instability of the 2-D AR texture model can be totally eliminated. The Moore-Penrose inverse has the property that if $|\mathbf{B}(\theta)| \neq 0$ then the standard inverse is employed. Otherwise the generalized inverse that minimizes $||\mathbf{Y}||_2$ is used. In terms of the synthetic image this is the solution that minimizes the image power or variance.

To complete the derivation of the texture synthesis technique, a solution to the product $\mathbf{Y} = \mathbf{F}^*\Lambda^+\mathbf{F}\mathbf{w}$ is found. Multiplication by $\mathbf{F}$ amounts to the application of a 2-D DFT while multiplication by $\mathbf{F}^*$ amounts to the application of a 2-D inverse DFT. Therefore the final expression specifying the synthetic SAR texture image can be directly written as:

$$P(i_1,i_2) = \sqrt{\rho}\,\{(1/M^2)\sum_{k_1=0}^{M-1}\sum_{k_2=0}^{M-1}\lambda^+_{k_1k_2}\,W_M^{-i_1k_1-i_2k_2}\left[\sum_{j_1=0}^{M-1}\sum_{j_2=0}^{M-1}w(j_1,j_2)\,W_M^{k_1j_1+k_2j_2}\right]\}+a1$$

where $\lambda^+(k_1,k_2)$ is an eigenvalue of the Moore-Penrose inverse of $\mathbf{B}(\theta)$,
a is the average pixel value,
1 is an M by M field of which each element is 1,
ρ is the variance of the spatial input noise, and
$P(i_1,i_2)$ is a pixel of the synthetic texture image.

The program TXTRGEN, which calculated and created a synthetic texture image, also calculated and created the texture PDS image, since both calculations involved an inverse 2-D DFT over θ. The width M of both of these square images was 128 so that an FFT form of the DFT could be used. The block diagram in Figure 5.3 demonstrates the combined PDS generation and SAR texture synthesis process.

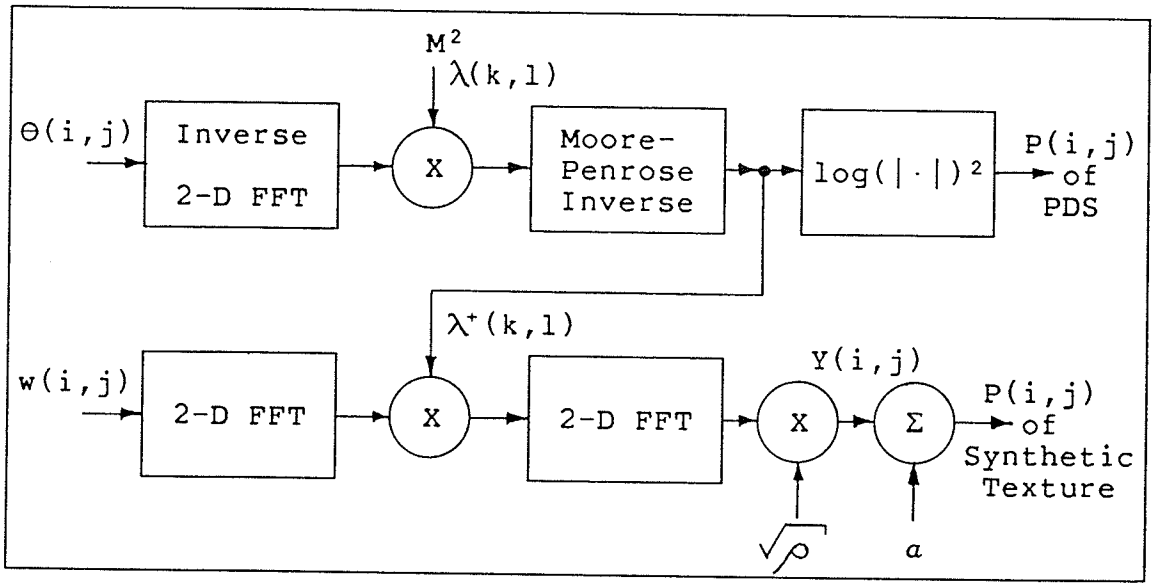


Figure 5.3: PDS Generation and SAR Texture Synthesis

For deriving the least squares AR parameter estimation technique, nothing was assumed about the spatial input noise $\sqrt{\rho}\,w(i,j)$. However, during texture image synthesis, w(i,j) must be uncorrelated if the texture is to have exactly the desired SAR parameters. This is stated by Kashyap. In practice w(i,j) must have a certain amount of correlation

since it is a zero mean real random field. The objective of the experimenter while using the SAR texture synthesis technique then is to maintain the correlation within w(i,j) at a minimum in order that the reproduction of SAR statistics in synthesized textures is optimized.

