2003

Unsupervised skin lesion classification and matching

Paula Yandow-Reilly

Rochester Institute of Technology
Computer Science Department

Unsupervised Skin Lesion Classification and Matching

By

Paula Yandow-Reilly

Thesis submitted in to the Faculty of the of the Computer Science Department, in partial fulfillment of the requirements for the Degree of Master of Computer Science
In the Golisano College of Computing and Information Sciences

Approved by:

_____
Professor Roger Gaborski

_____
Professor Hans-Peter Bischof

_____
Professor James Heliotis

February 2003

Thesis Reproduction Permission Statement

Permission granted

Title of thesis : ***Unsupervised Skin Lesion Classification and Matching***

I, ***Paula Yandow-Reilly***, hereby **grant permission** to the RIT Library of the Rochester Institute of Technology to reproduce my thesis in whole or in part. Any reproduction will not be for commercial use or profit.

Date: _February 25, 2003

Signature of Author: _____

Skin Lesion Analysis via Image Processing and Neural Net Matching

Paula Yandow-Reilly
February 2003

Graduate Computer Science Department
Rochester Institute of Technology
Rochester, New York 14616

Abstract

According to the American Cancer Society (ACS), since 1973, the mortality rate for melanoma has increased by 44%. The number of serious skin cancers diagnosed has also more than doubled in that same period. Even though serious skin cancers (melanoma) account for only 4% of skin cancer diagnoses (and skin cancer is the most common cancer) it is responsible for almost all (79%) cancer deaths. The ACS reports about 7,300 people in the United States are expected to die of melanomas in 2002, other sources put the number as high as 7,800. There are about 130,000 cases of melanoma worldwide, and about 37,000 related deaths. Many physicians think the increase in melanoma diagnoses represents an epidemic. Currently, there is work to improve diagnostics once a lesion comes under suspicion, and there are also systems to do whole body images of skin lesions. Where there seems to be a gap is in tracking and classifying the lesions in image histories. The critical problem is not so much how to treat the lesion once its discovered, but to detect it in the first place. In addition, in the classification systems encountered, there didn't seem to be any using all combinations of color, texture, and shape, any or all of which can help detect a malignant growth. Since almost all lesions are slow-growing, and very often on the back, it can be difficult for both patient and doctor to detect when a lesion has begun to change, which is one of the first warning signs of skin cancer.

This work is comprised of an analysis system written in Matlab, which pre-processes the image, removing background artifacts via morphological operations to segment the lesion. The lesion is then processed for shape, color content, and texture. This occurred for a small database of images comprising melanomas, dysplastic nevi, and moles, and 10 feature vectors were captured for each image along with the filename and matching diagnosis. Additional images were procured from the web, and also from photographs of individuals using a Cannon EOS Rebel G, which were scanned in using an Acer ScanPrisa 640U. These images were then processed with the same software used for the database images. The results were classified based on these feature vectors and assigned a FWL (Feature Warning Level). Lastly, the input results were compared to the database for matches within a range for similarity. The closest match (if within a reasonable range) is reported.

This system could be attached to existing tracking systems (like MoleMap) or used as a stand alone tracking tool for dermatologists. Any change in one of the feature vectors, or in a group of features could trigger a closer look by the physician. According to literature, and a dialog with a dermatologist, history is the one of the most critical factors in early detection, when the cancer can be completely cured.

Unsupervised Skin Lesion Classification and Matching

By

Paula Yandow-Reilly

Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Computer Science

In the Golisano College of Computing and Information
Sciences, Department of Computer Science

February 2003

DEDICATION

To my loving and long-suffering family, my deepest gratitude for their support and fortitude while they persevered without me for so many hours. To my immediate managers and my staff, who, supported, encouraged, and tolerated my preoccupation with this research effort my sincere thanks. I also want to thank Dr. Bischof and Dr. Carithers for supporting me in my hour of need. And finally, and especially, to Dr. Gaborski, for not only supporting and encouraging me, but for re-firing my interest and enthusiasm for science.

Table of Contents

Section I : Introduction

1.1 Research Motivation

According to the American Cancer Society (ACS), since 1973, the mortality rate for melanoma has increased by 44%. The number of serious skin cancers diagnosed has also more than doubled in that same period. Even though serious skin cancers (melanoma) account for only 4% of skin cancer diagnoses (and skin cancer is the most common cancer) it is responsible for almost all (79%) cancer deaths. The ACS reports about 7,300 people in the United States are expected to die of melanomas in 2002. [1] on their site, other sources put the number as high as 7,800.[2] There are about 130,000 cases of melanoma worldwide, and about 37,000 related deaths.[3] In addition, while fair skinned individuals living in sunny climates are most at risk, it is a myth that they are they only vulnerable demographics. Individuals with darker skin are also at risk, especially on the soles of the feet, palms of the hands, under nails, or in the mouth. Many physicians think the increase in melanoma diagnoses represents an epidemic. Here is an excerpt from a white paper on the American Cancer Society web site:

> " Malignant melanoma will be diagnosed in over 40,000 people this year; about 7,300 will die from it. Deaths from melanoma have climbed steeply, an increase many doctors attribute to more recreational sun exposure and, possibly, to the thinning of the ozone layer which acts as a buffer between the earth and the sun's rays. " [4]

As with any cancer, early detection is key, and this could be aided by an easy to use diagnostic system, which would allow the user to enter an image for classification. Since the importance of early detection cannot be over-emphasized, even if the lesion is mis-classified, this is better than not catching a lesion before it's metastasized.

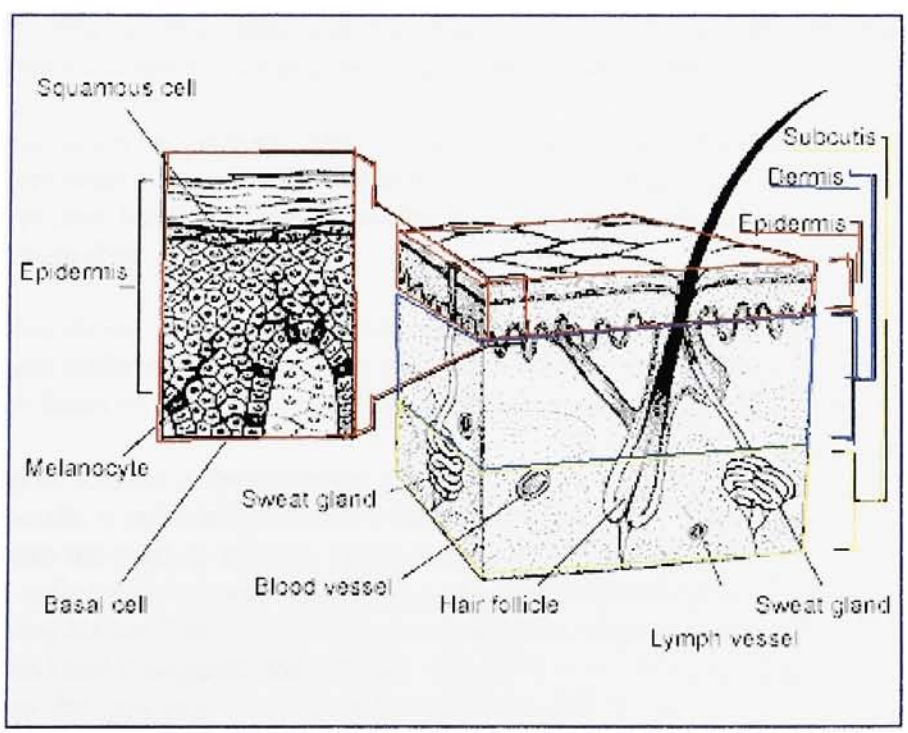


Figure 1.1 Anatomy of normal skin from the American Cancer Society Website *

---

* http://www.cancer.org/docroot/CRI/content/CRI_2_2_1X_What_is_melanoma_skin_cancer_50.asp?siteareа=CRI

1.2 Skin Anatomy

Weighing on average, about six pounds, skin is the largest organ for humans. Skin plays a vital role in protecting internal organs, and also provides a critical barrier for both keeping things out (like microbes and other pollutants) and keeping things in (like water and other fluids). It also helps us maintain a steady temperature, and helps the body excrete excess water and salts. Lastly, specialized cells in the skin allow sensations of temperature, touch, and pain.

As indicated in figure 1.1, the skin has three major layers: epidermis, dermis, and the subcutis. The epidermis is the outer layer, and is thin, (around 0.2 mm) and protects the deeper layers of skin. The stratum corneum is the outer horny layer composed of dead keratinocytes which are continuously shed. These cells are pushed up from the lowest level in the epidermis: the basal cells. The round shaped basal cells produce the flat, scale-like living squamous or keratinocyte cells that make up the inner layer of the epidermis. As these cells age and die off, they move to the surface to be shed. This process takes around 30 days. Interspersed in the lower part of the epidermis layer are also the melanocytes that produce the protective pigment, melanin, that helps shield the deeper layers of the skin from harmful solar radiation. This layer ends with something called the basement membrane, which separates the epidermis from the dermis.

The dermis layer is thicker and contains hair follicles, blood vessels, sebum and sweat glands, and nerves. The sweat and sebum reach the skin's surface via tiny pores. These important structures are held in place by a substance called collagen. Fibroblasts make the collagen, which gives skin strength and flexibility. The papillary dermis contains blood vessels that provide both the epidermis and the non-vascular epidermis with vital nutrients. Further, the vasculature is organized so that by increasing or decreasing blood flow, heat can either be conserved or dissipated. [3] The papillary dermis also contains the free sensory nerve endings and structures called Meissner's corpuscles that provide the sensory data to the brain for touch, pain, and temperature.

The demarkation between the dermis and the deepest layer, the subcutis, is a bit blurred. Both of these layers contributing to a buffer layer of fat and collagen that provides a kind of shock absorber to protect internal organs, and also help conserve body heat.

1.2 Skin Anomolies

As detailed above, skin is actually a heterogenous organ, made up of a diverse group of tissues. These different kind of tissues develop a wide variety of benign as well as malignant tumors. The focus of this work is on melanocytic lesions, as opposed to other growths and rashes.

Melanocytic lesions is the technical term for what is often called a "beauty mark" or mole. These are usually a uniform brown in color and uniform in texture, with a round or oval shape, and sometimes the mark is raised. These lesions are formed by groupings or "nests" of melanocyte cells (which produce the brown pigment, melanin,hence the color of the moles.) Basically these lesions fall into three loose categories : normal or benign, dysplastic (often called pre-cancerous) and malignant melanoma. The trick is to correctly determine which is which. The image on the next page is of a normal mole: its uniform in color, shape and texture.

figure 1.2 Benign Nevus *

The two images below are the one's of interest to the clinician: the first image is usually indicating a warning sign, and the second requires surgical removal.

figure 1.3 Dysplastic Nevus *

figure 1.4 Malignant Melanoma *

The table below details some of the more common skin anomalies.

type of anomaly	description	benign	malignant
moles	tumors that develop from melanocytes	X	
Seborrheic keratoses	tan, brown, or black raised spots with a "waxy" texture or rough surface	X	
hemangiomas	blood vessel growths (aka "Strawberry spots or port wine stains") Hemangiomas are a common vascular birthmark. They are usually painless and benign. The cause of hemangiomas development is unknown. The color results from a proliferation of blood vessels at the sight.	X	
lipomas	soft growths of fat cells	X	
warts	rough surfaced growths caused by a virus	X	
pigmented birthmark	A congenital pigmented skin marking that ranges in color from brown or black to bluish or blue-gray	X	
freckle	small area of pigment, often found in light skinned individuals	X	

*

http://www.cancer.org/docroot/PED/content/ped_7_1_Skin_Cancer_Detection_What_You_Can_Do.asp?sitearea=PED

1.3 Melanoma Detection

One of the most challenging tasks for a medical care provider is accurate diagnosis and treatment of skin lesions. In some cases, its only possible to correctly diagnose a lesion after excisement and microscopic analysis. Its possible for a early-stage melanoma (in-situ melanoma) to appear as a normal non-cancerous mole (benign nevus) and much more common, for a normal mole to appear cancerous.

In an interesting lecture at Yale University, Dr. James Grichnik reported when a physician encountered a skin lesion, the odds are 200,000 to 1 that it's early stage melanoma.[5] Therefore, it appeared to him to be both impractical and over-kill to remove all moles for every patient with a concern. He further indicated that this that he had four steps to try to tease out the correct diagnosis:

1. listen to the patient (for reasons unclear to the clinician, the patient correctly worries about the one mole of many that is cancerous.)
2. total body scan (determining what kind of moles an individual grows)
3. dermoscopy (closer look)
4. photographic record (checking for changes)

A group of Scottish dermatologists in Glasgow developed a seven-point checklist to determine lesions that should be examined further. The list has 3 major features: change in size, shape, or color – and 4 minor features: inflammation, crusting or bleeding, sensory change, and diameter greater than 7 mm. It is possible that in the first point of Dr. Grichnik's detection strategy mentioned above, that it is some small sensory change that the patient notices that causes them to worry, which is not detectable to the physician.

The American Academy of Dermatology web site[5] does a good job explaining the "ABCD's of Melanoma Detection". The idea is that individuals should perform these self exams periodically. The 'A' stands for Asymmetric. This is in reference to the shape of the lesion. For example, the image below is asymmetric:

figure 1.5 Asymmetric Border Example *

The 'B' stands for Border irregularity. Again, this is related to the shape of the lesion, but in particular the smoothness of the edge is the focus. Below is an example of a lesion with significant border irregularity:

figure 1.6 Border Irregularity *

---

* images from http://www.cancer.org/docroot/PED/content/ped_7_1_Skin_Cancer_Detection_What_You_Can_Do.asp?sitearea=PED

The 'C' is in regard to the lesion color or colors. Lesions with many colors, particularly high percentages of red, black, or blue are indicators of melanoma. The image is an example of a lesion with a large percentage of red and black:

figure 1.7 Multiple Color*

The final metric, indicated by the letter 'D', is for dimension, or size of the lesion. Anything greater than six millimeters is cause for concern.

figure 1.8 Dimension > 6 mm*

Another measure more important for treatment, rather than detection is the depth of the lesion. This has been shown to be a critical factor in both initial recovery and tumor recurrance. In a recent study, researchers from the University of Louisville and Duke University determined that tumor thickness played a role in the recurrence of the cancer.

> "But by far the most important factor seemed to be the thickness of the tumor. In the worst-case scenario (tumor thickness greater than 8 mm), the cure rate for women with tumors on extremities was only 31%. For men this rate was 24%. Those in this group who had a recurrence survived for an average of about five years from the time of first treatment." [6]

The article further underlines, that if a cancer is caught early, it can be treated with surgery. However, once the tumor has spread, it can be very difficult to treat.

Another metric, helpful in initial diagnosis is lesion texture. A lesion with a very bumpy, or rough texture, combined with other symptoms is should be investigated.

1.4 Dermatology Tools

Aside from excising the mole and performing a biopsy (which, given the 200,000 to 1 likelihood of melanoma doesn't seem practical.) There are several non-invasive diagnostic systems. Given the success of public campaigns for getting patients in to have thin lesions examined, and general practitioners attempting to diagnose lesions that are difficult for dermatology experts to correctly classify, there has been significant growth in a variety of non-invasive diagnostic tools.[7]

---

* images from http://www.cancer.org/docroot/PED/content/ped_7_1_Skin_Cancer_Detection_What_You_Can_Do.asp?sitearea=PED

Duke University working with DigitalDerm, Inc. has developed a tracking tool called MoleMapCD that allows high definition viewing and storage of up to 33 images, supporting a whole body scan of patient skin. This system inherently provides an archive of patient mole images that can then be used to track any mole changes with subsequent visits.

Art Papier,M.D., assistant professor of dermatology at the University of Rochester, working with Logical Images, Inc. in Rochester created a software system, VisualDx, with a database of thousands of searchable images. The search engine has been tuned, so users can enter descriptions of the skin lesion, rash, or whatever, and the system will return images that match. This way, the health care provider doesn't first have to turn to textbooks to try to guess at a diagnosis first and then try to find a matching image. According to the report on the UniSci website[8] the software more than doubled the rate of accurate diagnosis.

There are a number of researchers, as well as commercial providers such as Dermlite (www.Dermlite.com) that provide tools for Epiluminescence Microscopy (ELM). This tool allows health providers the ability to not only see the surface lesion, but to also view some of the structures beneath the skin. Basically, the process involves a drop of oil on the lesion site. A special type of dermatoscope or binocular stereo microscope is used (also called dermoscopy, dermatoscopy, or surface microscopy) to view the site. The oil makes the epidermis somewhat transparent by reducing the refractive index mismatch between the corneum layer and the air. The site is covered with a glass slide and the subsurface features are viewed with the viewing device with magnitudes from 10x to 40x.

In some applications of ELM, two filters are used to improve viewing. The first filter is placed over the light source. The filter is polarized to have the same phase angle as the source light, which allows some of the light to enter the skin. The light entering the skin becomes diffuse and is reflected back out from white collagen fibers at the dermis level. The second filter is on the viewing device and is polarized to only allow the light re-emitted from the skin to enter the viewing device. If the viewing filter is set to the same phase angle as the source filter a surface view can be seen by capturing the regular reflection.[9]

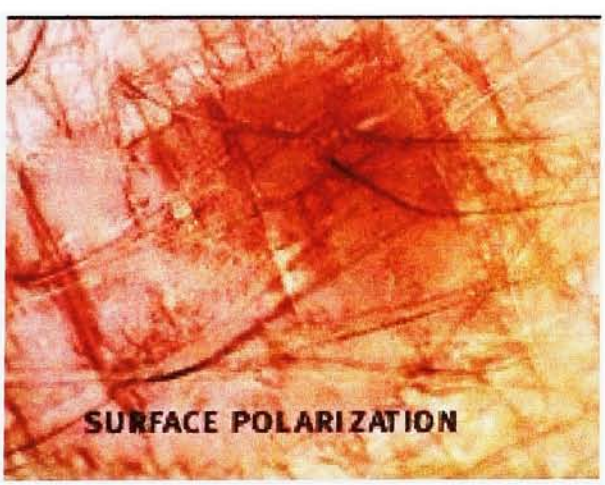


figure 1.9 Surface Lesion*

figure 1.10 Subsurface Lesion image*

The potential of the ELM technology is that it reveals additional metrics that can help the physician provide the correct treatment, which is critical for survival when the lesion is a

* images from http://www.dermlite.com/platinum.html

melanoma. Besides the features already discussed, ELM also allows measurement of an erythmatous blush (seen above in figure 1.9), displacement of blood in the papillary dermis by the lesion, holes in the papillary collagen from the invading tumor, collagen arranged in circular patterns around the tumor, or nodules, and dermal melanin in haphazard arrangements in the lesion area.

Another methodology worth mentioning is called Spectrophotometric Intracutaneous Analysis scope (SIAscope). This was developed by a team of researchers in the United Kingdom and Germany. This device uses both visible and infrared light to extract information about the composition, concentration and position of collagen, blood vessels, and melanin.

"SIAgraphs are obtained by capturing eight filtered waveband images of a skin lesion extending from 400 to 1000nm. These waveband images are then calibrated and act as inputs to a series of computer algorithms that extract information regarding the microarchitecture of the skin [10].
First the algorithm utilizes infrared wavebands to ascertain the quantity of collagen within the papillary dermis for every point over the skin lesion. This is the crucial step for this technique and provides a necessary transformation on the wavebands allowing accurate extraction of total melanin and blood. The total melanin, collagen and blood SIAgraphs can now be displayed." [10]

Clearly, these more in-depth techniques are used when the practitioner is already suspicious. While numbers seem to vary a bit from organization to organization, in general, about 27% of patients who contract melanoma die, which would seem to make it critical to make that initial diagnosis as quickly and effectively as possible so that these further diagnostics can be performed.

1.5 Research Goals

The things this research will not do is to provide a definitive diagnosis. Its meant only as a sort of computer "guess" at what might be a problem. It currently will not provide a searchable database, although it seems like it could be a nice marriage between what Dr. Papier has already done with an auto classifier. The system is not matching features between the pre-diagnosed image database, and the input image…its only matching against feature values that were found to be indications of melanoma.