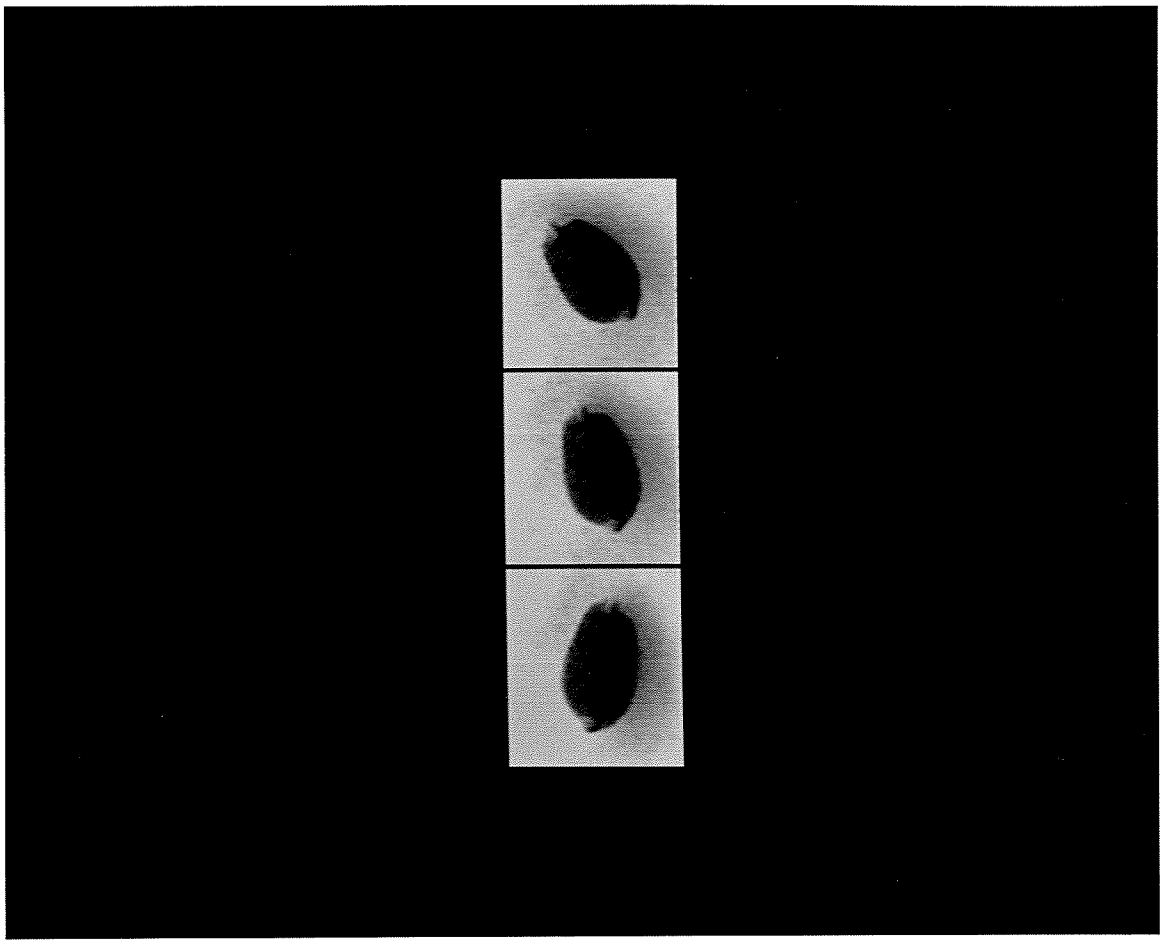
5.6 EXPERIMENTAL RESULTS

5.6.1 Rotation Normalization Test

The first test of the texture analysis technique examined the operation of its rotation normalization facility. The technique was applied to three images of the same wheat kernel. In each succeeding image, the kernel was rotated by a further angle of roughly 30 degrees about an axis parallel to the camera axis. The other imaging conditions for each image were identical. Thus the desired results were that the AR parameters and the PDS of the kernel region in each image would be exactly the same. Figure 5.4 shows these images, called ROTEST1, ROTEST2 and ROTEST3.

The following table lists the AR parameters obtained from

Figure 5.4: ROTEST1, ROTEST2 and ROTEST3



each image with a 3 by 3 influential neighborhood.

Image	Angle	a	AR Coefficients	ρ
ROTEST1	225.289	102.469	* .58853 .13596 .36972 .11603 -.15208 .03370 .02411 -.09673	57.4370
ROTEST2	250.872	105.447	* .57535 .12950 .85755 -.28382 -.10115 -.13275 -.00891 -.02860	32.3359
ROTEST3	281.118	114.391	* .63195 .08628 .96416 -.30657 -.15258 -.18383 -.00547 -.01885	24.0637

*Corresponds to dependent pixel.

For ROTEST2 and ROTEST3 the corresponding AR coefficients, and therefore the AR models, were as expected quite similar. However the AR coefficients for ROTEST1 had many dissimilarities with those of the other two images. In fact some corresponding coefficients had different signs. The most likely cause for this difference in the AR models was that while the elliptical-object detector had accurately specified the kernel contour in ROTEST2 and ROTEST3, it had omitted the small region representing the left half of the brush of the kernel in ROTEST1. Consequently this extremely white patch of the kernel surface had none of the influence on the ROTEST1 AR model which it had had on the AR models of the other two images.

Figure 5.5 shows the PDS and synthetic texture images determined from the 8 by 8 influential neighborhood AR models of ROTEST1, ROTEST2 and ROTEST3. The 8 by 8 influential

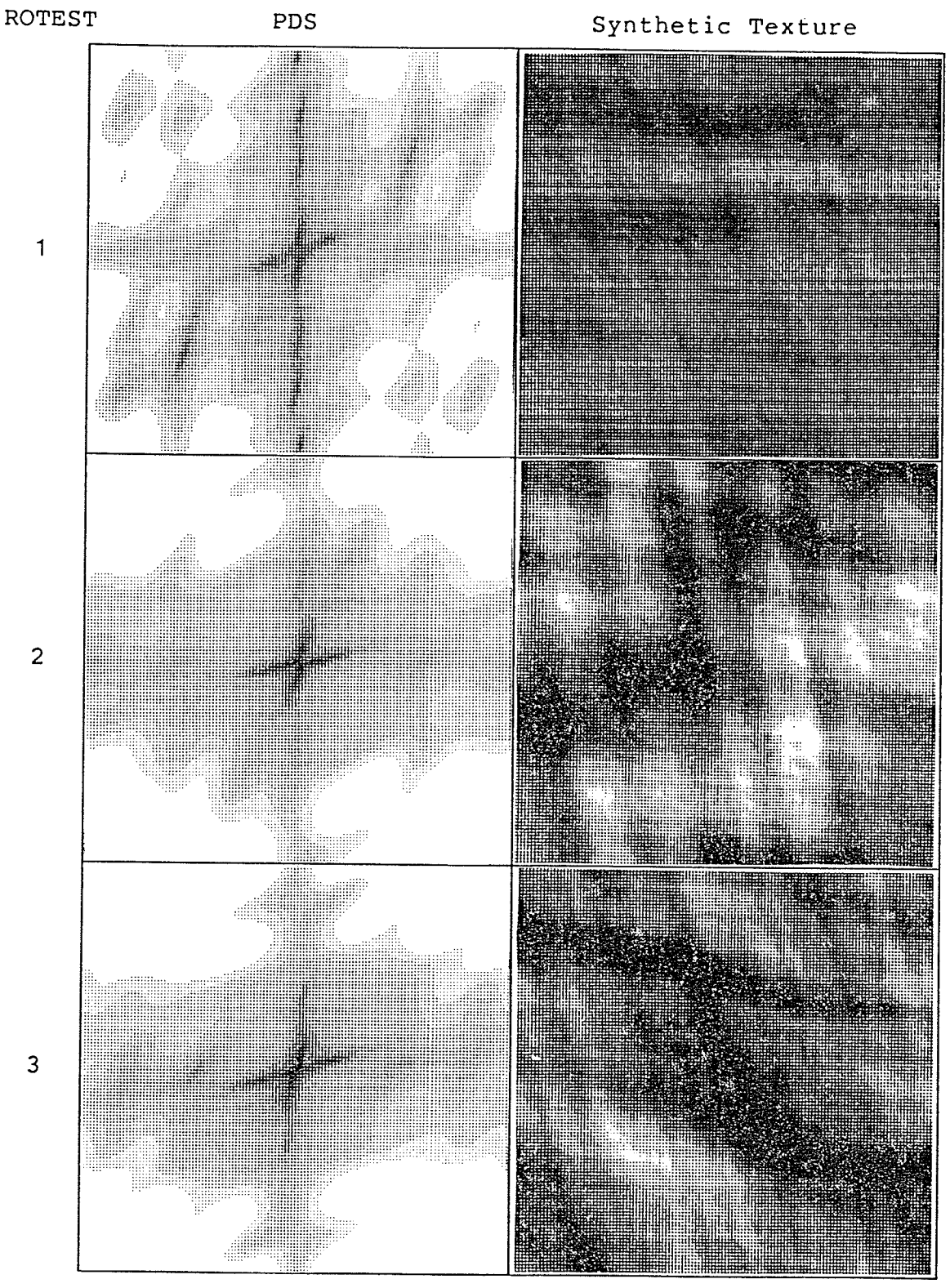


Figure 5.5: PDS and Synthetic Texture Images of ROTEST1, ROTEST2 and ROTEST3

neighborhood size was the arbitrary maximum size allowed by the AR model estimation program. This size provided the most definition in the generation of the PDS. However it required too much computation to be of use in a practical wheat grading system.

As shown in this figure, the PDS and synthetic texture images for ROTEST2 and ROTEST3 had a very similar appearance. The pixels in the synthetic texture images did not have a one-to-one correspondence since each image represented a random process, but it is clear that the texture models which they represented were similar. However the PDS and synthetic texture images of ROTEST1 were not similar to those of ROTEST2 and ROTEST3. Hence the PDS and synthetic texture images upheld the similarities and dissimilarities indicated by the AR coefficients themselves.

The PDS of ROTEST1 displayed a phenomenon typical of images of ventral side up kernels like this one. The step edge formed by the crease was often manifested in the PDS by a ridge that was perpendicular to the crease direction and which intersected the zero frequency point. Since the rotation normalization caused the column axis of the PDS to be aligned with the crease direction, this ridge was always parallel to the row axis of the PDS.

5.6.2 Texture Classification Test

The second test of the texture analysis technique investigated its performance when analyzing representatives of three classes of overall kernel surface quality. These classes were sound, wrinkled and shrivelled. Sound kernels have a smooth surface texture. Wrinkled kernels exhibit a fine surface texture. Shrivelled kernels are smaller and sometimes have a coarse surface texture.

Two calibrated intensity images for each texture class and for each kernel orientation, ventral side up or down, were used. Each image contained five wheat kernels. As a result, ten kernels represented each surface class and kernel orientation combination.

Intensity calibration was necessary because the absolute gray level is an important facet of texture. Intensity calibration ensured the correctness of a gray level and the consistency of the imaging process.

Figure 5.6, Figure 5.7 and Figure 5.8 show the first image of each surface class for the ventral side down (crease not visible) kernel orientation. These images were SOUND1, WRINK1 and SHRIV1 respectively. The surface class represented in each image is obvious from the image's name.

In the first stage of this test, a stepwise linear discriminant analysis was applied to evaluate the merit of the AR model parameters for discriminating surface class. The