The system also is not using the many additional features that could be extracted from a subsurface ELM image…this could be very useful, but as mentioned above, it is the initial warning that this research is interested in, before someone gets interested enough to think about using a dermatoscope.

The system was written entirely using the Matlab application and Matlab *.m and *.mat files. Its possible, and maybe probable that the application could work faster or more efficiently if some other languages or platforms were used, although, the average image only took a few minutes to process. That however, was not the focus of the research.

The goal of this research was to investigate an automatic classifying process that would provide clinicians, and possibly the general public an early warning mechanism for worrisome spots on the skin. Imaging technology can do much with a color image of a lesion. There are algorithms for determining the shape of a lesion, histograms to analyze the color content, and texture algorithms that can be used to assign a measure of an image's smoothness. This allows for a system to accept an input, preferably color image and perform feature extractions. These features can then be analyzed and assigned a Feature Warning Level (FWL) that indicates lesions

that should be more carefully examined. In addition, the software can mine the site and display the image and associated feature vectors of the closest match.

1.6 Outline of Thesis

Section 1 explains the purpose of the research: creating a system of feature extraction and matching to explore an automatic diagnostic tool for skin lesions.

Section 2 briefly describes the sources for the data used, and how some data was collected independently.

Section 3 is an overview of previous work in the areas of edge detection, color analysis, and texture measurements. Since these are each huge areas of research and there is a wealth of material, they will be covered superficially, but with enough depth to explain the background algorithms that support the research.

Section 4 details the actual algorithms used and covers the system design.

Section 5 will cover experiments and results.

Section 6 conclusions from the results.

Section 7 will discuss future work that could be done to extend this effort.

Section II : Data Sources

Images were collected either from the web or from photographs taken with a still Camera EOS REBEL G with zoom lens EF 35-80nn f/4-5.6 III. After being developed at both high-end and one-hour kind of processing establishments, the glossy images were scanned in using an Acer ScanPrisa 640U, which is a inexpensive scanner. Several photos were taken in bright sunlight, but most were taken indoors with a flash.

The image database was created completely from images taken from dermatology web sites. This supported a desire for pre-diagnosed images, and also for avoiding any special efforts to produce very high quality images. The images used were exclusively JPEG/JPG, which seemed to be the format of choice for these sites. The web sources for the data can be found in Appendix A.

The following web sites were the prime sources for images:

The University of Iowa http://tray.dermatology.uiowa.edu/DermImag.htm has an absolutely impressive image database. Approximately seventy percent of images were culled from this source.

State University of California at Davis http://matrix.ucdavis.edu/tumors.html was a good source of images, mostly courtesy of Art Huntley M.D..

University of Florida : The Molehill part of the Health Science Center. Images courtesy of Dr. Frank Flowers, MD. http://www.health.ufl.edu/molehill/molehill.html

Homepage of New Zealand Dermatological Society http://www.dermnetnz.org/

Loyola University Medical Education Network which was created by Jeffery L. Melton MD and Jason R. Swanson
http://www.meddean.luc.edu/lumen/MedEd/medicine/dermatology/melton/atlas.htm

The University of Utah with images by John L. Bezzant
http://medstat.med.utah.edu/kw/derm/

The University of Indiana http://erl.pathology.iupui.edu/cases/dermcases/dermcases.cfm

This page provided a very good collection of links for mostly academic sites for dermatology images:

http://www.fammed.wisc.edu/education/presentations/derm/Dermcurriculum.html

Section III : Previous Work

3.1 Introduction

For each of the three main features: shape, color and texture, there are a wealth of algorithms to select from. For the most part, all of these algorithms can segment and identify regions of interest with good accuracy with supervised processing. When working with large databases, however, its desirable to process images without supervision.

In this research, the expectation is that each image in the database will contain a central feature of interest in a fairly uniform, lighter background. The most important first step is to segment out the region of interest, as this is critical to allow examination of the region for shape, color and texture.

In skin analysis, the first hurdle to jump is to find a way around the very textured nature of the skin, i.e. the background. Directly applying an edge detection process to an apparently simple image produces a very noisy result with edge segments distributed uniformly across the image plane.

Since the application is designed to expect a centered image on a uniform, clear background, a good solution is to use morphological operations to filter out the texture of the skin, but preserving the better-defined central object.

3.2 Morphological Operations

The word morphology is from the Greek words morfh and logos, meaning "the study of forms" and in terms of image processing really means mathematical morphology. This process was developed in the early 1960's by two Frenchmen, Jean Serra and Georges Matheron, and was built on the foundations of set theory. Together they developed the discipline of mathematical morphology. In 1964, Serra coined the term mathematical morphology, and the subject was documented in detail in a treatise entitled "Image Analysis and Mathematical Morphology" published by Academic Press in 1988. This continues to be an area of active research. Dr. Edward R. Dougherty (who has ties to at least one graduate student at Rochester Institute of Technology) a professor in the Department of Electrical Engineering at Texas A&M University in College Station has helped advanced this field and produced many papers and at least one text on the topic: An *Introduction to Morphological Image Processing.*

The main operations of morphological image processing are dilation and erosion. The processes are shaped by a structuring element. If an image is complex, the morphological process may require hundreds or even thousands of different structuring elements. One area of interest is to discover how to automatically produce these structuring elements to produce optimal results.

The underlying operations are based on logical AND and OR operations described in figure 3.2.1.

Logical AND	**Logical OR**
0 AND 0 = 0	0 OR 0 = 0
0 AND 1 = 0	0 OR 1 = 1
1 AND 0 = 0	1 OR 0 = 1
1 AND 1 = 1	1 OR 1 = 1

figure 3.2.1

In figure 3.2.2 below, is an example of a structuring element created by Matlab. Figure 3.2.3 shows a table of the various options in Matlab for structuring elements. As you can see in the figure, Matlab always uses the center of the structuring element as the origin. The origin indicates the pixel that will be operated on and potentially altered in the output matrix. The erosion process uses the AND operation. As you can image, in a binary image, this has three chances out of four to replace the pixel being processed with a '0'.

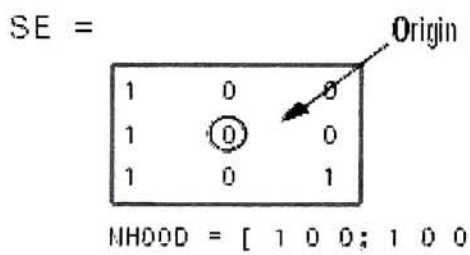


figure 3.2.2*

Flat Structuring Elements	
'arbitrary'	'pair'
'diamond'	'periodicline'
'disk'	'rectangle'
'line'	'square'
'octagon'	
Nonflat Structuring Elements	
'arbitrary'	'ball'

figure 3.2.3 **

* ** figures from http://www.mathworks.com/access/helpdesk/help/toolbox/images/strel.shtml

As you can guess from the table in figure 3.2.3, this process can become very complex. Erosion, and the companion operation, Dilation are probably best described by looking at a toy example. The structuring element is move systematically over the image matrix. In the diagram in figure 3.2.4 a tiny structuring element, B, operates on the toy image A. The output matrix is displayed in C. In this diagram, the structuring element is covering pixels at (2,2) and (2,1). The '1' in (2,2) ANDs to a '1', but the '1' in (2,1) is AND-ed with a '0' and results in a '0'. The next step in the process, is for these two results to be AND-ed again, so the '1' and the '0' are AND-ed and result in a final value for the result pixel of a '0'.

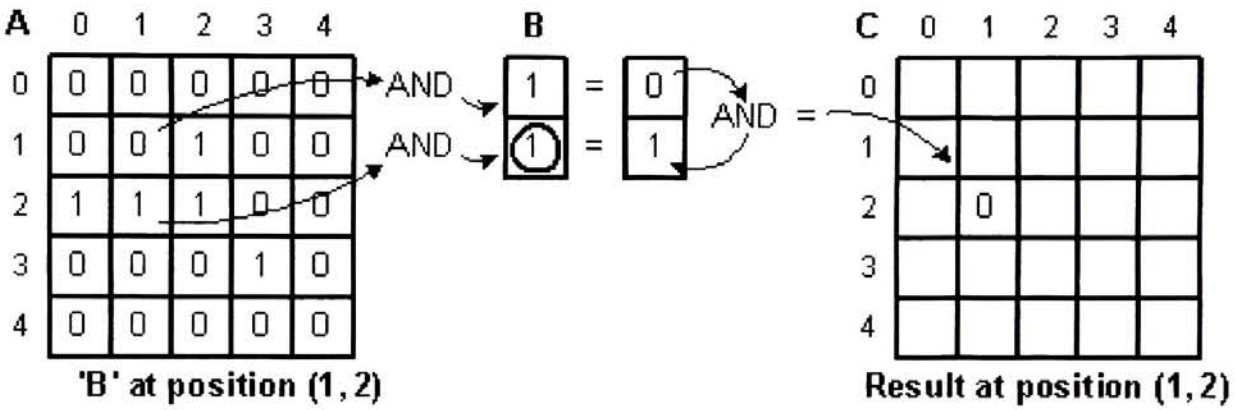


figure 3.2.4 Erosion*

As you can see, this operation eroded the number of '1' pixel to a '0'. In figure 3.2.5, the process has been executed over the full toy matrix. As you can see, the pixels with a '1' value have been eroded down to a single pixel.

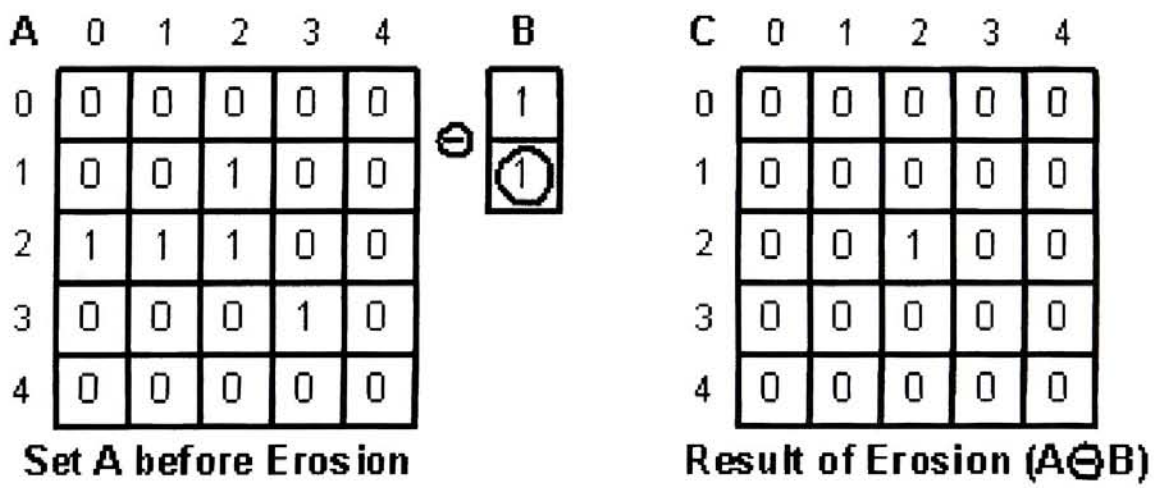


figure 3.2.5 full Erosion operation on Matrix A**

* ** figures from : http://www.coreco.com/web/news.nsf/frm_lbbsarc?OpenForm&Lang=Gb

In the dilation operation, with the logical OR operation, the '1' pixel is preserved as shown in figure 3.2.6. As long as at least one member of the structuring element is overlapping the target set (the pixels with the '1' value) you get a '1' back.

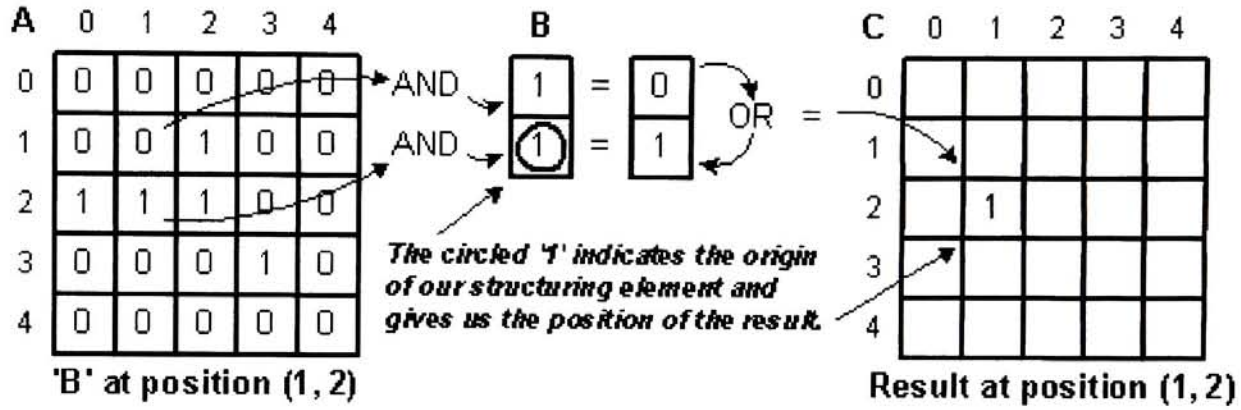


figure 3.2.6 Dilation*

In figure 3.2.7, the result from the full operation on set A is shown.

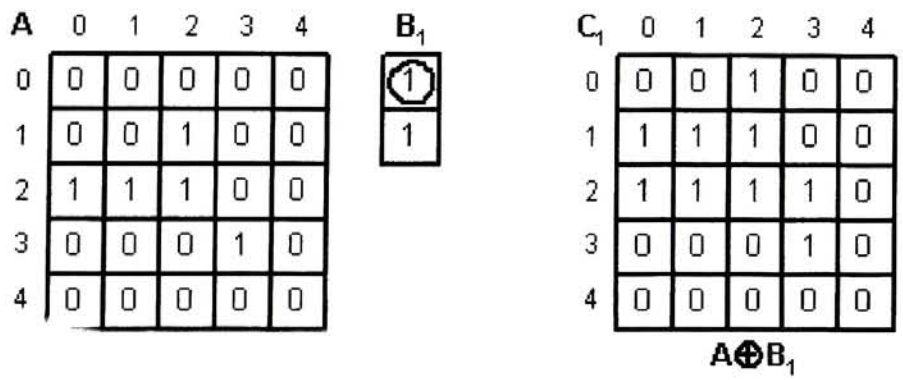


figure 3.2.7 Full Dilation Operation on Matrix A **

There are two other important operations in morphological operations that are combinations of erosion and dilation: the open and close processes. In the open operation, a dilation operation is followed by an erosion operation. This is demonstrated in figure 3.2.8. In real images, this operation is called opening because narrow gaps

* ** figures from : http://www.coreco.com/web/news.nsf/frm_lbbsarc?OpenForm&Lang=Gb

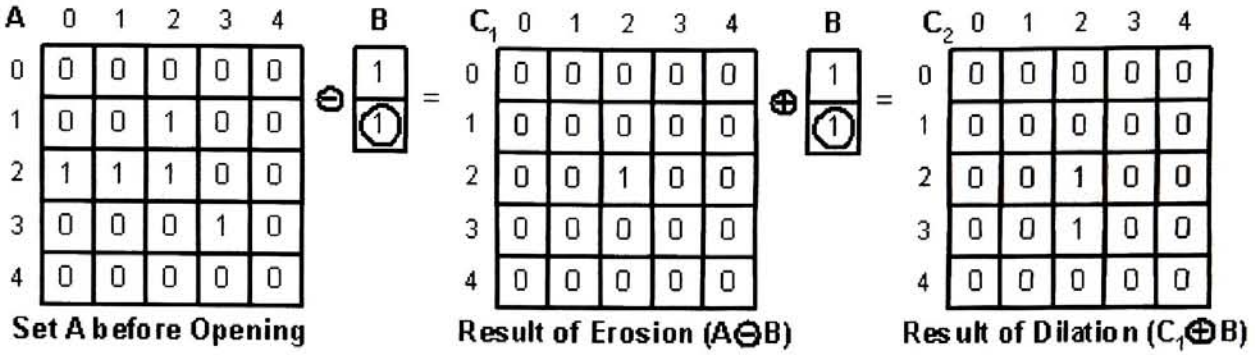


figure 3.2.8 Open Operation *

open up and thin protrusions disappear. In the skin images in this application, that aids in the removal of the edge fragments that appear from the skin's texture. In the close operation demonstrated in figure 3.2.9, the operations are performed in the opposite order.

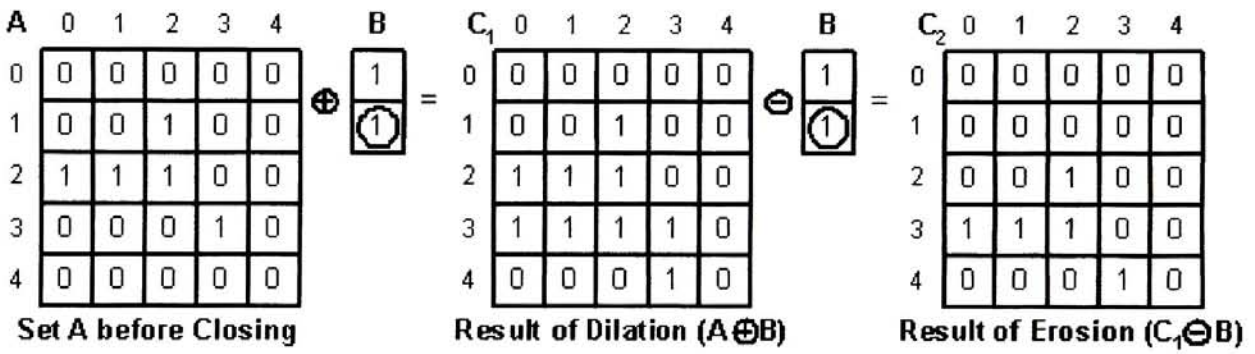


figure 3.2.9 Close Operation **

In this process, in a real image, small holes and gaps are filled in. This can aid in connecting edge fragments in the area of interest that you are trying to segment.

These basic operations are all functions that can be called in Matlab, streamlining the coding.

* ** figures from http://www.coreco.com/web/news.nsf/frm_lbbsarc?OpenForm&Lang=Gb

3.3 Algorithms for Measuring Shape

3.3.1 Compactness

There is a classical topological formula for measuring the shape of an object. It's called the compactness ratio. The formula is:

$$\text{Compactness} = \frac{4\pi A}{P^2}$$

Where A = the area of the region being examined and P is the perimeter of the region of interest. The general idea is that the most compact object would be a circle, as there would be no spaces in the object. In the event that the shape being analyzed was a perfect circle, the equation would be equal to 1 as shown below:

$$1 = \frac{4\pi * \pi r^2}{(2\pi r)^2} \rightarrow \frac{4\pi^2 r^2}{4\pi^2 r^2}$$

inserting the formula for circumference for P in the denominator, and the formula for the area of a circle, A, in the numerator. Since benign skin lesions tend to be round-shaped with smooth edges, these images should produce a lower compactness value than an oddly shaped object.

3.3.2 Convex Hull

One way to describe the convex hull of a set of points is if you wrapped the outermost points with a string. The more technical description is the smallest convex set containing the points.

Matlab supports a function called convhull, which in turn relies on an algorithm called Qhull, which was developed by a team working on a grant from the National Institute of Health.11 The team was composed of C. Bradford Barber (University of Minnesota), David P. Dobkin (Princeton) and Hannu Huhdanpaa (Configured Energy Systems, Inc.). For the most part, this algorithm builds on the Quickhull Algorithm introduced in the late 1970's (Eddy 1977, Bykat 1978, Green and Silverman 1979, and Preparata and Shamos 1985.) The quickhull algorithm is a process that recursively partitions a set of points until no more points can be partitioned. This partitioning step is where the main content of the algorithm is. It works as follows:

The algorithm is given the set of points, S, to be processed, and a line segment AB, whose endpoints are known to be on the convex hull. (A usual choice is a line between the most left and the most right points in the set.)

The points are Partitioned as follows:

- Find a point in the set that is farthest from the given line segment, say C. Use this point to create a triangle ABC by connecting the two end points from the initial line segment to this new point.
- The points inside this new triangle cannot be on the hull. Put them in set inside_S.
- Put the points that lie outside the AC edge in set S_left, and the points that lie outside the BC edge in set S_right.