Figure 5.6: SOUND1

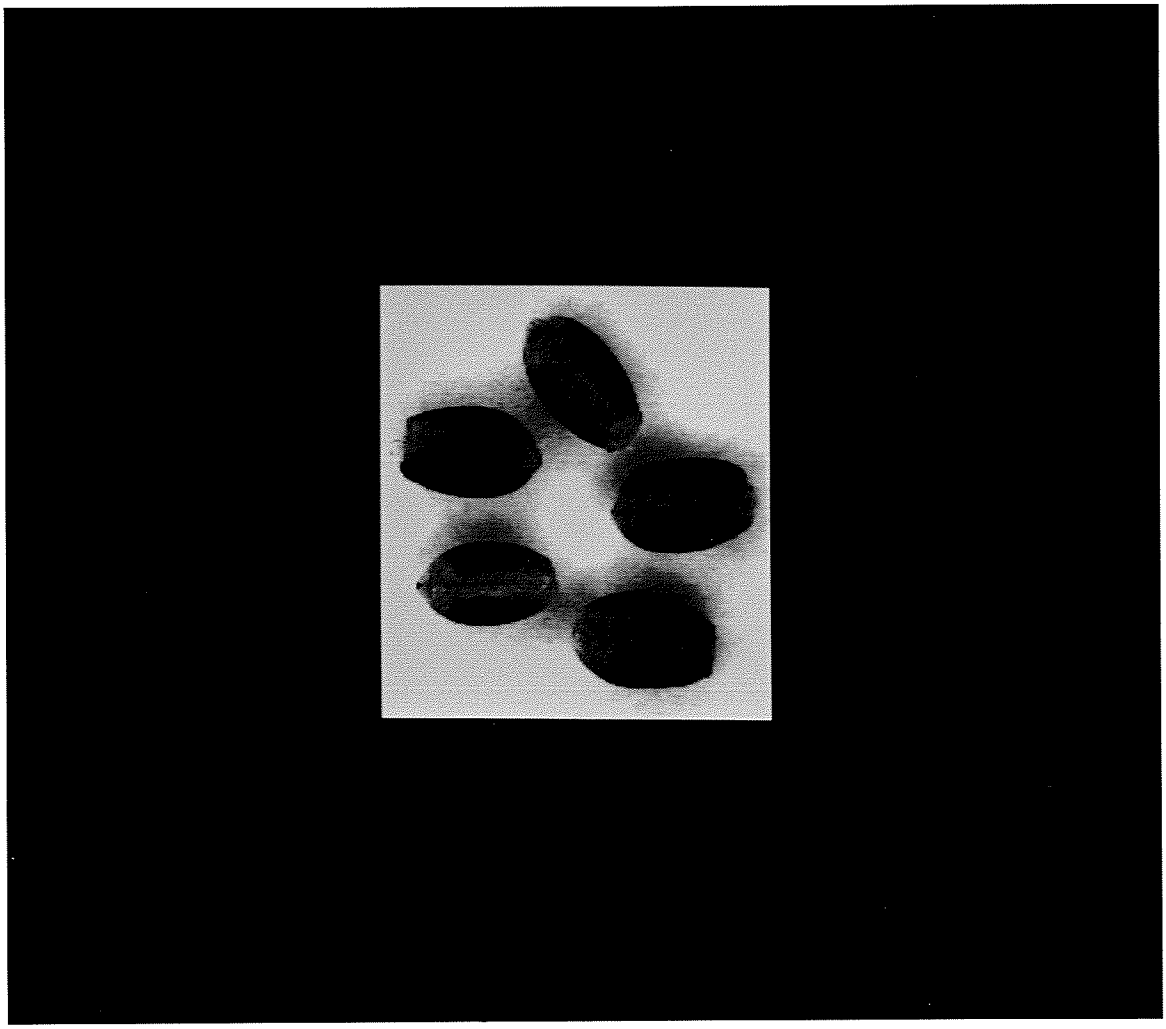


Figure 5.7: WRINK1

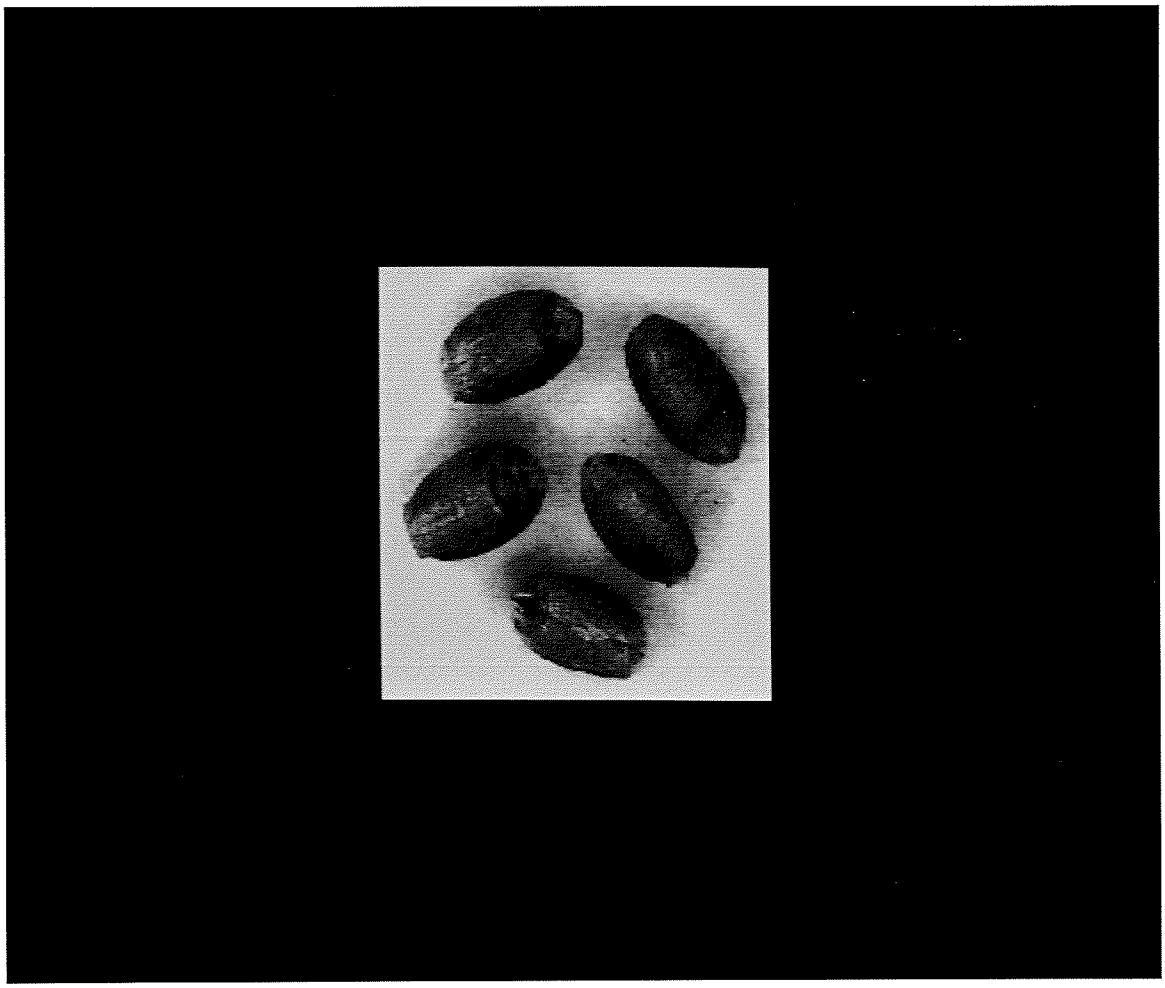


Figure 5.8: SHRIV1

program BMDP7M of BMDP Statistical Software Inc. [University of California, 1981] analyzed the parameters of the 3 by 3 influential neighborhood AR models for the kernel regions of the ventral side down images and then for the ventral side up images. This program found the two most discriminatory linear discriminant functions that utilized only the variables determined to be statistically significant. The first function attempted to classify the surface types for the ventral side down kernels, the second function for the ventral side up kernels.

The results were disappointing. For the ventral side down images, only the average pixel value a and the spatial input noise variance were deemed to be statistically significant. The classification success rate was poor. For the ventral side up images, only the AR coefficient $\theta(2,2)$ was considered to be significant. The classification success rate was even poorer. The following table lists the classification success rates.

Surface Type	Percent Correct	
	Ventral Side Down	Ventral Side Up
Sound	60	70
Wrinkled	70	20
Shrivelled	50	50

These unsatisfactory results indicated that, for a linear classification function at least, the AR parameters of the overall kernel image region were not suitable for discriminating surface type. This conclusion was not entirely unexpected. The surface classes of wheat kernels are given texture descriptions most easily related to energy and spatial frequency content of the surface texture. However the AR coefficients represented the spatial dependence of pixels; only through a Fourier transform would they yield the frequency content in a form such as the PDS.

As a result, a second stage of the test was undertaken. This stage was to provide an intuitive feel for the discriminatory value of the estimated PDS of the texture. PDS images were generated for the first object defined by the elliptical-object detector in each of the twelve test images. For each of these objects, a PDS image was generated corresponding to the 2 by 2, 3 by 3, 5 by 5 and 8 by 8 influential neighborhood AR models. Thus a wide range of definition of the PDS of each evaluated object was available.

Figure 5.9, Figure 5.10 and Figure 5.11 show the PDS images for the 2 by 2, 3 by 3 and 5 by 5 influential neighborhood AR models for the first object in SOUND1, WRINK1 and SHRIV1 respectively. In SOUND1 the first object was the top-left kernel. In WRINK1 the first object was the top kernel. In SHRIV1 the first object was the bottom-left kernel. For interest sake these figures also show a synthetic texture image for each AR model.