The process in step 2 is repeated for the points in sets S_right and S_left, and recurrsively in those regions until there are no points left to process in S_left and S_right. At this point, the points on the convex hull perimeter will be identified.

This process works the most quickly on random points since the initial partitioning step will capture a large set of the initial points.

3.4 Algorithms for Measuring Color

3.4.1 Colorspaces

The human vision system perceives colors via structures in the retina called cones. The fact that almost any color can be created from three primary colors is due to the fact that there are three kinds of these cones that recognize red, green and blue. This provides a sort of color alphabet that can be used to recognize a wealth of colors. This fact is leveraged with computer monitors using red, green and blue phosphors, allowing the screen to display virtually any color. It's used again for printer systems, which use cyan, magenta, and yellow inks (CMY). In this case, there are two different combinations because monitors emit light, whereas printers reflect light. When a white light is shined on ink, the color is the component of white light that is reflected and not absorbed by the ink. The Hue, Saturation, and Value (HSV) color system is closer to how humans perceive color. Humans normally think about color in terms of shade, how bright it should be, and if it should be pastel or vivid. Usually the first thing humans notice is the hue of a color. Hue defines the shade and where the color is found in the color spectrum.

This combination of three values is called a color space, and the shape of this space is often described as a cube. Although theoretically, any color space could be created by using independent colors but those color spaces are not identical to other colorspaces. This makes it a bit impractical for a real system. There will be a part of the color range that can't be reproduced in another colorspace. Some other colorspaces that will convert back and forth are YIQ or IYQ, and HIS (or ISH) which stands for Hue Saturation and Intensity. The YIQ colorspace is useful in color TV broadcasting: it converts to RGB easily, and the Y component supplies all the information needed for black and white broadcast. We'll look a little more closely at RGB (since that was used in this research) and HSV as a more detailed example of a different colorspace.

3.4.2 RGB

In this space the image is composed of 3 matrices for each of the primary color components. The color values range from 0 to 255. If all three values are 0, the color perceived will be black. If all three values are at or near 255, the pixel will be displayed as white. Values between 1 and 255 yield gradations in the color intensity. Pixels where the values are equal for the red, green and blue components will display as grey. As these equal value progress from 0 to 255 the grey will become lighter and lighter. A graphic representation of the color cube is depicted in figure 3.4.1. Note the diagonal

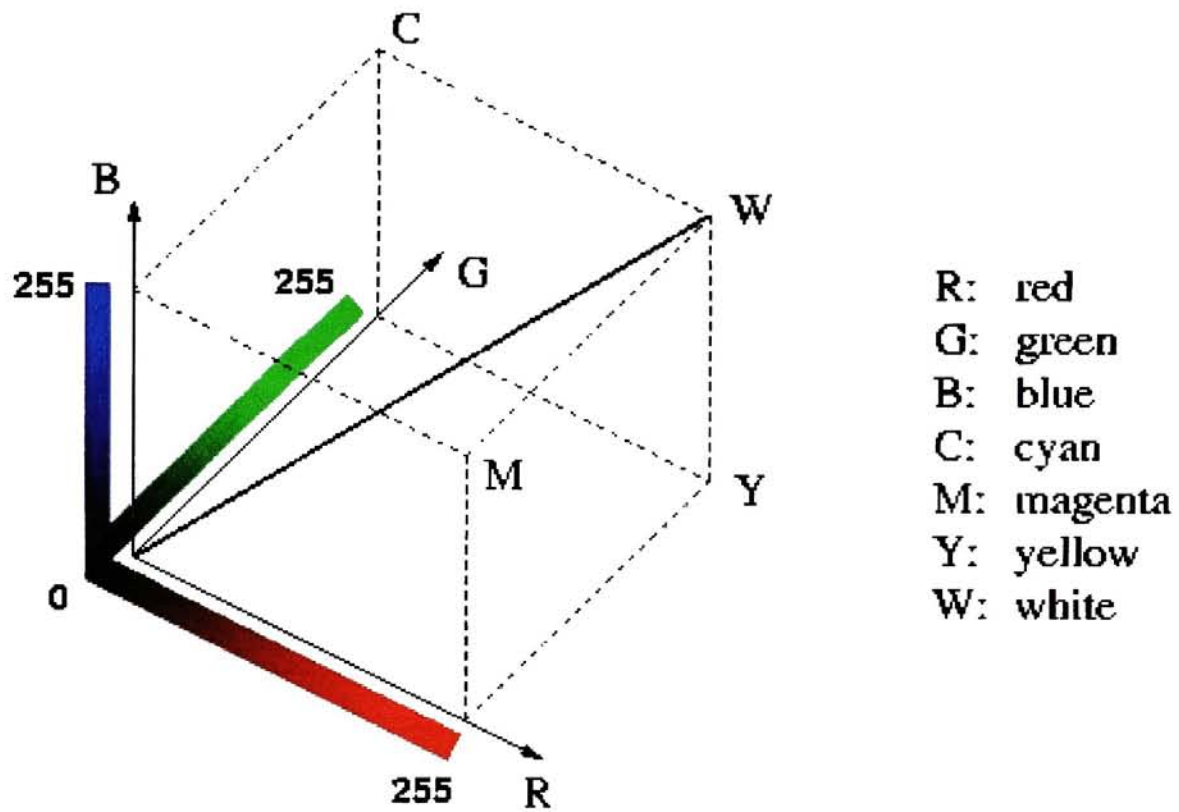


figure 3.4.1 RGB color cube*

line between 0 and W or 255…this indicates the grey line, where all color components are equal. This is referred to as the neutral axis.

3.4.3 HSV

As mentioned previously, this color space matches human perception most closely. The human vision system first notices hue and then saturation and value. In the HSV system, the saturation value describes how pure the hue is with respect to white. A color that is all blue with no white is fully saturated. If you add some white pixels to the image, the color shifts from blue to light blue. The hue is still blue, but its become less saturated. Lastly a color has a certain value. If you see a bright red poppy in full sunlight

* figure from http://gimp-savvy.com/BOOK/index.html?node50.html

and then again in the evening you will note that the color appears duller. Value and Saturation are represented with values from 0 to 100. If the value is a low number the color will appear dark. If the saturation is a low number the color will appear lighter or more diluted. Hue is represented by values from 0 to 360. Looking back at figure 3.4.1, you can see that red, yellow, green, cyan, blue, and magenta are distributed equally in angle around the neutral axis. The wedge defined by the neutral axis is a plane of constant hue. Since hue is a function of angle, the range will be from 0 to 360 degrees. (Although hue is constant, brightness and saturation will vary over the range of possible values.) Figure 3.4.2 depicts the HSV color space.

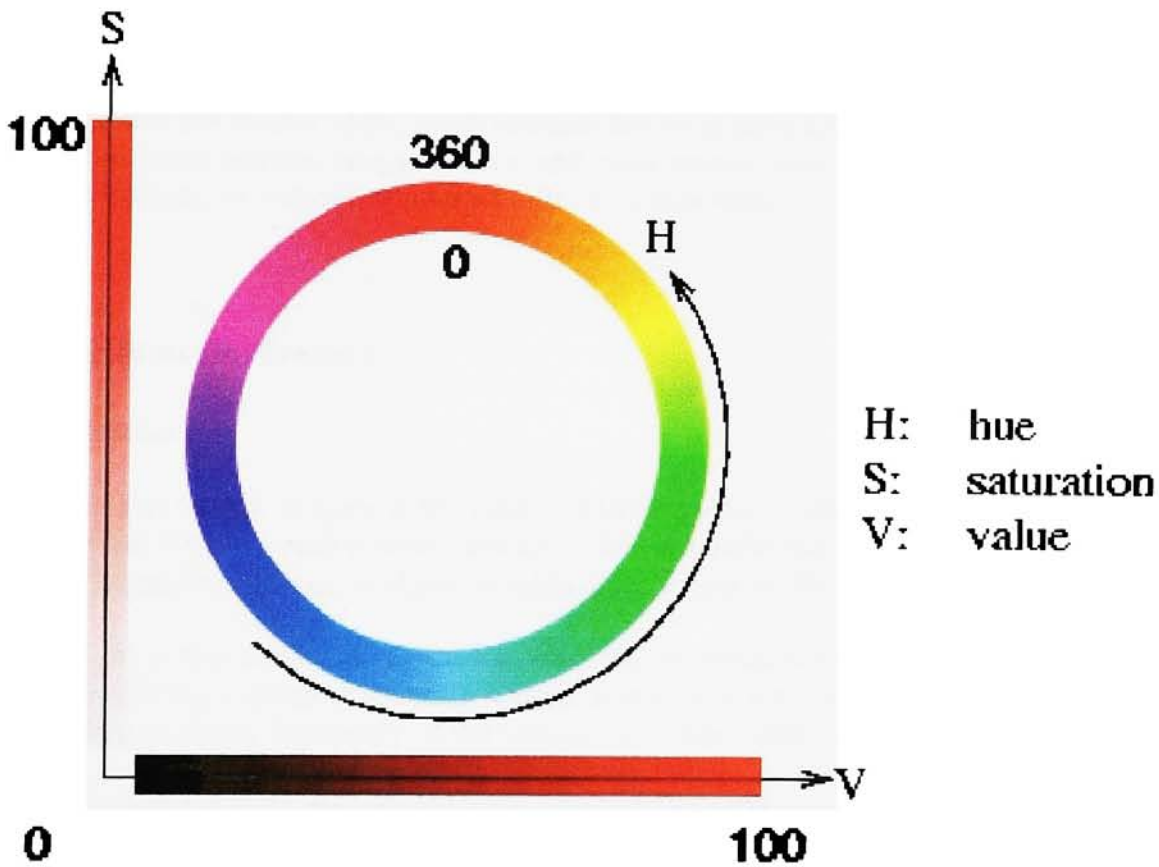


figure 3.4.2 HSV Color Space*

3.4.4 Value of Color Information in Image analysis

In 1991, Michael Swain and Dana H. Ballard published a paper in the International Journal of Computer Vision called "Color Indexing". While much of the imperative for doing this research was in support of robotic systems it turned out to have broad impact on image analysis. At this time, color was, apparently not being closely explored.

"The ease of recognition using color strands is contrast to the neglect given recently

* figure from http://gimp-savvy.com/BOOK/index.html?node50.html

to color as a recognition cue…Instead, much more attention has been given to geometric algorithms that extract shape from stereo, motion, and lighting cues." [12]

Then a bit later in the paper:

" …Geometrical cues will be the most reliable of object identity. While this may be generally true, it may not be true for routine behavior (Chapman, 1990). In such behavior, wherein familiar objects are interacted with repeatedly, color may be a far more efficient indexing feature." [13]

The thrust of this paper was to use color histograms to identify objects in an image where there was some knowledge of the approximate location of the object. This then allowed the researchers to compare a "model histogram" with the histogram from the region of interest in the image. They called this "Histogram Intersection". While this methodology was not used in this research, and in the twelve years, much research has been done using color to perform edge detection, segment images, image retrieval and color texture analysis, the concepts of using color to identify objects, or features of interest influenced this work.

3.5 Algorithms for Texture

3.5.1 **Definition**

There seems to be a struggle in the field to find the perfect definition of texture, probably due to the fact that it covers such a broad concept. Here is a definition from a Computer Vision textbook, Image Processing, Analysis, and Machine Vision by Milan Sonka, et al.

" We might define texture as *something consisting of mutually related elements*; therefore we are considering a group of pixels (a texture primitive or texture element) and the texture described is highly dependant on the number considered (the texture scale)[Harlick 79]." [14]

Texture is created from smaller elements, often called primitives or texels. Texture description is scale dependant (think of a woven fabric printed with flowers.). The main goal in texture analysis is to accurately represent the texture (or conversely, texture recognition) and also texture-based shape analysis. A texture primitive is a contiguous set of pixels with some average intensity, size, shape, and there is something that can describe their spatial relationships.

This visual texture is really variations in gray or color levels in an image. These textures can be described based on the pixel intensities in a primitive (tonal properties) and structure, which can be determined by pixel relative locations to each other.

If the texture primitives in the image are small, but have tonal differences, the texture will appear to be fine grained. Larger primitives will present a coarser texture. There are also strong (the spatial relationships between primitives is regular) textures and conversely, weak textures. Weak textures have small spatial relationships and can be handled well by statistical methods.

Texture algorithms, of which there are many, tend to group into two main approaches: statistical and syntactic. Statistical algorithms are most commonly used and work best if the texels map well to pixel sizes. Syntactic texture analysis is based on an analogy between the

texels spatial relations and the structure of a formal language. A grammar is constructed for each texture class. The analysis is based on using the grammar to detect attributes and regions of interest.

There is also some newer work in texture recognition based on some work by Bela Julesz.

> " Research on pre-attentive (early) vision [Julesz 81, Julesz and Bergen 87] shows that human ability to recognize texture quickly is based mostly on textons, which are elongated blobs (rectangles, ellipses, line segments, line ends, crossings, corners) that can be detected by pre-attentive vision, while the positional relationship between neighboring textons must be done slowly by an attentive vision sub-system. As a result of these investigations, another group of methods has begun to appear, based on texton detection, and texton density computation [Voorhees and Poggio 87, Ando 88].[15]

In this research, skin mostly presented a weak texture, and statistical methods appeared to be the best fit to the data. Of the many algorithms available, the co-occurrence matrices seemed to be the best fit to the data. In addition, Matlab is extremely well suited to handle matrices, so the platform software made it a good choice as well. This method is based on a repeated occurrence (vs. some general characteristic) and this indicates rapid change for between pixels in a fine texture and slower change for coarse textures.

3.5.2 **Co-occurrence Matrices**

Interestingly, the development of texture analysis and co-occurrence matrices was sparked by Bela Julesz, as mentioned in the quote above, he developed the idea of textons, which he called the fundamental elements of vision. Julesz wasn't really a computer scientist...he was a network researcher at Bell labs and later a professor of psychology at Rutgers university. He was very interested in how human visual perception worked and introduced the notion of detecting texture based on second-order statistics. He originated the concept of looking at pairs of information to detect order.

This sparked computer scientists to create algorithms to operate on co-occurrence matrices, one of the early researchers was Robert Haralick. Its interesting to note how young computer science really is, that these early researchers are still working in the field. Often these texel patterns are referred to as Julesz ensembles. A texture, or a Julesz ensemble is described as an equivalent class of mini-images on a 2D plane that share equivalent statistics.

An image co-occurrence matrix is a record of pixel relationships, that is from one pixel to another in a grey level matrix. (Where the concept of duality comes from that runs through Julesz work.) The co-occurrence matrix can capture information about pixel relationships driven by distance between pixels and also by direction. So, for example, you could create a co-occurrence matrix of information based on pixels that are ten pixels apart, and to the left of the pixel of interest. Commonly the direction is expressed as North, South, East and West. A variety of these matrices can be created to synthesize a statistical picture of the texture in an image. See figure 3.5.1 for a simple example of how to create a co-occurrence matrix. Notice that the size of the co-occurrence matrix is driven by the highest grey level value in the original image matrix.

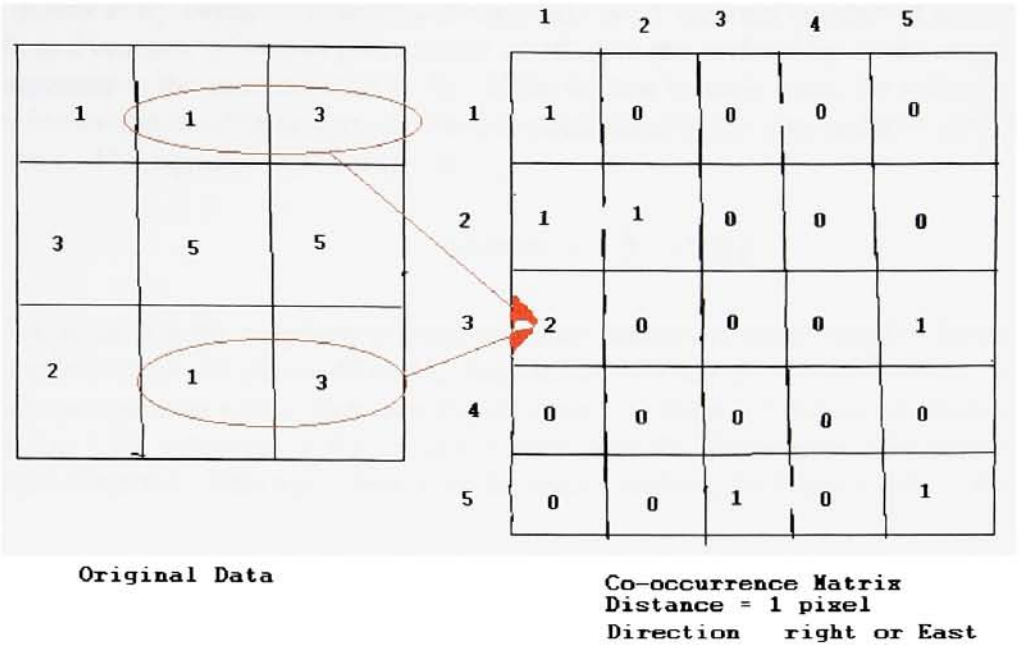


Figure 3.5.1 – Graphic depiction of a simple co-occurrence matrix.

So in the depiction, the highest pixel value is five, so the co-occurrence matrix is five by five. The other crucial information is the distance, which tells the algorithm how far apart to look for pixel pairs, and direction. If this were a symmetric co-occurrence matrix, the direction would be both east and west…and in that case the co-occurrence matrix would have a '2' in position (3,1) as well as position (1,3).

Some pre-knowledge about the images being examined usually shapes the co-occurrence operators (direction and distance). For example, if you know you are looking at something with a definite direction, like trees in a forest or hair – that would produce very different matrices depending on direction. If you have a weak or blurry texture, you might want to view pixels farther apart to find better matches (i.e. stay inside individual texels.) Haralick [2] presented some definitive algorithms, which are considered standards now. Those algorithms are energy, entropy, maximum probability, contrast, inverse difference movement, correlation, and run length. Later work by Gose, et. al. defined another algorithm for measuring distance. These algoritms or feature extractors germane to this work are examined below.

Energy measures the evenness, or smoothness of an image. The theory is that, if there is a high count of pixels with the same values, then the matrix will have high counts for pixel groups. So when you add up all the counts in the co-occurrence matrix, the higher the value, the smoother the image. Images with rough or coarse textures will have smaller numbers because there will be smaller pixel combination counts. Where M is the co-occurrence matrix, the equation for Energy is:

$$Energy = \sum M_{ij}^{2}$$

In the equation for calculating Entropy (or the amount of disorder in the image) the probability of a matrix entry being incremented is summed. So if the total number of counts in the matrix is 1000, and one pair of values gets a count of 10, then the probability of the count being incremented in the next iteration is .01. If the texture is fairly even, the entropy value will be high, indicating that the texture can be discerned visually (as opposed to a very disordered chaotic texture.). The equation for Entropy is:

$$Entropy = -\sum p \log p$$

The algorithm for weighted average absolute distance is measuring the distance of co-occurrence matrix M entries from the diagonal of the matrix. So for rough textures, you'll have a lot of combinations where there is a big difference in the pixel values reflected near the matrix edge (i.e. 1,5). However, if the texture is pretty smooth, distances will be smaller and will cluster near the diagonal. This algorithm tests the matrix looking for those trends. The formula is:

$$d = \frac{1}{M} \sum |i - j| m_{ij}$$

where

$$M = \sum m_{ij}$$

The last feature extraction formula is for contrast. This analysis determines how much difference there is between pixels. This give some notion of local image variations. This formula is:

$$\sum_{i,j} |i - j|^k \, P^{\lambda}{}_{\phi,d} \; (i, j)$$

Usually k = 2 and $\lambda = 1$, ϕ = direction and d = distance.

3.6 Other Work in Unsupervised Image Analysis

3.6.1 Overview

There is keen interest in being able to extract meaningful results from not just images, but from image databases. Technology has allowed us to create large databases of information, but there are not enough robust, accurate methods yet to make sense of all that data. This need can be anywhere from a commerical application to allow the consumer to select all images of a certain person or a certain event, or maybe more critically, to assess medical or scientific information from existing databases. Most systems accept some "steering" information to tune these extractions. Below are some samples of some of the work in this area.

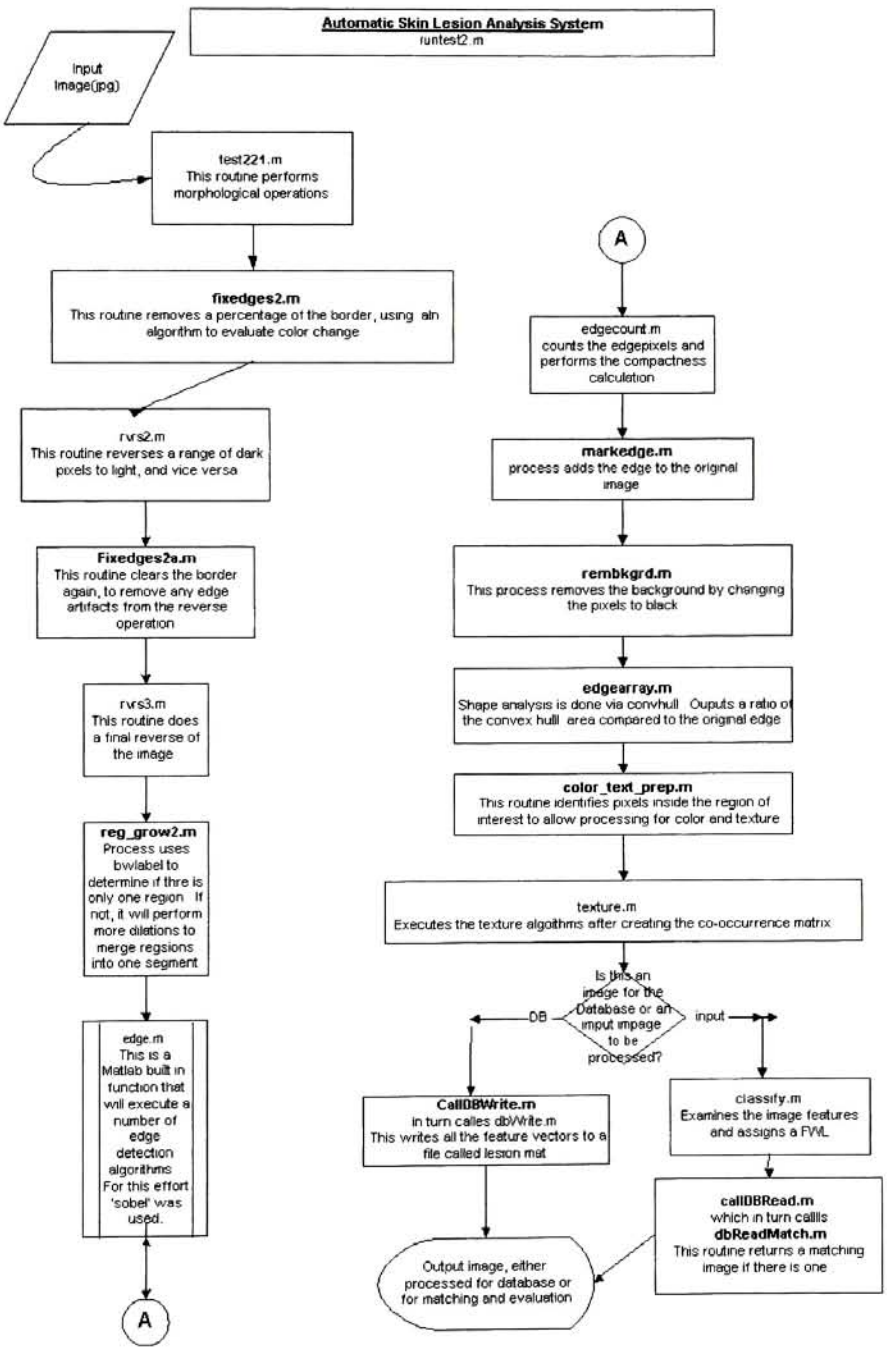
For example, in a paper by Eli Saber and A. Murat Tekalp, "*Integration of Color, Edge, Shape and Texture Features for Automatic Region-Based Image Annotation and Retrieval*", they accept direction for keywords to assist in a search. Because the thrust of their work is more generic, their system has a guidance systems for developing keywords to direct the search (like the keyword "grass" can be assigned to images that contain a green color and grass texture.). Their application also allows users to determine if they want feature vectors from color, texture, and/or shape. While this system is automatic in the sense that it performs the search without tuning during the search (although they do provide the user with the ability to tune combinations of texture, shape, and color), it requires some intelligent intervention during the data indexing to create useful keywords.

In another paper, " Initial Results of Automated Melanoma Recognition" by H. Ganster, M. Gelautz, and A. Pinz both color segmentation and shape analysis of Epiluminescence Microscopy (ELM) are used to analyze color images. In this work, the authors use morphology and color segmentation to find the edge of skin lesions, but they don't apply any texture algorithms. In this research, they focused on the color components in the lesion, as malignant tumors present characteristic color variations, like deeper pigment near the edges and tiny dark spots.

In a final example, Dr. Scott Umbaugh presents his work on automatic skin tumor border identification in his book, Computer Vision and Image Processing, A Practical Approach using CVIPtools. The focus of this work, is to again use color segmentation to identify the tumor border, and assess the shape, since again, melanoma often presents characteristics that allow classification. For example, any lesion that has a very irregular border with protrusions and indentations is a concern, however in the material presented in the book, he didn't pursue any algorithms to analyze the results, as the focus was on the success of finding the border.

Algorithm Section

Application FlowChart

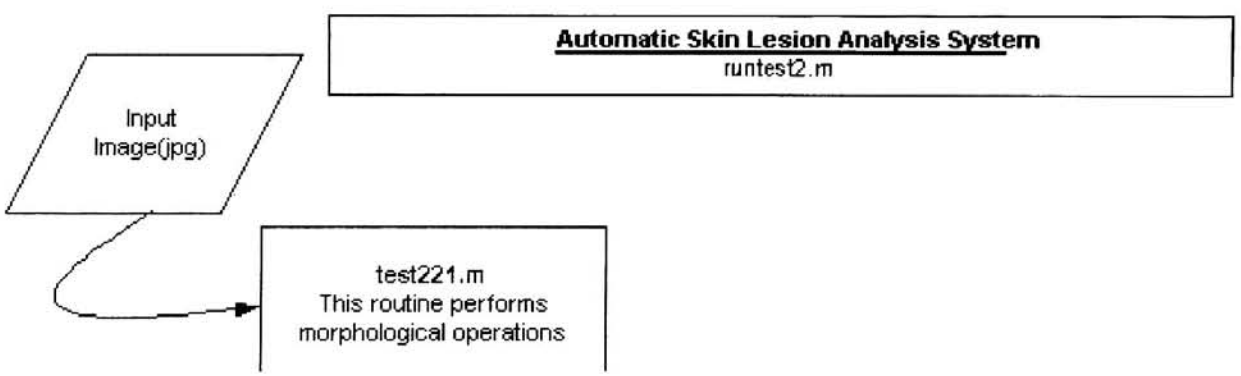


Section IV : Automatic Lesion Classification

All of the work on skin lesion analysis reviewed so far appears to be focused on performing analysis on image databases of pre-diagnosed lesions, or enhancing searches into these image archives. According to Dr. Art Papier, of the University of Rochester Dermatology department, "If a patient says a mole is itching but the photo shows no change you still remove it...". This highlights the fact, that if a dermatologist is going to the trouble of using some extraordinary measure to evaluate the lesion, like viewing it with Epiluminescence Microscopy, they will probably remove it as part of the treatment. What is truly at stake here, is early detection. Skin cancers can be completely removed, and the patient cured theoretically every time if it is caught early.

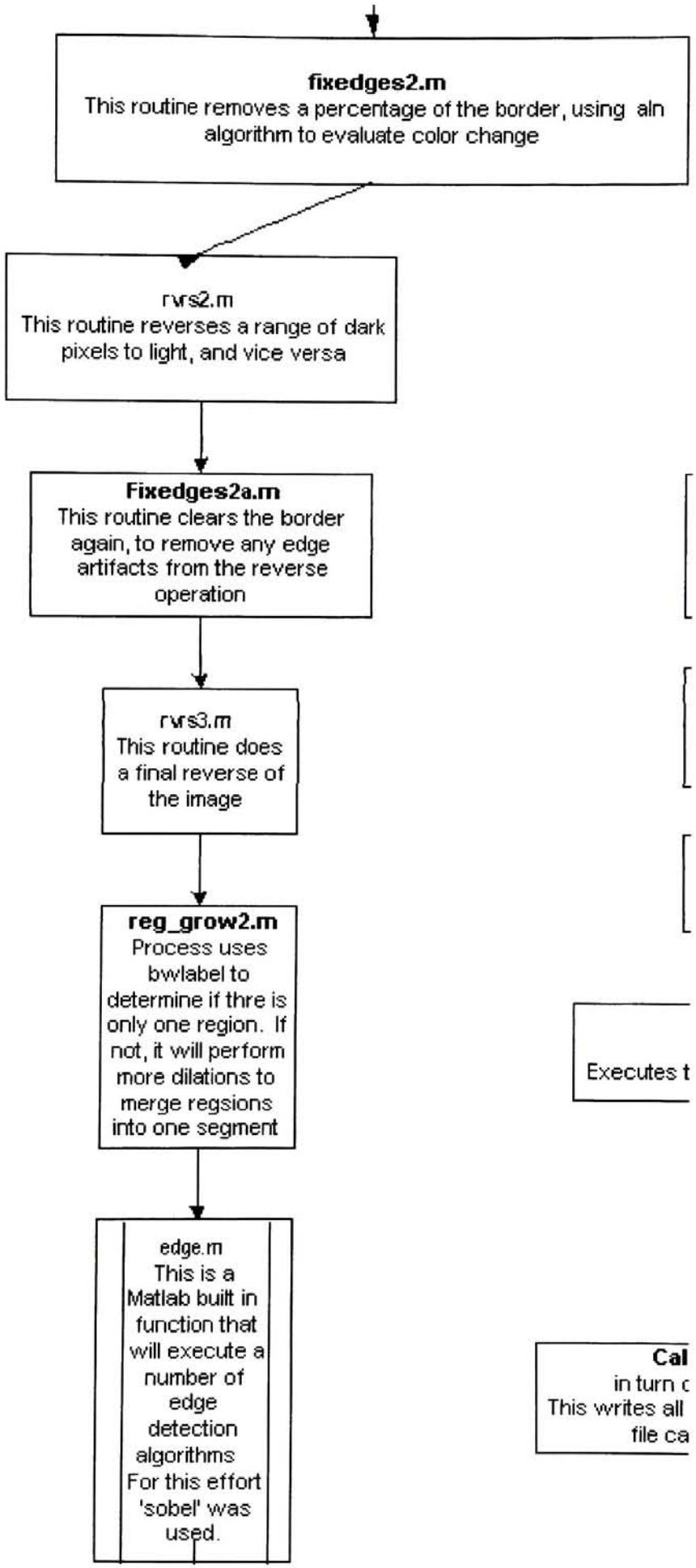
Dr. Papier was also asked how often skin cancers were found on the back. Often this is the area of the body most often sunburned, and least likely to be detected if a melanoma did begin. He replied: " Frequently, very frequently". In addition, he pointed out that history is as important as the observable features, and that photographs had proven helpful in tracking lesions, which are usually very slow growing, so change can be hard to detect.

What this work attempts to do, is provide an automatic classifier, requiring no tuning or steering that will record ten feature vectors about the lesion. This could serve as an immediate tool for analysis, but more importantly as a tool for historical compares. In addition, the images for a small database were deliberately pulled from the web, to try to avoid building a solution that only worked on high-quality, high-resolution images. In addition, input photographs were taken with an average 32mm camera and scanned in with a low-end scanner. Code was kept as simple as possible to try to avoid long processing times.



4.1 Morphological Operations

The first step is to isolate the lesion from the background. This was resolved through a trial and error period until a combination of close, open, and dilate operations were found that had good success in removing not only the skin texture, but quite often dark hairs if they were sparse. First the color image is converted to grayscale and then reversed via imcomplement. Next a disk shaped structuring element is created seven pixels wide. The following operations are then performed: two successive close operations, an open operation, and then ten successive dilations. Next the structuring element is reduced to three pixels in width (This is still disk shaped, that works well, since most lesions have rounded edges.) and three open operations are performed. Figure 4.1.1 demonstrates these operations on a basal cell carcinoma lesion with many thin, dark hairs.


fixedges2.m
This routine removes a percentage of the border, using aln algorithm to evaluate color change
rvrs2.m
This routine reverses a range of dark pixels to light, and vice versa
Fixedges2a.m
This routine clears the border again, to remove any edge artifacts from the reverse operation
rvrs3.m
This routine does a final reverse of the image
reg_grow2.m
Process uses bwlabel to determine if thre is only one region. If not, it will perform more dilations to merge regsions into one segment
edge.m
This is a Matlab built in function that will execute a number of edge detection algorithms For this effort 'sobel' was used.
Executes t
Cal
in turn c
This writes all
file ca

4.2 Image Border Processing

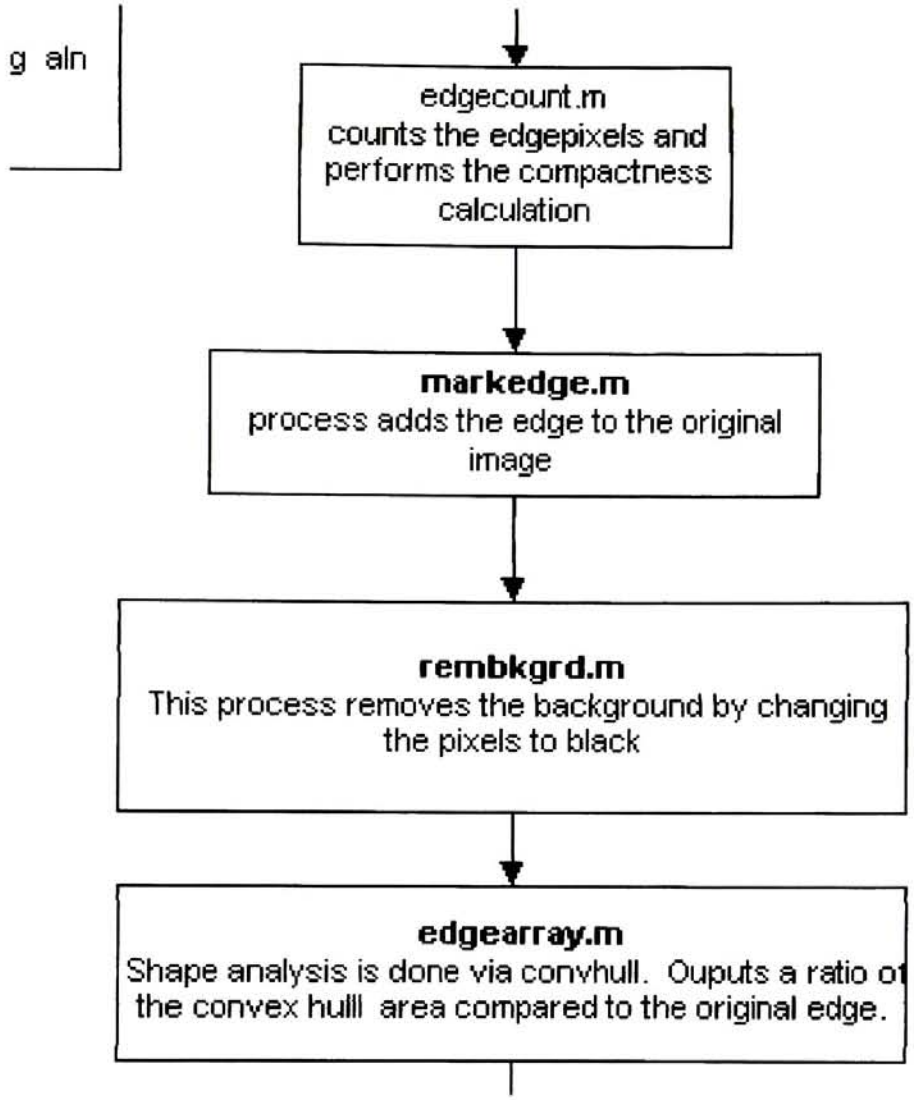
Since the application currently assumes the lesion is in the center of the image, some pains are taken to clear a certain percentage of the edge, to ease later processing, and to improve chances of a good segmentation. The mid point of the image is obtained by dividing the length and width of the matrix. Four points are calculated on the edge, called North, South, East, and West. In addition, the color at the mid point is compared to the color of the first pixel in the image (i.e. m(1,1,1). This color difference is compared to the average color of the pixels to check for a gradient change, as pixels are incremented toward the center in the four directions. The "creeping" in toward the center is to determine how much of that particular edge to remove. The movement stops when either, the color change is detected, the pointer has moved past ten percent of the image width or height, or a significant value change has been detected in the color value from the average pixel color in the chain. Once these depths have been determined, the routine changes the pixels beyond these boundaries to white, leaving a white frame around the image.

The next routine manually changes the lighter values to dark and vice versa. There is a Matlab function to do this, but this process allows for more flexibility in pixel values to be consider dark and which to be white. The routine attempts to determine which pixels have the highest value and turn those white and all others black. This is helpful for fainter images. In both the previous routine and this one, a dilate is performed using a very small structuring element to smooth away and last stray edges.

The massaged image from above is fed into a process that attempts to remove the white frame placed in the image by the previous operation. This algorithm expands the edge values slightly larger than the previous operation, turning all the pixels black – effectively merging them with the existing black background. The final step in the edge massaging is to reverse the image again, so the segmented lesion is now black on a white background.

The application now attempts to determine if the lesion has been segmented as one object. The next process uses a Matlab function called bwlabel. This routine will detect and number various regions, and return the number of segments it encountered. If the number is greater than 1, the program will iteratively call imdilate to try to merge the segments. Once that is complete, the routine calls another Matlab function called imfill, which as it sounds, fills in small holes in an object. Lastly, the program checks to make sure there are some lesion pixels left, and the lesion itself wasn't considered a hole and filled in.

At this point, the lesion should be segmented well enough, and the background removed enough so that and edge detection routine can be applied. For this application, the sobel operator was used. Several operators were tested, but Sobel performed the most reliably and with good speed. At this point, the preprocessing is complete, and we should now have an edge, and can begin analysis for Shape, Color, and Texture.



4.3 Shape Analysis

Interestingly, the shape feature values turned out to be two of the most reliable feature values. The first is a simple calculation. The object pixels are counted up and used for the area value of the compactness formula. The edge pixels are also counted up and squared, the edge pixel count is divided by the area/object pixel count to produce a compactness value.

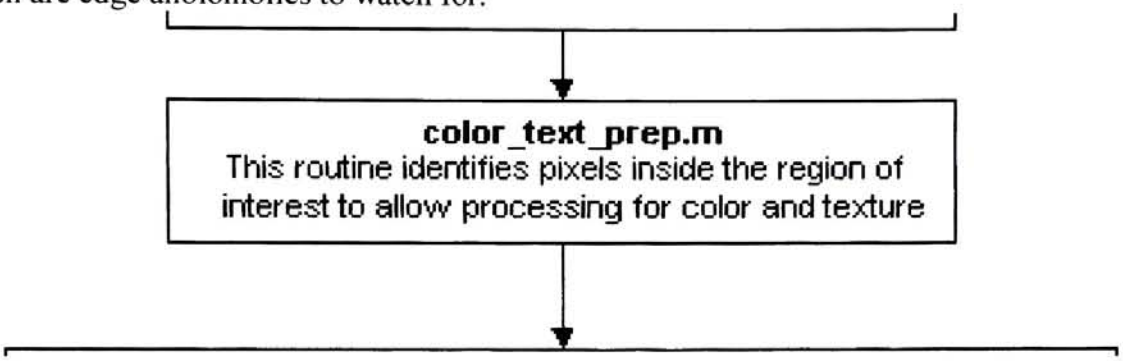
The second shape algorithm is more complicated. The function expects to be passed the edge image. In this image all pixels are black except the edge pixels, which are white. The image is read in, and all the x and y coordinates for the white pixels are stored in and X and Y array. These arrays are then passed to another function that starts with the first pair in the arrays, and performs a closest neighbor search to order the pixels. Once that is complete, the first pixel pair is added to the arrays to close the gap, and allow Matlab to use the pixel arrays to plot the complete edge image.