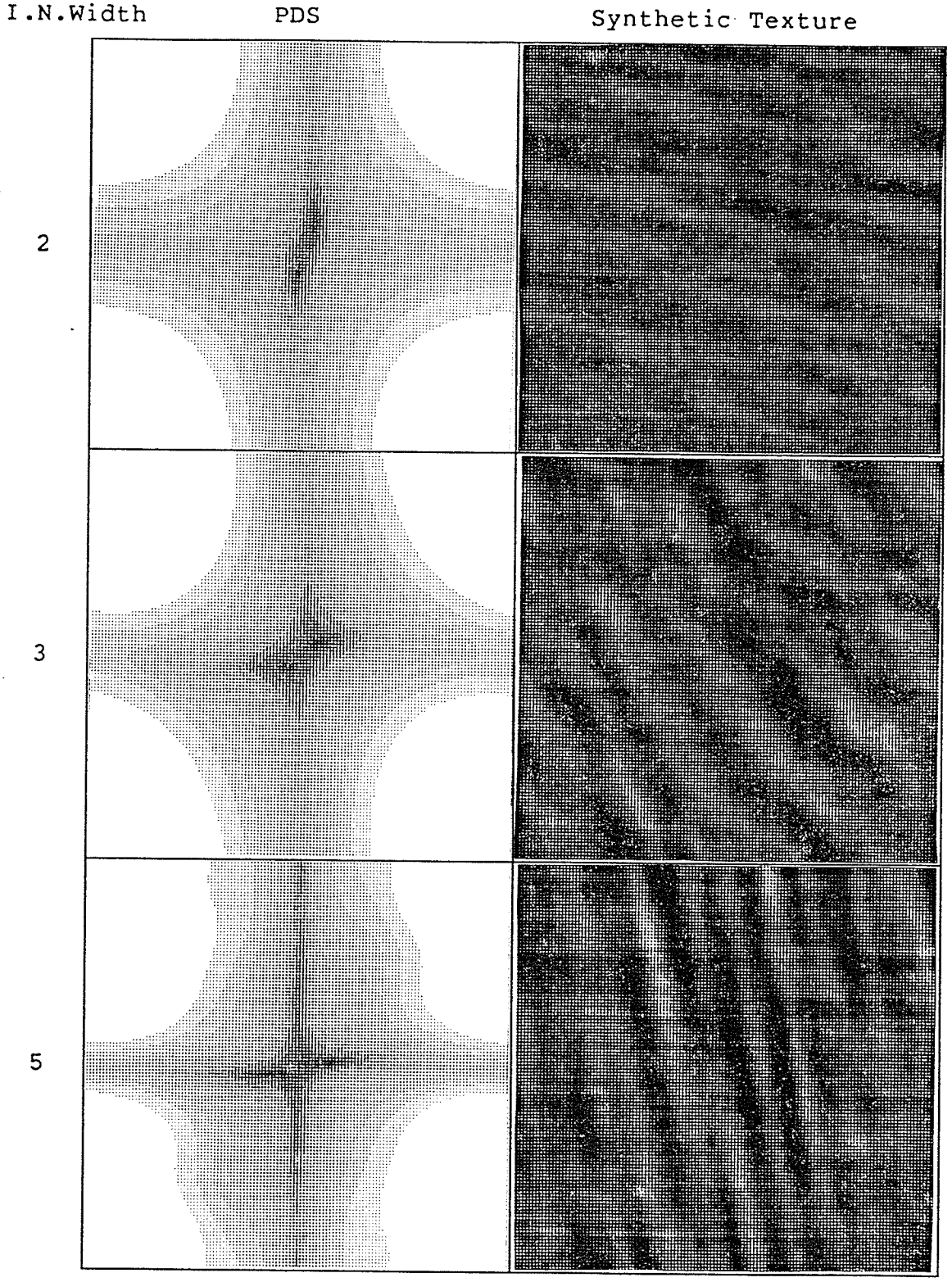


Figure 5.9: PDS and Synthetic Texture for Top-Left Object in SOUND1

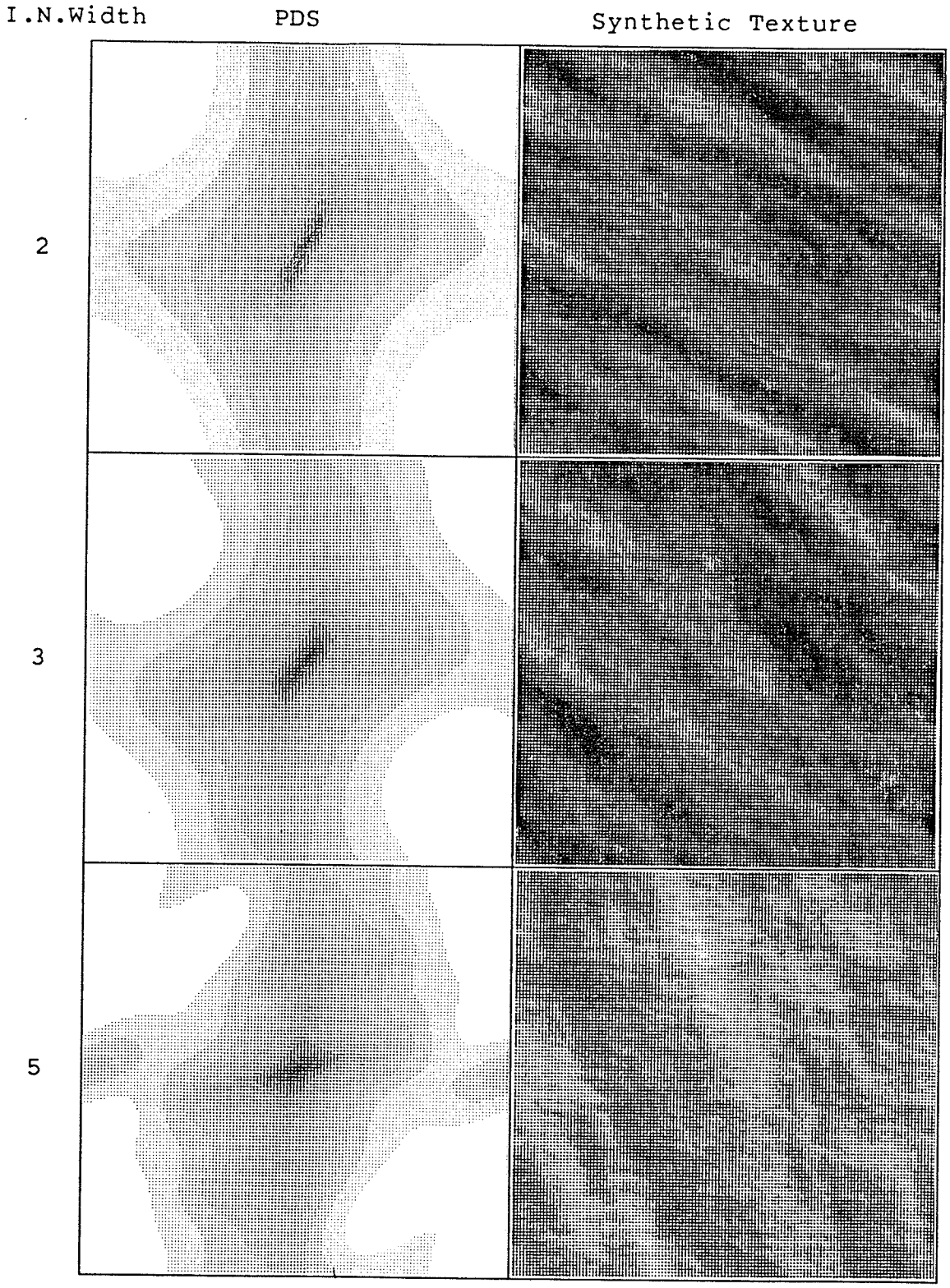


Figure 5.10: PDS and Synthetic Texture for Top Object in WRINK1

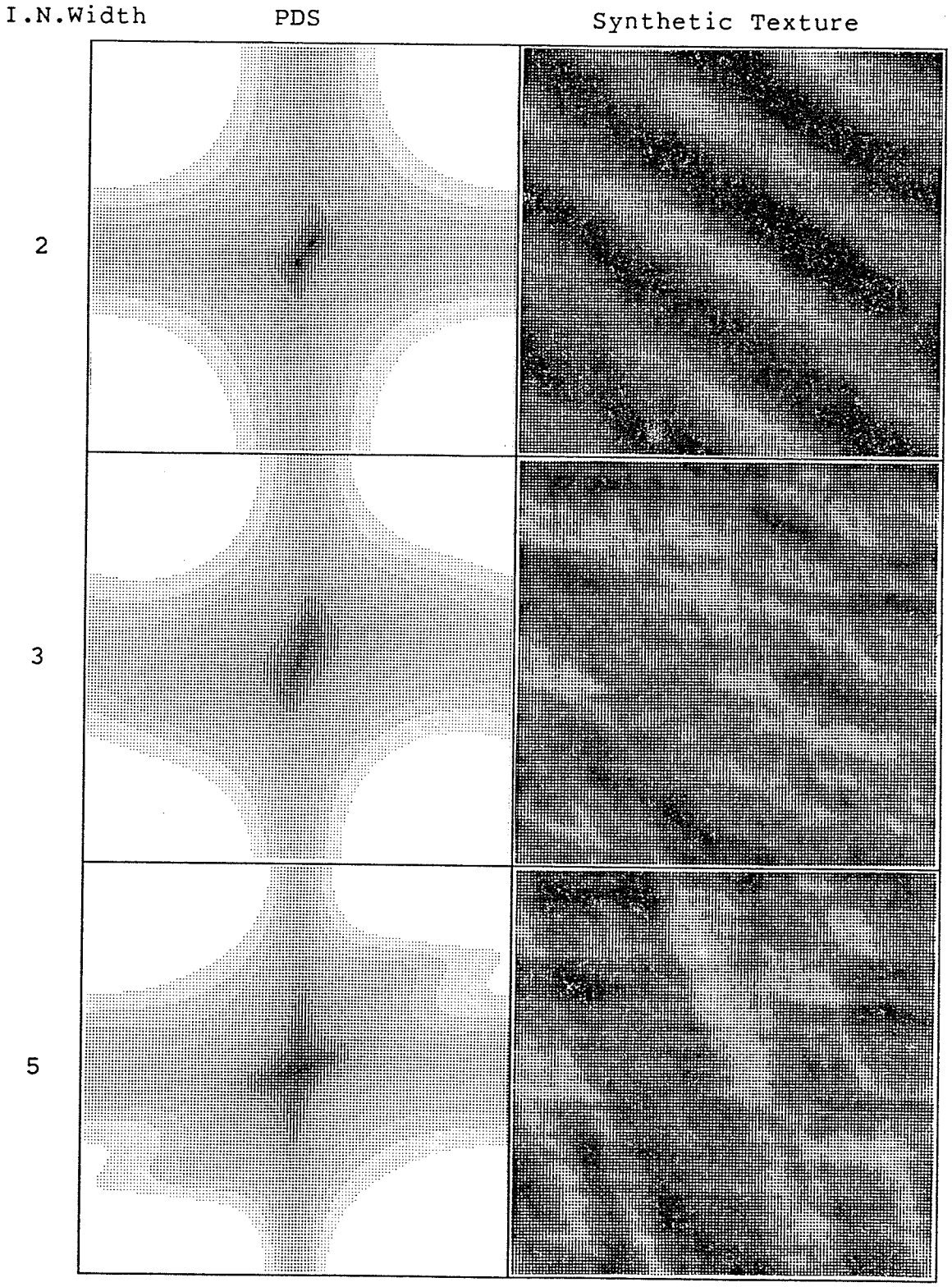


Figure 5.11: PDS and Synthetic Texture for Bottom-Left Object in SHRIV1

The conclusion reached from inspection of all the PDS images was that the PDS of the texture model of the entire kernel image region did not have any obvious features that indicated the surface class. The anatomical features, especially the crease, germ and brush, caused bands or ridges in the PDS. These effects tended to drown out features which originated from the actual surface texture. Therefore, it was decided that for the texture analysis technique to be useful at distinguishing the surface texture class of the overall kernel surface, it must be modified in future research in such a way that it can mask out the effects of the anatomical features on the visible kernel surface.

5.7 RECOMMENDATIONS

Texture analysis and its application to surface classification is by far the portion of this research most in need of future development and refinement. The texture model estimation, PDS estimation and texture synthesis techniques themselves performed well the tasks for which they were designed. However the model estimation program, as a source of texture features, had two general deficiencies which disallowed satisfactory surface classification. First, the AR model parameters described the spatial dependence of pixels, instead of the composition of the texture in the frequency domain. Second, the model estimation procedure was influenced by the gross anatomical features of the kernel to the extent of overpowering the influence of the actual surface

texture. This section presents suggestions for the solution of these two problems.

PDS features based on the AR model can be calculated in the same way that the PDS image was generated. However, the PDS image was actually a needless overspecification of the estimated PDS of the texture since it supplied $M^2 = 16384$ pixel values or parameters when the model had at most only 63 coefficients. An expression which completely specifies the PDS without redundancy is:

$$\hat{P}(k,l) = \frac{\rho}{\left| \sum_{p=0}^{N_1-1} \sum_{q=0}^{N_2-1} a(p,q)\, e^{-j\frac{2\pi pk}{N_1}}\, e^{-j\frac{2\pi ql}{N_2}} \right|^2} ,$$

$$\text{for} \quad 0 \le k \le N_1-1 \quad \text{and} \quad 0 \le l \le (N_2-1)/2$$

where $$a(p,q) = \begin{cases} 1, & p=0 \text{ and } q=0 \\ -\theta(p,q), & \text{otherwise.} \end{cases}$$

The variable l is employed over only half of its valid range since:

$$\hat{P}(k,l) = \hat{P}(N_1-k, N_2-l) .$$

The $N_1 \cdot (N_2+1)/2$ values of $\hat{P}(k,l)$ would be used, instead of the $N_1 \cdot N_2 - 1$ AR coefficients themselves, as features for a statistical analysis like BMDP7M.