Several very powerful Matlab functions are now used to get some metric of how irregular the border of the lesion is. First, the ordered X and Y arrays are fed into the convhull function. Next the arrays are fed into the polyarea function, which will provide an area value for the object. The area for the convhull object is also calculated. The original object's polygon area is divided by

the convex hull polygon (We expect that to always be either bigger or possibly the same, but never smaller.)

$$\frac{Area\_Original\_image}{Area\_Convex\_Hull\_image}$$

Since both of these metrics are calculations about percentages relating to the image itself, they don't need to be normalized to compare these values to other images. This gives us a little information about how rounded the object is, but especially about protrusions and indentations, which are edge anolomolies to watch for.



4.3 Color Analysis

The goal for both color and texture analysis, is to be able to focus on the pixels inside the lesion, and ignore the rest. This can be difficult in Matlab, because most data structures need to be in a rectangular matrix. To get around this problem, the function InPolygon was used. InPolygon requires that the X and Y coordinate values be passed in as separate arrays for an input image. So how do you do that? If you look at a simple matrix, you can see that the X values will be a repeating series as the matrix is scanned horizontally. For example, in the tiny

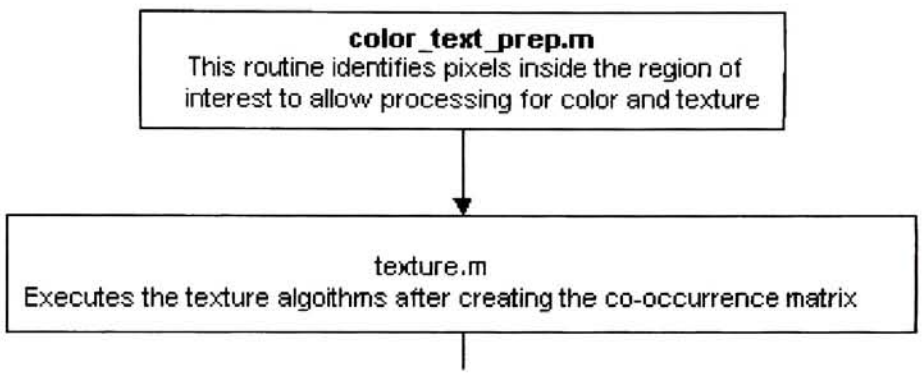
$$(1,3)(2,3)(3,3)$$
$$(1,2)(2,2)(3,2)$$
$$(1,1)(2,1)(3,1)$$

matrix above, the X array would be [1 2 3 1 2 3 1 2 3]. This is accomplished very easily and very quickly in Matlab using the colon operator, in this case it would be X = [1:3]. In a flash Matlab will create the little matrix [1 2 3]. A simple loop, the size of the number of rows will quickly create the X array. For the Y solution is not as easy. Matlab has a function called repmat, that will generate repeating matrices. This was done using a loop, and turned out to be extremely fast. Using our toy example again, what we'd want Matlab to generate would be the series [1 1 1 2 2 2 3 3 3].

These X and Y arrays are again ordered, and passed in to inpolygon. A slick Matlab function that checks each coordinate, and for those inside the image edge, will pass back a '1' – for coordinates outside of the image edge, the function passes back a zero. Using this output, it's now possible to only consider pixel values that have be identified as inside the lesion.

For color analysis, using the coordinates from the inpolygon array, the algorithm maps into the color image and splits each color pixel into the RGB values. 255 are added to the green pixels, and 510 is added to the blue pixels. This allows the algorithm to uniquely identify each color, as the values are added together to allow counting of how many unique colors there are in an image. The sums are added into a "color bin" array. (Otherwise, a color composed of 105 (R) 255 (G) 68(B) , and another, 68 {R) 105(G) 255(B) would appear to be the same color.)

The process also looks at the RGB values to try to determine if the pixel is red, black or green. Pixels with all three color planes with a very low value would appear black in the image. If the red to green ratio is less than .50 and the red to blue ration is less than .50, the pixel would appear red. These two colors are very often evident in a cancerous skin tumor. Lastly, it can be of value if there is pink in a lesion, often this is an indicator of regression, where the body is able to partially heal some of the damage from the cancer. In these cases, pink pixel values have a very high red value, above 195, and the red and blue values are between 87 and 90. Again, these measurements don't need to be normalized, because they are percentages of the image.

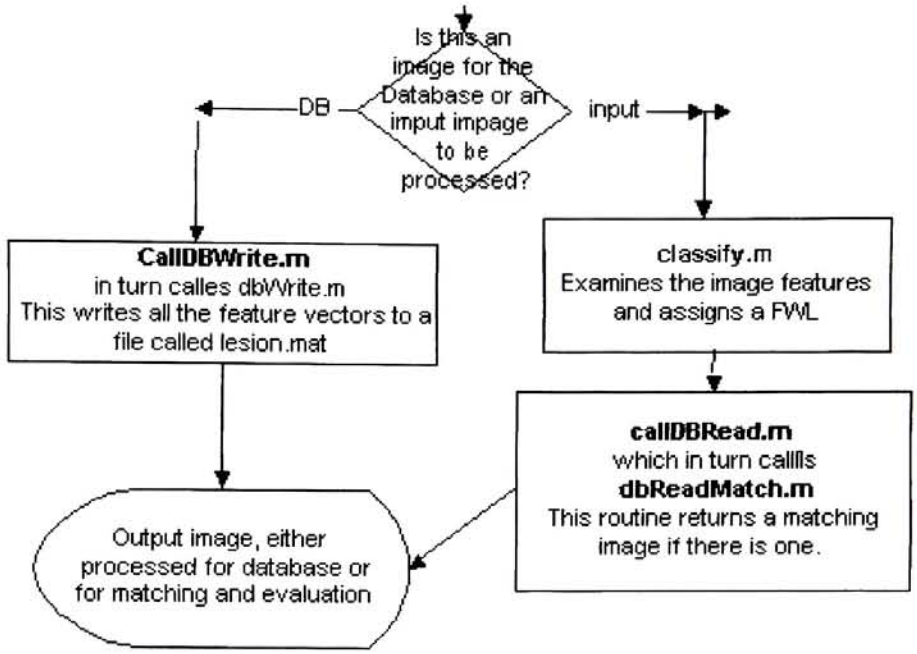


4.4 Texture Analysis

The same array out of inpolygon can be used for texture analysis. The texture values are the one set of feature vectors that would need to be normalized to allow comparison's between images, however the approach in this routine is to create a sub-matrix from the center of the image of a set size for every image. In this series of experiments, the matrix size was 129 X 129. This submatrix was created by first finding the minimum X,Y values in the inpolygon array. This value was then used to offset the output array values. If the lesion area is smaller than 129 X129, the application uses the imcrop function to select pixels from the middle of the image and creates a mini- texel. This is then used to fill in any small areas of white space.

Once the submatrix is created, the process then reduces the pixels to 10 grey level values. These values are all shifted up by one, as the grayslice Matlab function does produce zero values, but later we'll want to use those grey level values for indexing into an array. With that in mind, the gray levels are all shifted up by one.

In the last texture routine, the co-occurrence matrix is created, using a distance of 10 pixels, and a direction east. This resulting matrix is then passed to the energy, entropy, distance, and contrast functions.



4.5 Classification

This is a simple routine crafted based on the results from processing the image database. Ranges were observed for the leading feature vectors, and a simple decision tree was devised to assign the image one of 3 ratings: red, amber or green . The ratings are based only on the most reliable feature vectors: number of colors, the ratio for edge area to convex hull area, texture distance, and compactness. Ranges were determined for melanomas in the database, and set for these three comparisons.

4.6 Database Read, Match and Write

The next process is determined by whether the process is being run to enter data into the database or to evaluate an input image.

If the intent is to enter the data into the database, a simple script is called to write the file name, diagnosis, and the 10 calculated values to the Matlab file. The data is written with double precision, except for the character data, and the process terminates.

If the purpose of the process is to evaluate an input image against the database, the read/match routine is called. This routine loops through the records in the database and compares the database values to the input computed values. In this approach, the values for number of colors, compactness, image edge to convex hull edge ratio, texture entropy, and texture distance are all summed together, for both the input and the database image values. For the most part, this causes number of colors (the largest number) to be the arbiter of the match. (Interestingly, using a voting scheme checking each value, turned out to not be as effective. In this routine, the values are each checked against a minimum value. If the new value is smaller (i.e. the database feature vector is closer to the input value, the new value becomes the new value. This voting routine is called dbReadMatchVote.m) In addition, a min-value field is kept to determine which of the database records had the total closest to the output record. After this comparison is done, the "matching" image – or rather the image with the closest value to the input image is compared to minimum /maximum values to ensure that the match is reasonably close. The minimum and

maximum ranges plus or minus twenty percent of the input sum. For images that don't find a match in this range, the application returns 'UNKNOWN'.

All code for these algorithms can be found in appendix B.

Section V : Results and Conclusions

5.1 Images that Segmented well

For the most part, the application is finding the lesions and is able to process feature vectors successfully. While, often the edge is not found precisely, the original shape is maintained well enough to allow meaningful examination of the lesion. Below are a sample of some of the lesions successfully segmented.

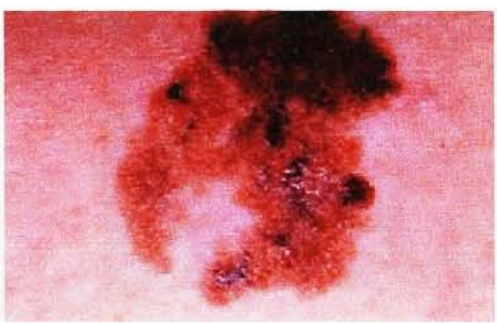


figure 5.1.1 *
melanoma image

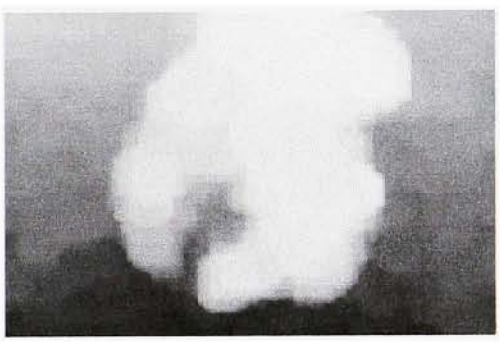


figure 5.1.2
Image during morphological processing,
note that the skin texture has been dilated out.

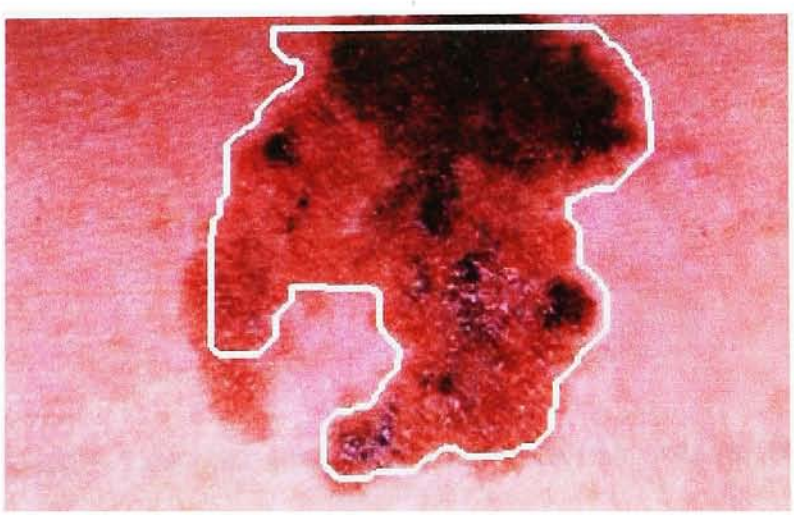


figure 5.1.3

* image from Indiana University: http://www.pathology.iupui.edu/drhood/melanoma.html

Segmented image. Some lighter areas are missed by The algorithm, but the character of the lesion is captured.

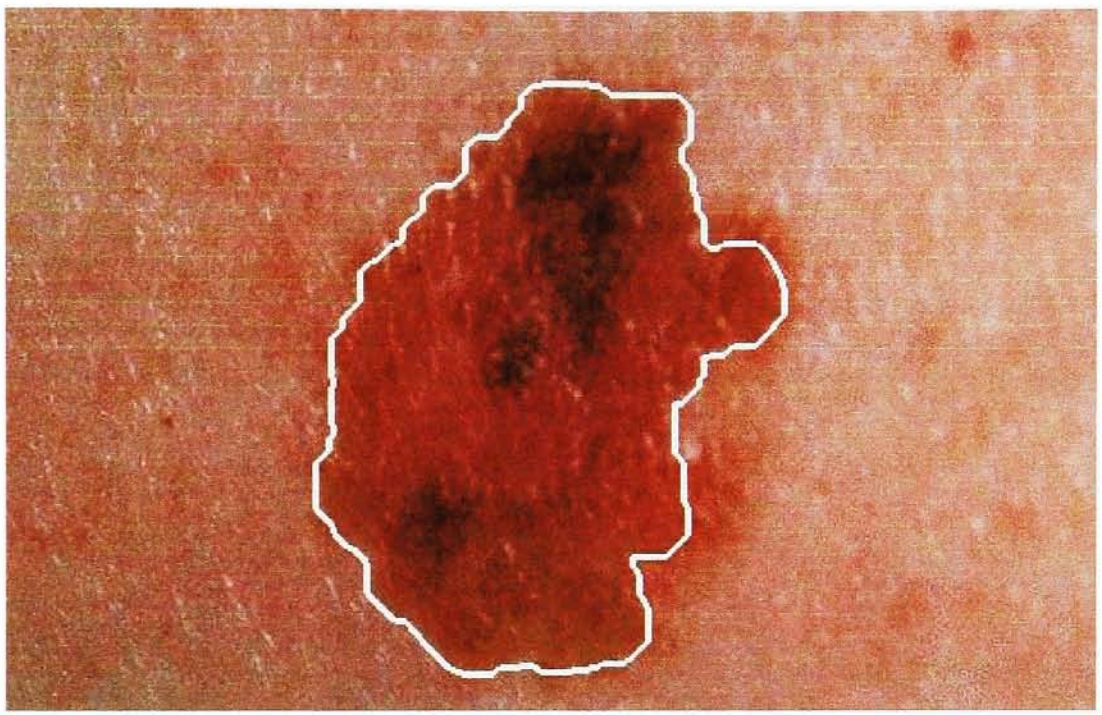


figure 5.1.4*

The interesting thing to me about this edge is that the more raised portion of the lesion has been well picked out, actually giving the lesion more definition than was noticable in the original image.

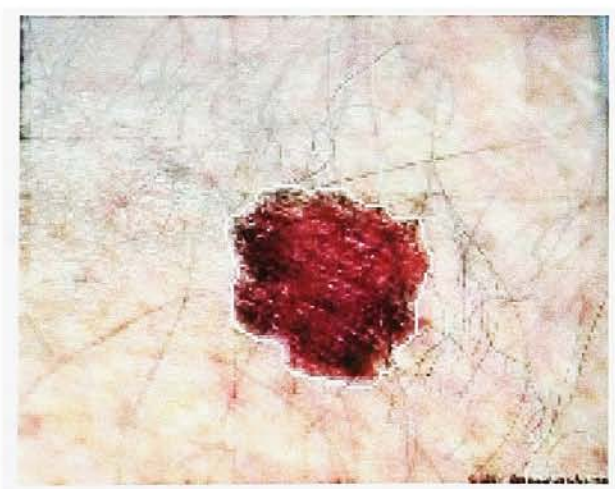


figure 5.1.5**

This image segmented particularly well, despite the numerous hairs in the background, and skin texture.

---

* ** Images from Iowa College of Medicine : http://tray.dermatology.uiowa.edu/DermImag.htm

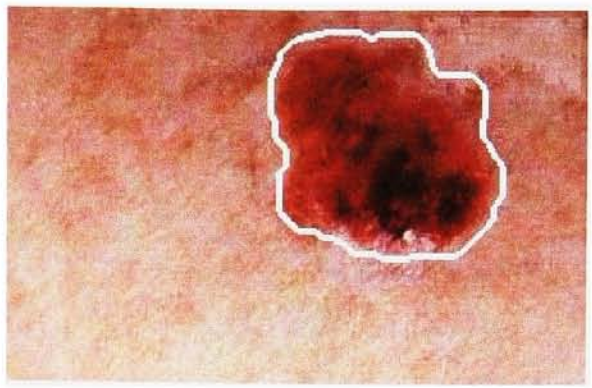


figure 5.1.6*
This lesion was segmented successfully, even thought the image is off center and very close to the border. In this case, the morphological processing artificially expanded the border of the lesion slightly, but again, the general characteristics of the lesion are captured.

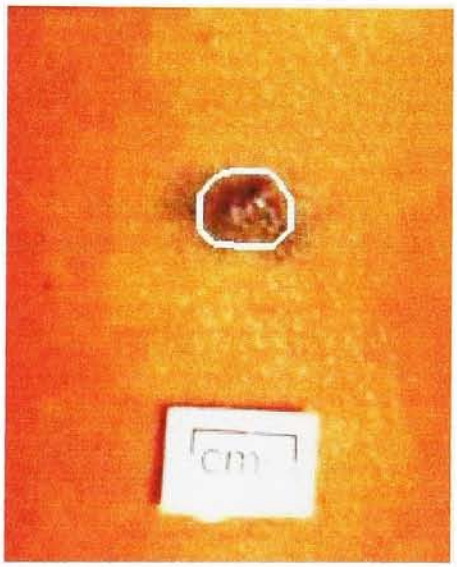


figure 5.1.7 **
This lesion is an example of what is called a blue nevus. It is a benign mole with a characteristic blue coloring. In this image, the edge-trimming software was able to avoid segmenting the ruler at the bottom of the image and locate the lesion.

The next set of images were procured with a Cannon EOS Rebel G camera and scanned on a low-end scanner. The first image is slightly out of focus. There was some processing done on the image to lighten the noticeable freckles, as the application is designed for a central lesion with a fairly uniform background. Interestingly as well, this birthmark classified as "Red", (meaning it has some high values for some of the characteristics of melanoma.) , and does have three of the characteristics: an irregular shape, much larger than a pencil eraser, and noticable texture. In many of the images I've viewed, lesions diagnosed as melanoma did look like freckles, which makes it that much more challenging for dermatologists to accurately diagnose, or even detect in the first place, that a lesion is becoming a problem.

---

* image from Indiana University http://www.pathology.iupui.edu/drhood/melanoma.html
** image from New Zealand Dermatological Society site : http://www.dermnetnz.org/

The application does a good job of segmenting this lesion as seen in figure 5.1.8.