An attempt could be made to minimize the effects of anatomical features on the AR model by limiting the AR estimation region to image areas where both the presence of these

features is unlikely and the surface texture should find the most expression. Figure 5.12 shows a pair of such areas, each of which follows a lateral edge or limb of the visible

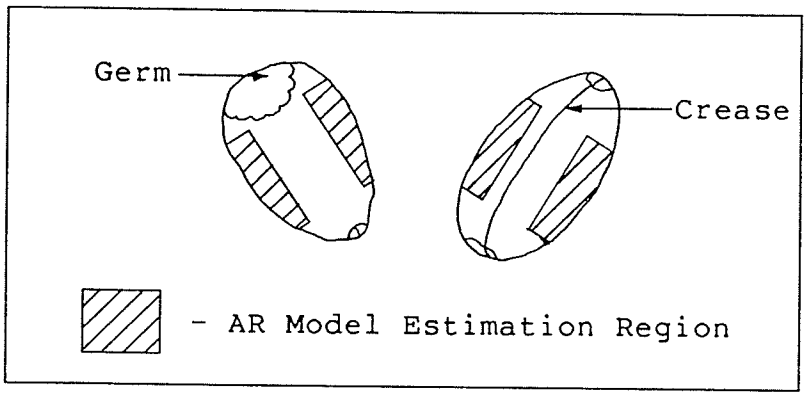


Figure 5.12: Suggested AR Model Estimation Regions

kernel surface. In both of these areas the surface texture will likely be most evident since the light of illumination only grazes the kernel surface near the kernel limb. In addition these areas are not where the crease, germ or brush are typically found for either a ventral side up or down kernel.

Chapter VI

CONCLUSIONS AND RECOMMENDATIONS

6.1 CONCLUSIONS

The goal of this research was to develop the computer software that will be initially required during the development of a successful, machine vision based, automated wheat grading system. This software fell into three broad categories: digital image formation, primitive image understanding and the extraction of shape and texture features directed by this image understanding.

The research proceeded in four general phases of software development. The first phase created assembly language programming for a microcomputer system. The remaining phases produced programs which were written in FORTRAN77 and executed on a mainframe computer. Each phase had a different degree of success.

In the first phase, the Image Manipulation Package (IMP), a collection of 8086 assembler routines, was developed. IMP provided facilities on the custom built black and white digital image acquisition system which are found on standard commercial imaging systems. IMP satisfied the imaging needs of this research.

In the second phase, four different approaches were developed for computer perception of the objects contained in a typical wheat grading image. Each of these approaches was able to perform the object perception function. However, the last approach, the elliptical-object detector, was the superior technique and it represented the culmination of this phase. The use of image context and a priori knowledge were maximized in this detector to produce a dynamic, reliable and efficient technique. It achieved good results even with scenes which suffered from shadow because of close and sometimes touching objects. The reliability of this detector made it suitable for use in an automated system.

During this phase a technique was also implemented to describe the shape of the image region of each perceived object. This technique was based on moments. It efficiently provided shape features that were normalized with respect to translation, rotation and size. It also provided characteristics, such as length, width and the principal major axis location, which guided succeeding analysis.

In the third phase, two techniques were developed to detect and specify two important anatomical parts of the wheat kernel: the crease and the germ. Knowledge of the presence and whereabouts of these parts implied the orientation, either ventral side up or down, of the kernel and the location of surface features relevant to wheat grading.

The performance of these two techniques was mediocre. Each performed satisfactorily if the image definition of the part was at least fair. However if the part was poorly defined, the technique would often miss the part and sometimes have difficulty determining that it had failed. By comparison, the human vision system could more easily detect the poorly defined parts in the same digital images. The essential deficiency of these techniques was their two-dimensional approach to a three-dimensional problem; they did not employ the knowledge that the images were representations of three-dimensional solids and that the crease and germ are manifestations of the kernel's surface topography.

In the fourth and final phase, a signal processing technique, two-dimensional autoregression modelling (2-D AR), was implemented to analyze the kernel surface texture. Four facilities of the texture analysis were created: rotation normalization of the analysis, estimation of the 2-D AR texture model, generation of the image of the estimated power density spectrum (PDS) of the texture and generation of a synthetic texture image which obeyed the estimated 2-D AR texture model. These facilities worked well to the extent of their design. However, this phase was by far that most in need of future development and refinement. This need was caused by the shortcomings of the texture model estimation facility.

The texture model estimation facility was the most important of the four facilities since it was the source of texture features. Yet it had two deficiencies which prevented successful surface classification based on these features. First, these texture features, which were the 2-D AR model parameters, described the spatial dependence of pixels. However, it would have been more desirable to employ features which succinctly described the estimated PDS of the texture. Second, the model estimation procedure allowed the gross anatomical features of the kernel to have more influence on the AR texture model than that of the actual surface texture. Evidently the texture model estimation procedure must be improved if it is to serve successfully in the wheat grading system.

6.2 RECOMMENDATIONS

This research was the preliminary part of a larger research project which has definite goals. Therefore several rather obvious ideas can be stated which are as much a commentary on the likely future pursuits of the research team as they are actual recommendations.

Evidently classification functions must be determined which will utilize the shape and texture features made available in this research to obtain degrading factors. Perhaps other features not produced during this research may also be required. In any event a vast amount of statistical

analysis of image data remains before satisfactory classification functions can be determined.

A definite plan for the overall wheat grading system's decision structure will have to be chosen. This decision structure will direct the analysis of each perceived object, from initial perception to identification and, if necessary, to feature extraction and grade determination. Knowledge of this structure is important to members of the research team since the information/processing structures which they will design must model it. Ballard and Sklansky [1976] present a classic example of the value of a well designed decision structure in an intelligent image analysis system. In this research a simple information/processing tree was utilized. Figure 6.1 shows this structure. The use of an expanded version of this structure would be advantageous in future work by the research team.

The research team may find it necessary to employ colour as a feature for detecting grass green kernels or wheats of other classes such as soft white spring and amber durum. The addition of colour could be inexpensively accomplished by attaching colour filters to the existing black and white imaging system camera.

Several recommendations can be made which pertain more directly to the results of this research. These are discussed in the following two subsections.

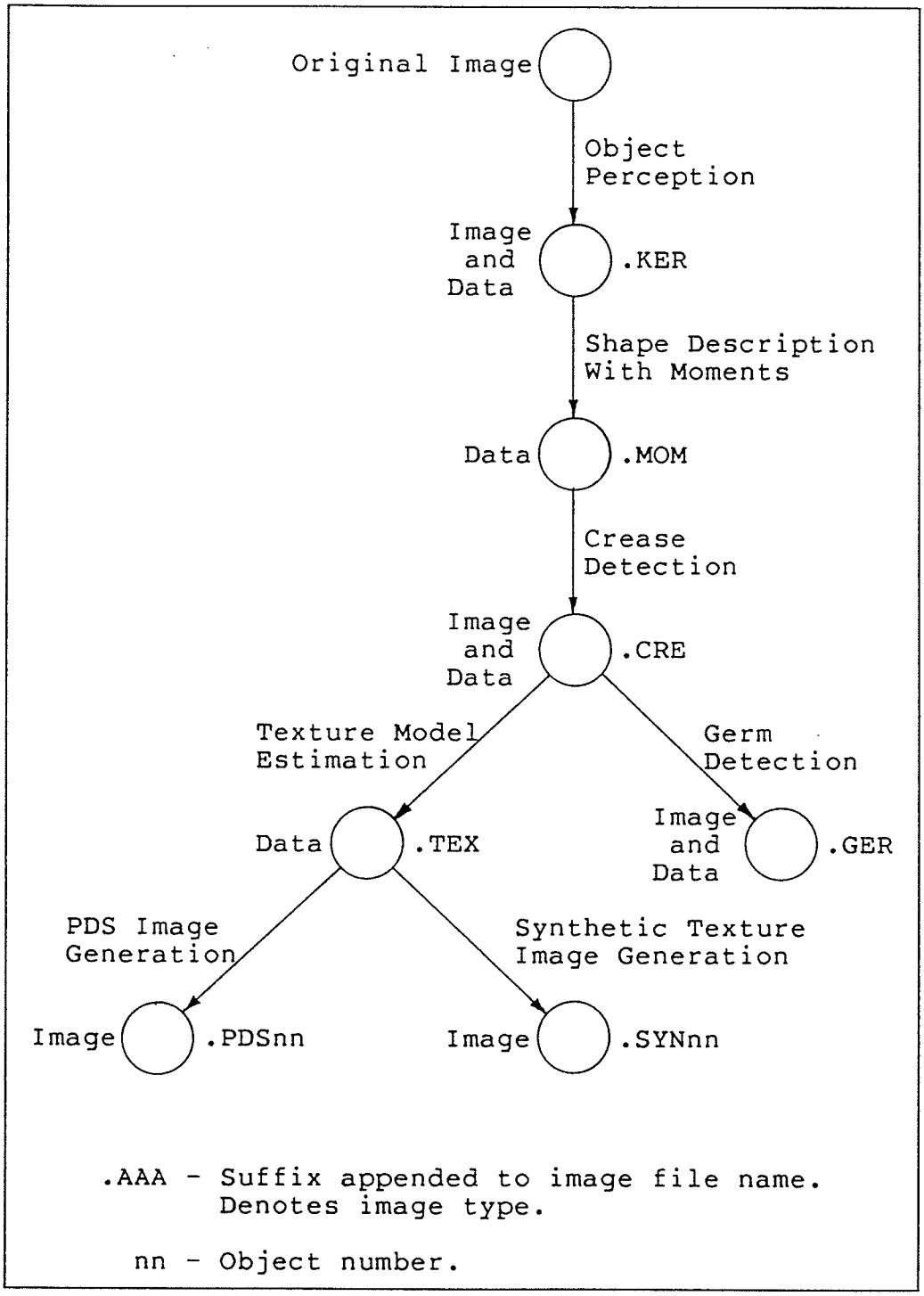


Figure 6.1: Information/Processing Tree Used in this Research

6.2.1 Short Term

Only one of the routines developed in this research, the AR texture model estimation procedure, is drastically in need of immediate improvement.