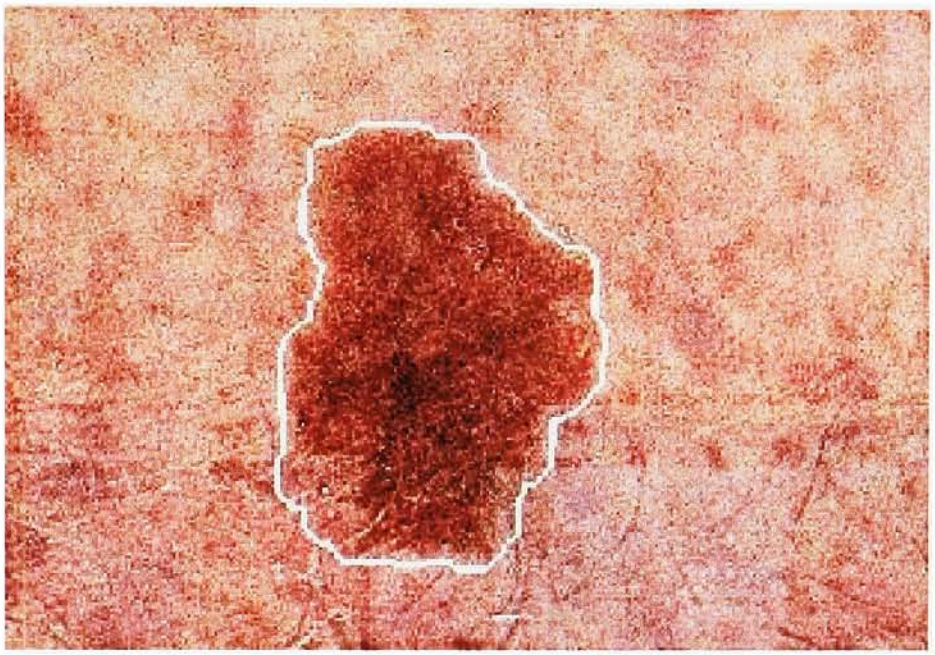


figure 5.1.8
Pigmented Birthmark
The lesion segmented well, except for a small are in the lower left side.

figure 5.1.9
This is a lesion captured with the Cannon with a flash in daylight. Again, the edge is imperfect, but the region of interest has been identified, and the shape appears reasonable.

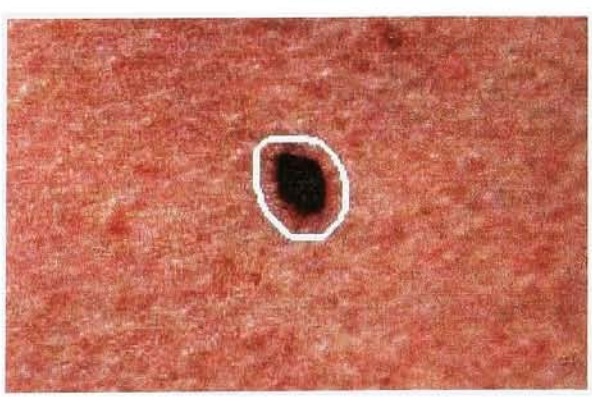


figure 5.1.10

This is a small dysplastic mole. The application located the lesion well, but the irregular border is missed. However, the values captured do mark is as a Warning.

Table 5.1.1 presents the results of the images processed for the database. Each column is subtotaled with the average value. This was used for the image matching in the database.

Table 5.1.2 captures the feature values for the input images, which images they matched in the database. It also indicates how they classified based on different feature value ranges. It is interesting to note that the two suspect images in the benign group did classify as AMBER (requires investigation).
Some of those results follow, the complete match ups can be found in Table 5.1.2.

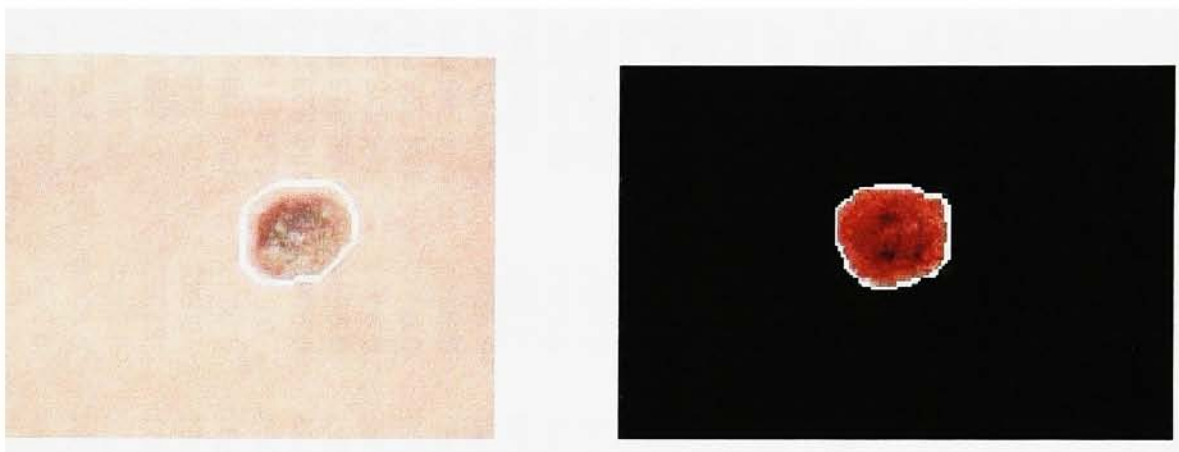
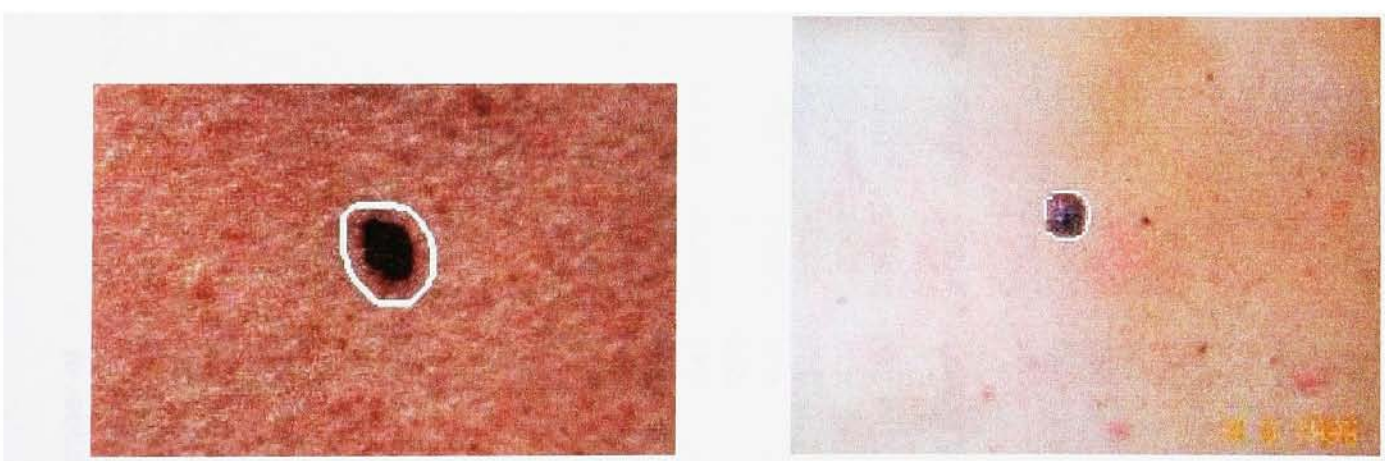


figure 5.1.11

Mole taken with the Cannon **Mole from the database** *

While these images are different in color, they appear to match well on the other feature vectors for shape and texture.



Dysplastic mole taken with Cannon **Congential nevus from database** **

figure 5.1.12

These two images are similar in compactness and are also both very dark.

---

* ** image from the University of Florida: Health Science Center: http://www.health.ufl.edu/molehill/molehill.html

Database Images

Image Name	Diagnosis	Number of Colors	% Red	%Black	%Pink	Compactness	Area Ratio	Texture_Energy	Texture_Entropy	Texture_Distance	Texture Contrast
basalpig	basal cell c	668	0.0128	0.0116	0.0012	13.1084	0.9561	60,205,822	-4.1214	0.5577	1.7663E+11
bcc_007	basal cell c	272	0.7878	0	0	12.554	0.9366	112,237,037	-2.1851	0.2704	1.6541E+12
bcc_12	basal cell c	615	0.0309	0.0038	5.10E-04	13.5766	0.9472	70,324,437	-3.9	0.5578	3.3597E+11
MM-002-low	basal cell c	588	0.006	0	0	20.6877	0.7952	41,728,015	-4.505	0.6647	1.9341E+11
dysnev2	dysplastic	519	0.5827	0.0095	0	13.3191	0.965	85,328,788	-3.5698	0.503	7.7397E+11
dysnev3	dysplastic	310	0.8248	0.0045	0	13.9796	0.9247	131,103,608	-2.2179	0.2279	1.9152E+12
blurrymel	melanoma	165	0.4012	0	0	13.0002	0.949	132,511,281	-1.6475	0.2512	2.0293E+12
dark_mel	melanoma	387	0.0071	0	0	10.7282	0.9864	212,523,270	-0.9736	0.0522	3.4795E+12
mel10	melanoma	363	0.1509	0	0	12.357	0.9561	88,815,860	-2.8875	0.2661	6.2302E+12
mel11	melanoma	394	0.4461	0.0016	0	16.1252	0.9166	115,635,078	-2.3991	0.2643	1.0710E+12
mel12	melanoma	318	0.4049	0.0028	0	16.5269	0.9007	104,786,816	-2.3953	0.2009	9.1765E+12
mel12	melanoma	318	0.4049	0.0028	0	16.5269	0.9007	104,786,816	-2.3953	0.2009	9.1765E+11
mel15	melanoma	540	0.0159	0.0158	0	17.0165	0.8838	68,074,257	-3.8715	0.5361	3.3820E+11
mel16c	melanoma	600	0.0866	0.0275	1.98E-04	16.8206	0.906	44328150	-4.5462	0.636	6.0894E+10
mel2	melanoma	541	0.3978	0.0227	0	22.1078	0.8374	63,323,395	-3.6573	0.4285	1.2722E+11
mel2	melanoma	541	0.3978	0.0227	0	22.1078	0.8374	63,323,395	-3.6573	0.4285	1.2722E+11
mel21	melanoma	275	0	0	0	10.595	0.9845	162,805,816	-1.4924	0.0769	2.6243E+12
mel29	melanoma	464	0.6425	0.0044	1.16E-04	13.1013	0.953	79,961,308	-3.0737	0.2578	5.9509E+11
mel6	melanoma	340	0.257	0	0.00E+00	12.1442	0.9815	98,891,783	-2.4832	0.1791	9.6944E+11
mel8	melanoma	417	0.2948	0.0481	0	15.856	0.9166	83,245,205	-3.0573	0.2953	4.7264E+11
roundmel	melanoma	320	0.5038	0.0107	0	12.7955	0.944	93,734,643	-3.009	0.3968	8.2125E+11
Averages		**426.428571**	**0.31697**	**0.00898**	**9.64E-05**	**15.00164286**	**0.922786**	**96,079,751**	**-2.954543**	**0.345338**	**1.6233E+12**
blue2_nevus	nevus	251	0.0016	0	0	11.4075	0.9885	239,789,874	-0.5615	0.0424	3.9361E+12
cong_nevus	nevus	352	0.4638	0	0	12.0141	0.9555	109549966	-2.4005	0.2546	1.6177E+12
cong2_nevus	nevus	292	0	0	0	11.2981	0.9865	213,996,846	-0.8917	0.0574	3.5008E+12
mole1	nevus	179	0	0	0	11.1154	0.9903	237,901,288	-0.5219	0.032	3.9077E+12
mole3	nevus	327	0.6018	0	0	11.9142	0.9793	162,844,366	-1.5083	0.0964	2.6103E+12
nevus2s	nevus	211	0	0	0	12.3325	0.9653	174,181,678	-1.2578	0.0958	2.8340E+12
nevus3s	nevus	138	0	0	0	12.0876	0.9755	195,494,022	-0.948	0.0577	3.1997E+12
nevus4s	nevus	238	0	0	0	11.8803	0.9763	193,821,584	-1.0846	0.0793	3.1528E+12
nevus7s	nevus	282	0	0	0	11.3696	0.9905	226,844,820	-0.7214	0.0317	3.7264E+12
sebker1s	nevus	320	0	0	0	12.1246	0.9638	180,248,620	-1.3019	0.0733	2.9294E+12
Averages		**259**	**0.10672**	**0**	**0**	**11.75439**	**0.97715**	**193,467,306**	**-1.11976**	**0.08206**	**3.1415E+12**

Automatic Analysis of Skin Lesions

Input Image Feature Values

FWL and Match Results

Image Name	Diagnosis	# Colors	% Red	%Black	%Pink	Compactness	Area Ratio	Texture Energy	Texture_Entropy	Texture_ Distance	Texture Contrast	FWL	Match
jhet1	?	404	0.0179	0	0	10.9987	0.965	151,972,080	-1.7943	0.1151	2.4296E+12	A	mel11
jhet3	?	288	0.0362	0	0	11.9588	0.9884	232,745,954	-0.6751	0.0436	3.8208E+12	G	cong2_nevus
jhet4	?	396	0.0172	0	0	11.6799	0.9743	158,834,648	-1.6824	0.0985	2.5501E+12	A	mel11
jhet4_change	?	406	0	0	0	11.7571	0.9711	122,641,430	-2.1561	0.1762	1.8846E+12	A	mel11
jhet8	?	386	0.1129	0	0	10.6689	0.9875	223,416,610	-0.8395	0.051	3.6627E+12	G	dark_mel
khmm7	?	628	0.1251	0.0015	0	14.3768	0.9359	64,583,421	-4.5517	1.1002	1.5575E+11	R	bcc_12
khmm7_chang	?	633	0.408	0.3193	0	21.2393	0.9721	211,626,596	0.5961	0.0342	3.4779E+12	A	bccpig
pywr	?	330	0	0	0	10.6564	0.9894	218,486,906	-0.8765	0.0578	3.5725E+12	G	mole3
pywr_change	?	356	0.01	0	4.15E-04	11.9695	0.986	209,200,120	-1.0377	0.0773	3.3994E+12	A	cong_nevus
dyspnev4	dysplastic	237	0.4516	0	0	10.9754	0.9802	132,990,977	-1.7501	0.1858	2.1123E+12	G	nevus4s
irreg_nevus	dysplastic	533	0.2122	0.005	0	12.1668	0.9558	48,152,940	-3.9926	0.4705	8.3345E+10	R	mel15
mel17	melanoma	614	0.0015	0.0014	0	15.3659	0.9441	124,149,983	-2.6652	0.2858	2.1544E+12	R	bcc_12
mel1b	melanoma	589	0.0059	0	0	20.658	0.7938	45,929,563	-4.292	0.6133	3.4589E+11	R	MM-002-low
mel22	melanoma	640	0.0346	1.7044	0	12.0294	0.9748	105,668,026	-2.9037	0.2948	1.5173E+12	A	bcc_12
mel25	melanoma	321	0.4771	0	0	13.2903	0.9439	159,246,070	-1.6007	0.1154	2.5473E+12	A	mel12
mel26	melanoma	421	2.64E-04	0	0	11.8634	0.9755	175,089,632	-1.5821	0.1417	2.8270E+12	A	mel8
mel27	melanoma	596	0.0552	0.0111	9.55E-05	20.5068	0.8632	57,170,406	-4.5175	0.7805	4.5703E+11	R	mel16c
mel28	melanoma	355	6.46E-04	0	3.23E-04	12.3423	0.9584	192,529,200	-1.253	0.1047	3.1260E+12	R	cong_nevus
mel3	melanoma	498	0.5748	0.5723	0	16.0729	0.9587	135,786,868	-2.2265	2.7637	1.1077E+12	R	dysnev2
cmpd_nevus	nevus	265	0.0032	0	0	11.3809	0.9894	215,026,416	-0.8934	0.0625	3.5185E+12	G	bcc-007
sebker2s	nevus	326	0	0	0	10.8834	0.9821	92,824,284	-2.8023	0.3355	1.4966E+12	G	mel12
sebker3s	nevus	639	0.4524	0	0.0024	10.9783	0.9801	45,545,635	-4.0823	0.5321	1.5568E+11	A	bcc-_12
wart	nevus	373	0.0033	0	0	11.9477	0.9607	148,232,526	-1.8036	0.1175	2.2319E+12	A	mel10

key:
R = Red
A = Amber
G = Green

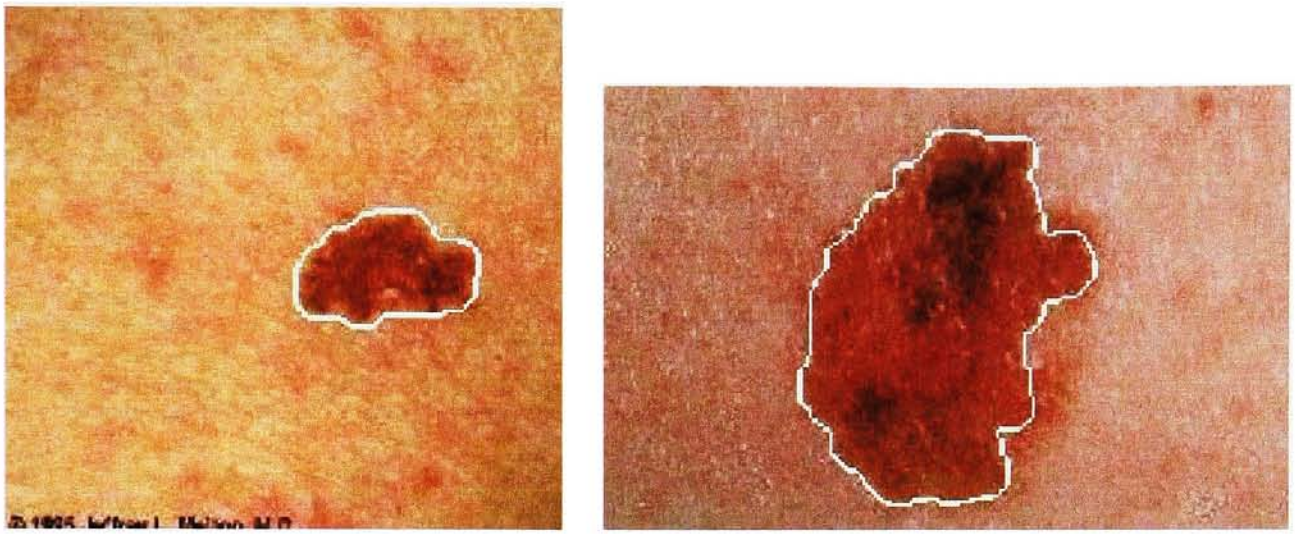


figure 5.1.13

input image (melanoma) * **matching image (melanoma)** **

While these images are positioned differently, they actually are very similar in shape, color and texture.

5.2 Types of Images That Did Not Segment Well

In table 5.2.1, the results of the overall attempts to segment images from both the web and locally obtained are shown. The images below indicate some of the issues encountered in identifying the region of interest. In the first image, its clear that a dark shadow on the lower portion of the image confused the segmentation software.