Section 5.7 suggested two modifications to this routine. The first was to utilize the nonredundant PDS values as texture features instead of the AR model parameters themselves. The second was to limit the AR model estimation region to image areas where the presence of interfering anatomical features is unlikely and the actual surface texture finds most expression.

It is recommended that this routine should be revised as suggested in the near future.

6.2.2 Long Term

Any of the following related problems may appear in work undertaken by the research team in the more distant future. It may become necessary to improve on the mediocre performance of the kernel anatomy detection software; the texture analysis methodology may have to be upgraded by modelling kernel surface texture with a three-dimensional (3-D) deterministic surface rather than the present 2-D stochastic process; the ability to perceive partially occluded objects may become necessary. These problems are similar in that the solution of each of them requires a 3-D approach to image understanding.

In Subsection 4.2.2 it was suggested that one method for implementing 3-D image understanding would be the following: estimate local surface orientation by using local surface gray level, generate a 3-D model of each object using this estimated surface orientation and, finally, direct subsequent analysis on this model, not on the image itself. Horn and Ikeuchi [1984] have implemented such a method.

It is recommended that the research team be aware of this radically more advanced approach to image understanding as a means of dealing with any of the previously mentioned problems.

REFERENCES

Ballard, Dana H. 1982. Computer Vision. Englewood Cliffs, New Jersey: Prentice Hall, Inc.

Ballard, Dana H. 1981. "Generalizing the Hough Transform to Detect Arbitrary Shapes." Pattern Recognition, Volume 13, Number 2, 1981: 111-122.

Ballard, Dana H.; and Sklansky, Jack. 1976. "A Ladder-Structured Decision Tree for Recognizing Tumors in Chest Radiographs." IEEE Transactions on Computers, Volume C-25, Number 5, May 1976: 503-513.

Brogan, William L.; and Edison, Allen R. 1974. "Automatic Classification of Grains Via Pattern Recognition Techniques." Pattern Recognition, Volume 6: 97-103.

Brogan, William L.; and Inguanzo, Jose M. 1978. "Application of a Covariance Matching Adaptive Filtering Technique to Pattern Classification." North Hollywood, CA, USA: Presented at the 21st Midwest Symposium on Circuits and Systems (IEEE), August 14-15, 1978.

Canada Grains Council. 1982. Grain Grading for Efficiency and Profit. Winnipeg, Manitoba: A report submitted by The Grain Grading Committee, Canada Grains Council, September 1982.

Canadian Grain Commission. 1981. Official Canadian Grain Grading Guide. 1981 Edition.

Candlish, V.E. 1984. "Computers in Elevators." Edmonton, Alberta: Presentation to the Canadian Grain Council, November 14, 1984.

Chellappa, R.; and Kashyap, R.L. 1981. "Synthetic Generation and Estimation in Random Field Models of Images." Proceedings of the IEEE Computer Society Conference on Pattern Recognition and Image Processing, Dallas, Texas, August 3-5, 1981: 577-582.

Data General Corp. 1983. FORTRAN77 Reference Manual. Revision October 2, 1983.

Davis, Philip J. 1978. Circulant Matrices. John Wiley & Sons.

Deguchi, Koichiro; and Morishita, Iwao. 1978. "Texture Characterization and Texture-Based Image Partitioning Using Two-Dimensional Linear Estimation Techniques." IEEE Transactions on Computers, Volume C-27, Number 8, August 1978: 739-745.

Digital Research. 1982. CP/M-86 Operating System User's Guide. Second Printing July 1982.

Draper, S.R.; and Travis, A.J. 1984. "Preliminary Observations With a Computer Based System for Analysis of the Shape of Seeds and Vegetative Structures." Journal of the National Institute of Agricultural Botany, 16: 387-395.

Duke, V. 1982. "Grading of Major Cereal Grains in Canada." Winnipeg, Manitoba: Presented to the Seventeenth International Grain Industry Course, September 10, 1982.

Gambotto, J.P. 1980. "Two Dimensional Time Series for Textures." Digital Image Processing: Proceedings of the NATO Advanced Study Institute held at Bonas, France, June 23-July 4, 1980.

Goodman, D.E.; and Rao, R.M. 1984. "A New, Rapid, Interactive Image Analysis Method for Determining Physical Dimensions of Milled Rice Kernels." Journal of Food Science. Volume 49: 648-649.

Haralick, Robert M. 1979. "Statistical and Structural Approaches to Texture." Proceedings of the IEEE, Volume 67, Number 5, May 1979: 786-804.

Haralick, Robert M. 1982. "Zero Crossing of Second Directional Derivative Edge Operator." Robot Vision, SPIE, Volume 336, 1982.

Hildreth, Ellen C. 1982. "Edge Detection for Computer Vision System." Mechanical Engineering, August 1982: 48-53.

Horn, Berthold K.P.; and Ikeuchi, Katsushi. 1984. "The Mechanical Manipulation of Randomly Oriented Parts." Scientific American, Volume 251, Number 2, August 1984: 100-111.

Intel Corp. 1983. iAPX 86, 88, 186 and 188 User's Manual Programmer's Reference.

Kashyap, R.L. 1980. "Random Field Models on Torus Lattices for Finite Images." Proceedings of the 5th International Conference on Pattern Recognition (IEEE), Miami, Florida, December 1980: 1103-1105.

Lester, J.M.; et al. 1978. "Two Graph Searching Techniques for Boundary Finding in White Blood Cell Images." Computers in Biology and Medicine, Volume 8: 293-308.

Makhoul, John. 1975. "Linear Prediction: A Tutorial Review." Proceedings of the IEEE, Volume 63, April 1975: 99-118.

Martelli, Alberto. 1972. "Edge Detection Using Heuristic Search Methods." Computer Graphics and Image Processing, 1972, Volume 1: 169-182.

Nilsson, N.J. 1971. "Problem Solving Methods in Artificial Intelligence." New York: McGraw-Hill.

Owen, C.H.; and Ainslie, M.M. 1971. Varietal Identification of Barley, Wheat and Small Oilseeds by Kernel Characters. Ottawa, Canada: Queen's Printer for Canada.

Pavlidis, Theo. 1981. "A Flexible Parallel Thinning Algorithm." Proceedings of the IEEE Computer Society Conference on Pattern Recognition and Image Processing, Dallas, Texas, August 3-5, 1981: 162-167.

Reeves, Anthony P.; and Rostampour, Abdolrahim. 1981. "Shape Analysis of Segmented Objects Using Moments." Proceedings of the IEEE Computer Society Conference on Pattern Recognition and Image Processing, Dallas, Texas, August 3-5, 1981: 171-174.

Slicer Computers Inc. 1983. Slicer Documentation, June 1983.

Symbolics, Inc. 1982. LGP-1 Technical Manual, Preliminary Edition, November 1982.

Tang, Gregory Y. 1981. "A Discrete Version of Green's Theorem." Proceedings of the IEEE Computer Society Conference on Pattern Recognition and Image Processing, Dallas, Texas, August 3-5, 1981: 144-149.

Tipples, Keith. 1982. "Effects of Grading Factors Upon Utilization Quality." Winnipeg, Manitoba: Presented to CIGI Seventeenth International Grain Industry Course, September 10, 1982.

University of California, Department of Biomathematics. 1981. BMDP Statistical Software, 1981 Edition. University of California Press.