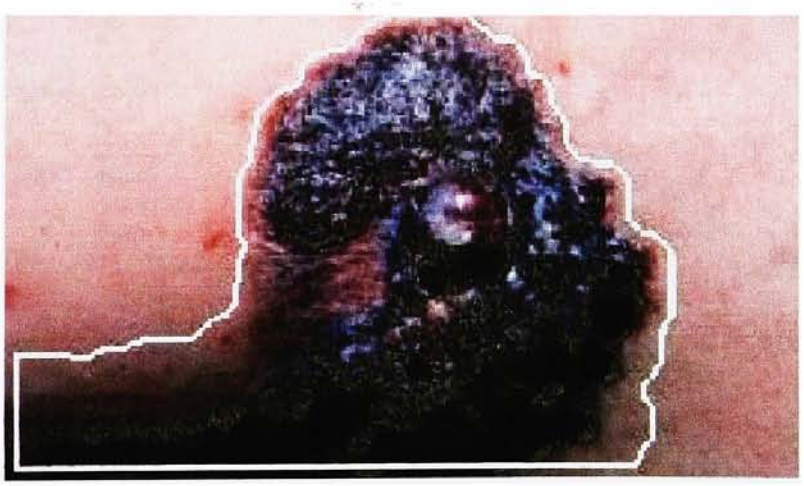


figure 5.2.1 ***
Melanoma, background shadowed in lower part of image

* Images from Loyola University :
http://www.meddean.luc.edu/lumen/MedEd/medicine/dermatology/melton/atlas.htm
** Images from Iowa College of Medicine : http://tray.dermatology.uiowa.edu/DermImag.htm
*** Image from the University of Utah: http://medstat.med.utah.edu/kw/derm/

Image Name	Diagnosis	Quality of Edge Detected	Problem
blue2_nevus.jpg	nevus	OK	
cong2_nevus.jp	nevus	OK	
dark_mel.jpg	melanoma	OK	
dysnev1.jpg	dysplastic	OK	
dysnev2.jpg	dysplastic	OK	
dysnev3.jpg	dysplastic	OK	
dyspnev4.jpg	dysplastic	OK	
irreg_nevus.jpg	dysplastic	OK	
jhet1.jpg	nevus	OK	
jhet3.jpg	nevus	OK	
jhet4.jpg	nevus	OK	
jhet4_change.jp	dysplastic	OK	
khmm7_change	dysplastic	OK	
mel10.jpg	melanoma	OK	
mel11.jpg	melanoma	OK	
mel15.jpg	melanoma	OK	
mel16c.jpg	melanoma	OK	
mel17.jpg	melanoma	OK	
mel1b.jpg	melanoma	OK	
mel21.jpg	melanoma	OK	
mel22.jpg	melanoma	OK	
mel25.jpg	melanoma	OK	
mel26.jpg	melanoma	OK	
mel27.jpg	melanoma	OK	
mel28.jpg	melanoma	OK	
mel29.jpg	melanoma	OK	
mel3.jpg	melanoma	OK	
mel6.jpg	melanoma	OK	
mole1.jpg	nevus	OK	
mole3.jpg	nevus	OK	
nevus2s.jpg	nevus	OK	
nevus3s.jpg	nevus	OK	
nevus4.jpg	dysplastic	OK	
nevus7s.jpg	nevus	OK	
pywr_change.jp	dysplastic	OK	
pywr2.jpg	nevus	OK	
sebker1s.jpg	sebherractic keratosis	OK	
sebker2s.jpg	sebherractic keratosis	OK	
sebker3s.jpg	sebherractic keratosis	OK	
wart.jpg	verruca	OK	
jhet8.jpg	nevus	OK	
bcc_007.jpg	basal cell carcinoma	good	
blurrymel.jpg	melanoma	good	
cong_nevus	nevus	good	
khmm7.jpg	pigmented birthmark	good	
mel12.jpg	melanoma	good	
mel14.jpg	melanoma	good	
basalpig	basal cell carcinoma	Very Good	
bcc_12.jpg	basal cell carcinoma	Very Good	
cmpd_nevus	nevus	Very Good	
mel2.jpg	melanoma	Very Good	
mel7.jpg	melanoma	Very Good	
MM-002-low	basal cell carcinoma	Very Good	
roundmel.jpg	melanoma	Very Good	
bcc_17.jpg	basal cell carcinoma	poor	white glare in lesion segments out
bcclow/jpg	basal cell carcinoma	poor	required a different structuring element
germbcc.jpg	basal cell carcinoma	poor	very red background, orange lesion
img0089.jpg	melanoma	poor	very faint lesion - light orange
mel1.jpg	melanoma	poor	shadowed background
mel13.jpg	melanoma	poor	hole fill routine removed lesion
mel8.jpg	melanoma	poor	shadowed background
mole2.jpg	nevus	poor	thick dark hair in background
sqCCa001.jpg	squamous cell carcino	poor	shadowed background
uneven_mel.jpg	melanoma	poor	white glare in lesion segments out

In the next image, it can be seen that there is a great deal of white glare in the image.

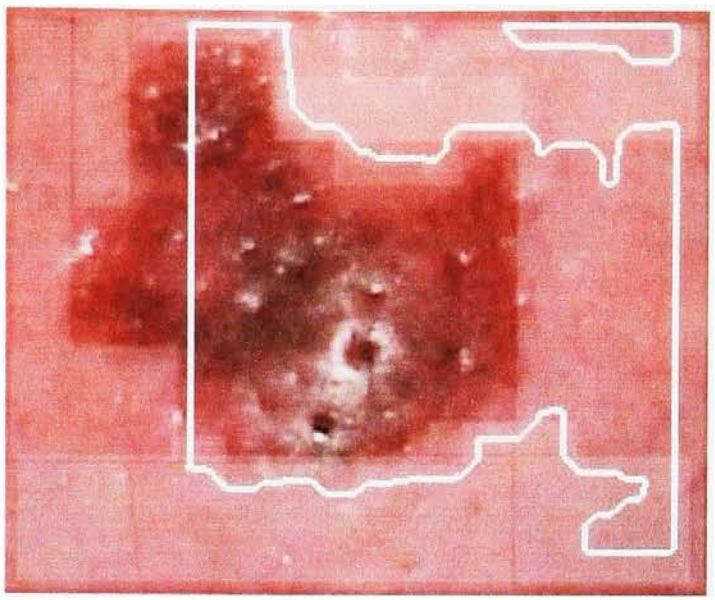


figure 5.2.2*
The glare in the center of this image and the speckles of white light in the background caused the segmentation to fail

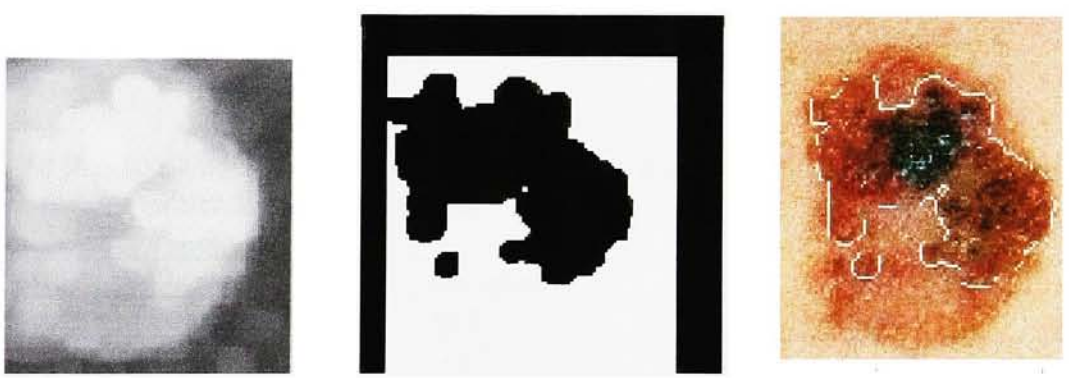


figure 5.2.3
This image represents the third problem that caused the segmentation to fail, because there areas of lighter shade within the lesion, the morphological processes failed to completely fill the a hole inside the lesion, and also to join another region to the whole. Its worth noting, however, that the application would note this as a region to investigate further, which is valuable, as this is a melanoma image.

* Images from Iowa College of Medicine : http://tray.dermatology.uiowa.edu/DermImag.htm

5.3 Tracking History

In order to determine whether the Application would truly be useful in tracking changes in a lesion, several of the input images were modified and fed back through the application to determine what changes would be recorded, and if this would affect classification. The results are below:

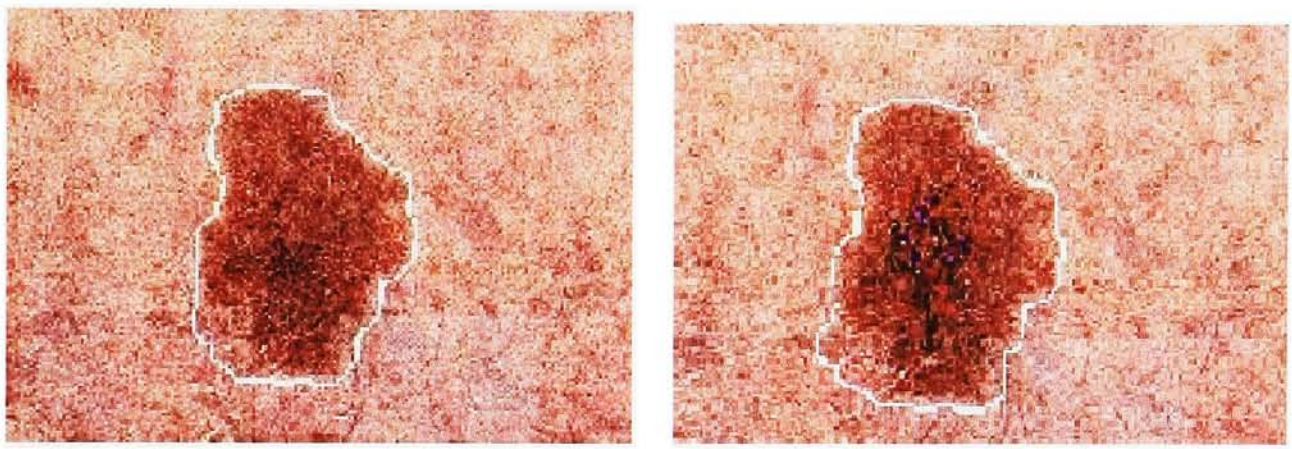


figure 5.3.1

Original **Modified**

Note that some additional colors were applied to the modified version. The table below indicates the feature values for the two images.

Feature	Original	Modified
Number of colors	628	633
Percent Red	.1251	.408
Percent Black	.0015	.3193
Percent Pink	0	0
Compactness	14.3768	21.2393
Area Ratio	.9359	.9721
Texture Energy	64583421	211626596
Texture Entropy	-4.5517	-.5961
Texture Distance	1.1002	.0342
Texture Contrast	1.5575e+11	3.4779e+12

As can be seen, there are significant value changes that can be captured, even though viewing the image the changes are subtle and could be missed.

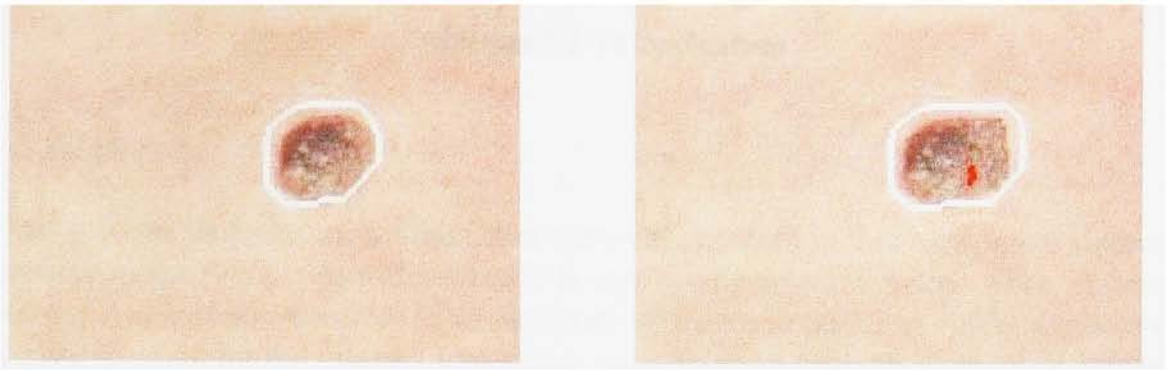


figure 5.3.2

Original Modified

Feature	Original	Modified
Number of colors	330	356
Percent Red	0	.01
Percent Black	0	0
Percent Pink	0	04.15e-04
Compactness	10.6564	11.9695
Area Ratio	.9894	.986
Texture Energy	218486906	209200120
Texture Entropy	-.8765	-1.0377
Texture Distance	.0578	.0773
Texture Contrast	3.5725e+12	3.3994e+12

Again, this table indicates measurable changes that could be otherwise missed. In addition, the classifier rated the modified image as Amber (warning). The original image was rated Green (benign).

Section VI : Conclusions

6.1 Summary

Much of the segmentation issues that were encountered in this work have already been overcome in the field. Algorithms could be easily employed to reduce glare, lighten shadows, and increase contrast in an automatic fashion. In addition, many applications use color to segment the images, and that is more tolerant towards changes in intensity. As table 6.1.1 demonstrates, approximately 84% of the images segmented well enough to allow data capture. That percentage could easily be improved with more robust background handlers. In addition, the failures usually still segemented out part of the lesion in question and would be able to at least draw attention to the region of interest.

It devolved that number of colors, compactness, the convex hull / edge area ratio, texture energy, and texture_distance turned out to be most obvious indicators of a lesions characteristics. The red, black and pink metrics did not turn out to be helpful, in fact where those numbers did register, particularly red, usually turned up in the benign lesions. These capture techniques could be refined to more accurately capture this color information.

In the database matching routine, where an input image is matched against the database, it became apparent that the choice of database images was important to get meaningful matchings. If the database contains many melanomas that look a lot like a normal mole, the results returned may not be especially helpful. Using both a voting scheme and a scheme that added values together (which turned out to be governed by match on number of colors) the number of colors match provided better results. Clearly, this requires better turning of both database image selection, and matching criteria. One obvious improvement would be to weight the feature values, so that a match on the percent red, for example, carried less value than a match number of colors, which was a better indicator.

I think there is merit here, for an unsupervised system to assist dermatologists, particularly those who are using some kind of image capture now. Some tweaking to the system sped up the processing, so that for most images, it only takes about 60 - 180 seconds. Images taken by a professional will likely be of a reasonable quality, and most likely intended for image tracking, and probably taken from a similar aspect. At a minimum, the automatic classification could be an assist for the dermatologist, seeing many patients a day, especially in sunny climates.

Section VII : Future work

I think there are many opportunities to grow this work. The next logical step would be to work on improving the background algorithms to address the three issues discussed. This would probably boost the segmentation success into the 90 percent level.

Its also possible that the application could be enhanced to be able to process more than one region in an image. This would support processing of more types of skin ailments. The system could also have a polygon checker to determine if the edge is complete to test the success of the segmentation. Several flavors of morphological structuring elements and iteration of the close and open operations could be devised to allow mulitiple processing to improve segmentation results.

It also occurred to me that one of the most sensitive issues for a patient and dermatologist is someone who has a dysplastic nevus. Very often, in the images I encountered these turned out to be rectangular in shape, and usually had at least one corner....so a corner finding algorithm on the edge would probably greatly improve the classification of this type of lesion.

I think it would also be beneficial to use a real database...the input/output functions provided by Matlab are workable, but not friendly...the initial processing could be performed in Matlab, and the files exported to another language for loading into a database. This would also make it easier for the user to search and retrieve independantly.

Its also possible that this work could have application for other images...for example analysis of wounds to access both healing and treatment. An important component to determine wound status is the amount of necrotic tissue, which color analysis could track. In addition, image history analysis might help determine how healing is progressing.

When I began this work, I was thinking that possibly this application could be web enabled, allowing users of the world wide web to load images up for analysis. However, when I hinted at this sort of use Dr. Papier mentioned that there might be onerous liability insurance, since the great risk is that if a cancer did get missed, whoever supported the website could get sued. I think he's right, and it's a sad commentary on our society, but perhaps if this tool is at the disposal of professionals it might make early detection a little easier.

However, far and away, I think the most important contribution of this application is that it can be a tool for tracking mole change over time. This appears to be an area where the software is not available, and where it could provide an critical link in assisting physicians in discovering a cancer, and hopefully in an early stage.

End Notes

1.http://www.cancer.org/docroot/CRI/content/CRI_2_4_1X_What_are_the_key_statistics_for_melanoma_50.asp?sitearea=&level=

2. http://unisci.com/stories/20013/0816013.htm "How Precancerous Moles Progress to Deadly Melanoma", UniSci

3. http://www.cs.bham.ac.uk/~exc/Research/Papers/miua2001_marc.pdf " SIAscopy assists in the Diagnosis of Melanoma by Untilizing Computer Vision Techniques to Visualize the Internal Structure of the Skin" , Moncrieff, Marc Addenbrooke Hospital, Cambridge UK

4. http://www.cancer.org/docroot/MED/content/MED_2_1X_Moles_On_the_Skin_Can_Foretell_a_Person_s_Risk_of_Melanoma.asp

5. http://www.digitalderm.com/Yale_Lecture.pdf "Early Melanoma Detection", Grichnik, James M., Yale University/ Fujisawa Lectureship Series in Dermatology.

6. http://www.cancer.org/docroot/NWS/content/NWS_1_1x_Melanoma_Can_Return_Many_Years_After_Treatment.asp

7. Lee, Tim Kam. "Measuring Border Irregularity and Shape of Cutaneous Melanocytic Lesions", The University of British Columbia, Doctoral Thesis, 2001, pages 8-9.

8. http://unisci.com/stories/20022/0515024.htm Daily University Science News. May 15, 2002.

9. Lee, Tim Kam. "Measuring Border Irregularity and Shape of Cutaneous Melanocytic Lesions", The University of British Columbia, Doctoral Thesis, 2001, page 9.

10. Moncrieff, Marc, Cotton, Symon, Hall, Per, Schiffner, R., Lepski, U. Claridge, Ela. " SIAscopy Assists in the Diagnosis of Melanoma by Utilizing Computer Vision Techniques to Visualize the Internal Structure of the Skin", Addenbrooke Hospital, Cambridge, UK, Astron Clinica, UK, University of Birmingham, UK, Department of Dermatology, University of Regensburg, Germany. htttp://216.239.51.100/search?q=cache:z57DEj4yFJsC:www.cs.bham.ac.uk/~exc/Research/Papers/miua2001_marc.pdf+SIAscopy+assists+in+the+diagnosis+of+melanoma&hl=en&ie=UTF-8

1. Barber, C.Bradford, David P. Dobkin, Hannu Huhdanpaa. "The Quickhull Algorithm for ConvexHulls", ACM Press NY, NY, pages 469 – 483, December, 1996.

12. Swain, Michael, J., Dana H. Ballard. "Color Indexing". The International Journal of Computer Vision. Kluwer Academic Publishers Hingham, MA, USA, page 12, November, 1991.

13. Swain, Michael, J., Dana H. Ballard. "Color Indexing". The International Journal of Computer Vision. Kluwer Academic Publishers Hingham, MA, USA, page 12, November, 1991.

14. Sonka, Milan, Vaclav Hlaval, Roger Boyle. Image Processing, Analysis, and Machine Vision. PWS Publishing, 511 Forest Lodge Road, Pacific Grove, CA 93950, page 646, 1999.

15. Sonka, Milan, Vaclav Hlaval, Roger Boyle. Image Processing, Analysis, and Machine Vision. PWS Publishing, 511 Forest Lodge Road, Pacific Grove, CA 93950, page 649, 1999.

Bibliography

1. Argenziano, Giuseppe MD, Gabriella Fabbrocini, MD, Paolo Carli, MD, Vincenzo De Giorgi MD, Elena Sammarco, MD, Mario Delfino, MD, "Epiluminescence Microscopy for the Diagnosis of Doubtful Melanocytic Skin Lesions", Arch Dermatol, 1998;134:1563-1570.
2. Bach, J.R., S. Paul, R. Jain, "A Visual Information Management System for the Interactive Retrieval of Faces", IEEE transactions on Knowledge and Data Engineering, vol 5, 619 – 628, August 1993.

3. Barber, C. Bradford, David P. Dobkin, Hannu Huhdanpaa, " A Quickhull Algorithm for Convex Hulls", The ACM Digital Library, December 1996.

4. Canosa, Roxanne. "Analysis of Textured Regions Based on Gray Scale Co-occurrence Matrices", Computer Vision project, Rochester Institute of Technology, May 2000.

5. Casper, Jennifer, Tammy Williams, Texture Analysis and Tissue Segementation of Cryosection Images, Technical Report, Visible Human Project, Department of Computer Science, University of Wisconsin-La Crosse, La Crosse, WI 54601.

6. Cohen, Scott, " Finding Color and Shape Patterns in Images", PhD Thesis, Stanford University, Computer Science Department, Stanford University, Standford CA 94305, May 1999.

7. Chahir, Youssef, Liming Chen, "Searching Images on the Basis of Color homogeneous Objects and Their Spatial Relationship", Journal of Visual Communication and Image Representation 11, 302-326, 2000.

8. Deng, Yining, B. S. Manjunath, " Unsupervised Segmentation of Color-Texture Regions in Images and Video", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol 23, no 8, August 2001.

9. Ganster, H., M. Gelautz, A. Pinz, M.Binder, H.Pehamberger, M.Bammer, J.Krocza, " Initial Results of Automated Melanoma Recognition", Austrian Research Center Seibersdorf, A-2444 Seibersdorf, Austria.

10. Gevers, Theo, Sennay Ghebreab, Arnold W.M. Smeulders, "Color Invarient Snakes", ISIS, University of Amsterdam, Kruislaan 403 1098 SJ Amsterdam, The Netherlands. (gevers@wins.uva.nl)

11. Gevers, Theo, C.A. Groen, "Segmentation of Color Images", In proceedings of 7th Scandinavian Conference on Image Analysis, 1991.

12. Grichnik, James M. M.D., "Early Melanoma Detection", Yale University/ Fujisawa Lectureship Series in Dermatology.

13. Hanselman, Duane, Bruce Littlefield, Mastering Matlab 5, A Comprehensive Tutorial and Reference, Prentice Hall, Upper Saddle River, New Jersey 07458, 1998.

14. Haralick, Robert, M. "Statistical and Structural Approaches to Texture", Proceedings of the IEEE, v 67, n 5, pp 786-804.

15. Kliot Michael, Ehud Rivlin, "Invarient-based Shape Retrieval in Pictorial Databases". Computer Vision and Image Understanding, Vol 71, no 2 August, pp 182-197, 1998.

16. Lee, Tim K., M. Stella Atkins, " A New Approach to Measure Border Irregularity for Melanocytic Lesions". This paper can be found on Dr. Atkins site at http://www.cs.sfu.ca/people/Faculty/Atkins/papers/tim.spie00.doc

17. Lee, Tim Kam, "Measuring Border Irregularity and Shape of Cutaneous Melanocytic Lesions", Doctoral Dissertation, Simon Fraser University, January 2001.

18. Moncrieff, Marc, Symon Cotton, Per Hall, R. Schiffner, U. Lepski, Ela Claridge, " SIAScopy Assists in the Diagnosis of Melanoma by Utilizing Computer Vision Techniques to Visualise the Internal Structure of the Skin:, University of Birmingham, UK, Department of Dermatology, University of Regensburg, Germany.

19. Partio, Mari, Bogdan Crameriuc, Moncef Gabbouj, Ari Visa, "Rock Texture Retrieval using Gray Level Co-Occurrence Matrix", 5th Nordic Signal Processing Symposium, Norway, October 2002.

20. Patel, Wima L., Jose F. Arocha, Melissa Diermeier, Robert A. Greenes, Edward H. Shortliffe, " Methods of Cognitive Analysis to Support the Design and Evaluation of Biomedical Systems: The Case of Clinical Practice Guidelines", Academic Press, March 13, 2001.

21. Pauwels, E.J., G. Frederix, "Clustering Strategies for Perceptual Grouping, ESAT-PSI, Dept of Electrical Engineering, K.U. Leuven K. Mercierlaan 94, B-3001 Leuven, Belgium, June 19, 1998.
22. Roning, Juha, Jukka Kontinen, "Measurement of the Area of Involvement in Skin Disease", Proceedings of Intelligent Robots and Computer Vision XV: Algorithms, Techniques, Active Vision, Material Handling (SPIE vol 2904), 18-22 November 1996 Boston, Massachusetts, pp 382-388.
23. Saber, Eli, A. Murat Tekalp, "Integration of Color, Edge, Shape and Texture Features for Automatic Region-Based Image Annotation and Retrieval", Journal of Electronic Imaging, Vol 7, no 3, July 1998.
24. Seul, Michael, Lawrence O'Gorman, Michael J. Sammon, Practical Algorithms for Image Analysis, Cambridge University Press, 40 West 20th Street, New York, NY 10011-4211, 2000.
25. Tao, Bo, Bradley W. Dickinson, "Texture Recognition and Image Retrieval Using Gradient Indexing", Journal of Visual Communication and Image Representation 11, 327-342, 2000.
26. Tong, Arther K.F, MD, Thomas B. Fitzpatrick MD, "Neoplasms of the Skin", American Cancer Society downloads: www.cancer.org/downloads/PUB/ cancer_medicione_e.5_toc.pdf
27. Umbaugh, Scott, Computer Vision and Image Processing, Prentice Hall, Inc Upper Saddle River, NY 07458, 1998.
28. Vicario, Enrico, Image Description and Retrieval, Plenum Press, 233 Spring Street, New York, NY 10013, 1998.
29. Wesolkowski, Slawo, Ed Jernigan, " Color Edge Detection in RGB Using Jointly Eclucidan Distance and Vector Angle", Vision Interface '99, Trois-Rivieres, Canada, 19-21 May.
30. Williams, Paul Stephan, Mike D. Adler, "Generic Texture Analysis Applied to Newspaper Segmentation", The University of Western Australia, Nedlands W.A. 6009, Australia.

Appendix A

The pages that follow contain the images from the sources indicated in Section II.

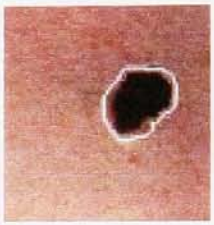
- The first page contains images collected by me.
- The next page is from **State University of California at Davis** http://matrix.ucdavis.edu/tumors.html .
- Next is **University of Florida :** The Molehill part of the Health Science Center. Images courtesy of Dr. Frank Flowers, MD. http://www.health.ufl.edu/molehill/molehill.html
- Following that is **The University of Iowa** http://tray.dermatology.uiowa.edu/DermImag.htm has an absolutely impressive image database. Approximately seventy percent of images were culled from this source.
- The next page is **Loyola University Medical Education Network** which was created by Jeffery L. Melton MD and Jason R. Swanson http://www.meddean.luc.edu/lumen/MedEd/medicine/dermatology/melton/atlas.htm
- Second to last is **Homepage of New Zealand Dermatological Society** http://www.dermnetnz.org/
- Finally, the last set of image are from **The University of Utah** with images by John L. Bezzant http://medstat.med.utah.edu/kw/derm/

In addition, the folder name printed at the top of the form gives some indication of the source of the data.

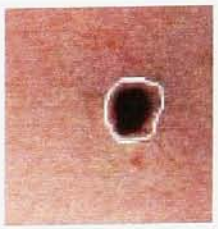
khmm42_out.jpg

jhet1_out.jpg

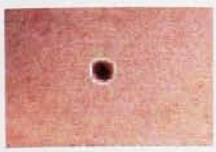
jhet3_out.jpg



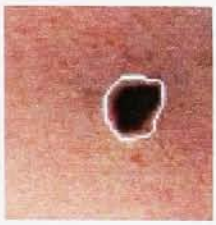
jhet4_change_out.jpg



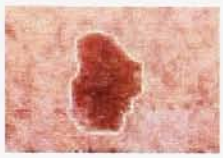
jhet4_out.jpg



jhet8_out.jpg



jhet_change_out.jpg



khmm35_out.jpg

khmm35_out_small...

khmm35_out_small...

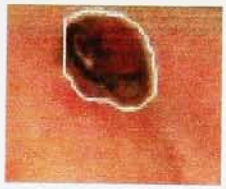
khmm7_change_ou...

khmm7_change_ou...

pywr_change_out.jpg

pywr_out.jpg

mel11_out.jpg

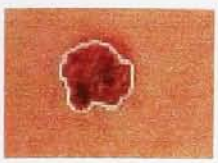


mel6_out.jpg

nev7_out.jpg

nevus2s_out.jpg

nevus3s_out.jpg



round_out.jpg

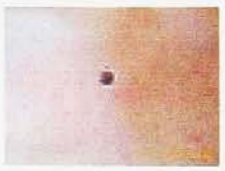
sebker1s_out.jpg

sebker2s_out.jpg

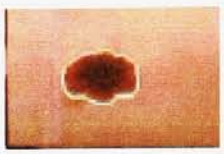
sebker3s_out.jpg

cmpd_nevus_out.jpg

cong2_nevus_out.jpg



cong2_nevus_out_...



cong_nevus_out.jpg

dysnev1_out.jpg

dysnev2_out.jpg

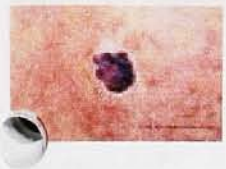
dysnev3_out.jpg

dyspnev4_out.jpg

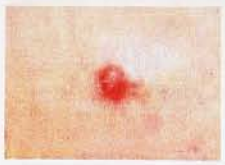
mole1_out.jpg

mole3out.jpg

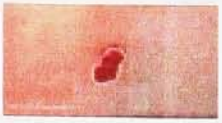
mole3out_smaller.jpg



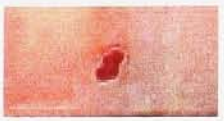
basalpig_out.jpg



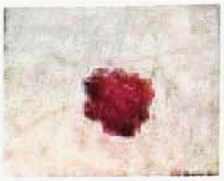
bcc017_bad_out.jpg



bcc_007_out.jpg

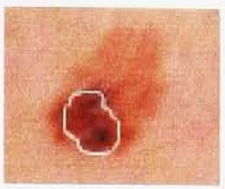


bcc_007_out_small...



bcc_12_out.jpg

bcc_12_out_smalle...



blurry_out.jpg

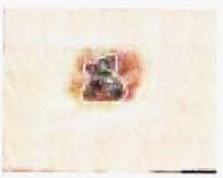
dark_mel_out.jpg

irreg_nevus_out.jpg

mel10_out.jpg

mel12_out.jpg

mel12_out_smaller....

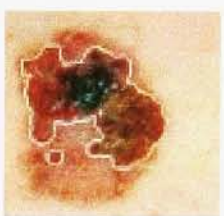


mel15_out.jpg

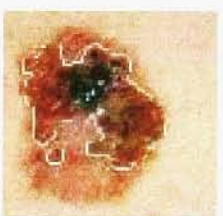
mel16_out.jpg

mel17_out.jpg

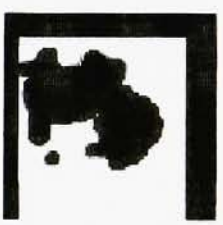
mel1b_out.jpg



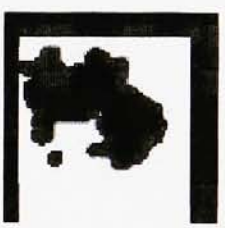
mel7_bad_out.jpg



mel7_bad_out_sma...



mel7_morph_fail1.jpg

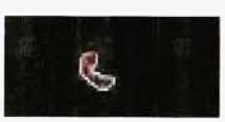


mel7_morph_fail1_...

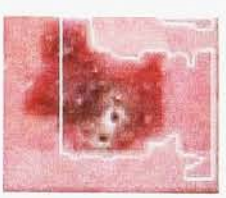
nel7_morph_fail2.jpg

mel7_morph_fail2_...

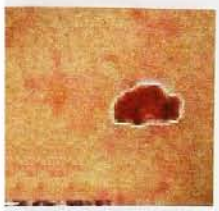
mel8_out.jpg



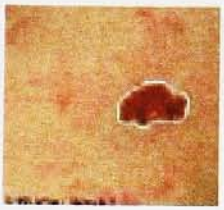
MM-002-out.jpg



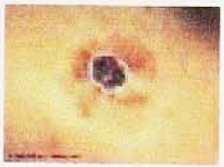
uneve_out.jpg



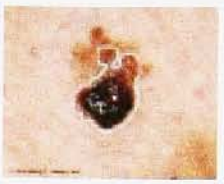
mel25_out.jpg



mel25_out_smaller....



mel26_out.jpg

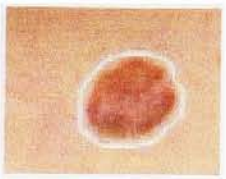


mel27_out.jpg

mel28_out.jpg

mel29_out.jpg

blue2_nevus_out.jpg

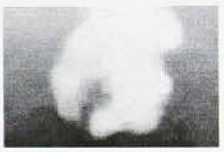


mel21_out.jpg

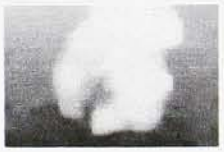
mel22_out.jpg

mel1_bad_out.jpg

mel2.jpg

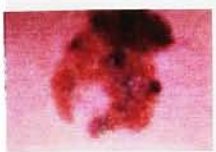


mel2_morph.jpg



mel2_morph_small...

mel2_out.jpg



mel2_smaller.jpg

mel3_out.jpg

mel3_out_small.jpg

Appendix B

```
%*****************************************************************
%*  Run the Processors to Analyze Skin Anomolies
%*   Paula Yandow-Reilly
%    June 28, 2002
%
%*****************************************************************
RW = input('\n Run Classify = R, Database = W/n');
%
%                          EDGE DETECTION
%
%  this process performs the morphological processes
%
test221;
%
%  this one removes artifacts from the image edges
%
fixedges2;
%
%  this routine reverses the black and white pixels
%
rvrs2;
%
% removes edge artifacts as a result of the prev operations
%
fixedges2a;
%
% perform another reverse
%
rvrs3;
%
% morph operation to region grow
%
reg_grow2;
%
% edge detection step
%
Yfix21 = edge(Yfix19,'sobel');
%
%                          SHAPE analysis
% count the edge pixels for (SHAPE) compactness computation
%
img_compactness = edgecount(Yfix21,Yfix19);
%
% using the processed image and the pixel count, remove background
% from original image
%
Yfix35 = markedge(Yfix21,B);
imwrite(Yfix35,'Afix.jpg','jpg');
Yfix42 = rembkgrd(Yfix35);
```

```
figure,imshow(Yfix35);
img_area_ratio = edgeArray(Yfix21);
%
%            COLOR  and Texture Matrix Preparation
%
[img_pct_red,img_pct_black,img_pct_pink,img_no_colors,t] = color_text_prep(Yfix21,Yfix42);
%
%                          TEXTURE analysis
%
[E,EN,D,C] = texture(t);
img_texture_energy   = E;
img_texture_entropy  = EN;
img_texture_distance = D;
img_texture_contrast = C;
%
img_name = file
img_diagnosis = 'melanoma
img_no_colors
img_pct_red
img_pct_black
img_pct_pink
img_compactness
img_area_ratio
img_texture_energy
img_texture_entropy
img_texture_distance
img_texture_contrast
figure,subplot(2,1,1),imshow(B),title 'original ;
subplot(2,1,2),imshow(Yfix42),title 'after processing';
%
if RW == 'R'
  classify;
  callDBRead;
else
  if RW = 'W'
   callDBWrite;
end;
%
%
```

```
%%*****************************************************************
%*   Region detection = using morph operations                    *
%*   Paula Yandow-Reilly                                          *
%    August 2001                                                  *
%    The purpose of this assignment is to investigate             *
%     threshholding and segmentation                              *
%                                                                  *
%*****************************************************************
%set up some variables for file reading....
%
basalcc = 'bcclow.jpg';   %test1  basal cell carcinoma
basalpig = 'bccpig.jpg';  %test2  pigmented basal cell carc.
bcc007 = 'bcc_007.jpg';   % red nodule
bcc012 = 'bcc_12.jpg';    % irreg flat maroon dark hairs
bcc017 = 'bcc_17.jpg';    % irreg flat maroon dark hairs
mel1b = 'MM-002-low.jpg'; %regressed image
germbcc = 'germbcc.jpg';  %very red image, orangish lesion
img0089 = 'img0089.jpg';  % faint lesion, light orange color
khmm = 'khmm.jpg';        % Kevin freckle
khmm7 = 'khmm7.jpg';      % Kevin freckle - massaged
jhet1 = 'jhet1.jpg';      % John mole 1
jhet2 = 'jhet2.jpg';      % John mole 2
jhet3 = 'jhet3.jpg';      % John mole 3
jhet3 = 'jhet3a.jpg';     % John mole 3  cropped
jhet4 = 'jhet4.jpg';      % John mole 4 - cropped
jhet8 = 'jhet8.jpg';      % John mole normal looking
round = 'roundmel.jpg';   % round melanoma
blurry = 'blurrymel.jpg'; % blurry faint image
uneven = 'unevenmel2.jpg'; % very irregular border
mole1 = 'mole1.jpg';      %
mole2 = 'mole2.jpg';      %
mole3 = 'mole3.jpg';      %
mel1 = 'mel1.jpg';        % blurry melanoma
mel2 = 'mel2.jpg';        % partially occluded, very red image
mel3 = 'mel3.jpg';        % partially occluded, very black image
mel4 = 'SqCCa001.jpg';    % proturbuerant lesion-light red,black hairs
mel6 = 'mel6.jpg';        % raised bump, red and black
mel7 = 'mel7a.jpg';       % freckle-looking
mel8 = 'mel8.jpg';        % mole looking
mel10 = 'mel10.jpg';      %
mel11 = 'mel11.jpg';      %
mel12 = 'mel12.jpg';      %
mel13 = 'mel13.jpg';      %
mel14 = 'mel14.jpg';      %
mel15 = 'mel15.jpg';      %
mel16 = 'mel16c.jpg';     % dark cloverleaf
mel17 = 'mel17.jpg';      % dark teardrop shape - massaged bkgrd
drkmel = 'dark_mel.jpg';  % dark small - massaged bkgrd
pywr = 'pywr2.jpg';       % round lesion..trunk
```

test221.m

```
wart = 'wart2.jpg';       %
irnev = 'irreg_nevus.jpg'; % irregular mole
congnev = 'cong_nevus.jpg'; % congenital mole
cmpdnev = 'cmpd_nevus.jpg'; % compound   mole
dysnev1 = 'dysnev1.jpg';
dysnev2 = 'dysnev2.jpg';
dysnev3 = 'dysnev3.jpg';
%
%  read the basal image into matrix B
%
file = dysnev1;
[B,map] = imread(file,'jpg');
thresh = .005;
B2 = rgb2gray(B);          %%  DEBUG  5/6/2002
Yfix = imcomplement(B2);
%figure,subplot(3,3,1),imshow(Yfix),title 'lplac - no imfilter'

%
%  create structuring element, perform close operation
%
 SE = strel('disk',7);
 Yfix2 = imclose(Yfix,SE);
% about to comment out image show 11/23
%subplot(3,3,2),imshow(Yfix2),title 'disk sd, close operation  ;
%
%  do it again, perform close operation
%
 Yfix2 = imclose(Yfix2,SE);
%subplot(3,3,3),imshow(Yfix2),title 'disk sd, close again ';
%
% now try maxing  it...
%
 Yfix3 = imopen(Yfix2,SE);
%subplot(3,3,4),imshow(Yfix3),title 'disk sd, open operation  ;
%
% do it again...
%
 for i = 1:10
  Yfix4 = imdilate(Yfix3,SE);
 end;
%
%subplot(3,3,5),imshow(Yfix4),title 'after dilate';
%

 SE = strel('disk',3);
 for i = 1:3
   Yfix5 = imopen( Yfix4,SE);
 end;
%subplot(3,3,6),imshow(Yfix5),title 'disk  se, after 3  opens  ;
```

```
%-----------------------------------------------------------------
%  try filling in the cell
%
%BWdfill = imfill(Xfix3,'holes );
%subplot(3,3,6), imshow(BWdfill),title('binary image with filled holes');
%BWnobord = imclearborder(BWdfill, 4);
%subplot(3,3,7),imshow(BWnobord),title('cleared border image');
%
% comment out for now....
%
%seD = strel('diamond',1);
%BWfinal = imerode(BWnobord,seD);
%BWfinal = imerode(BWfinal,seD);
%subplot(3,3,7),imshow(BWfinal),title('segmented image');
%BWoutline = bwperim(BWfinal);
%subplot(3,3,8),imshow(BWoutline),title('outline of cell');
%Segout = imadd(BW1, immultiply(BWoutline, 255));
%figure,imshow(Segout),title('outlined original image');
```

```
%%*****************************************************************
%*   Region detection = using morph operations                    *
%*   Paula Yandow-Reilly                                          *
%    August 2001                                                  *
%    The purpose of this assignment is to investigate             *
%     threshholding and segmentation                              *
%                                                                  *
%*****************************************************************
%set up some variables for file reading....
%
basalcc = 'bcclow.jpg';   %test1  basal cell carcinoma
basalpig = 'bccpig.jpg';  %test2  pigmented basal cell carc.
bcc007 = 'bcc_007.jpg';   % red nodule
bcc012 = 'bcc_12.jpg';    % irreg flat maroon dark hairs
bcc017 = 'bcc_17.jpg';    % irreg flat maroon dark hairs
mel1b = 'MM-002-low.jpg'; %regressed image
germbcc = 'germbcc.jpg';  %very red image, orangish lesion
img0089 = 'img0089.jpg';  % faint lesion, light orange color
khmm = 'khmm.jpg';        % Kevin freckle
khmm7 = 'khmm7.jpg';      % Kevin freckle - massaged
jhet1 = 'jhet1.jpg';      % John mole 1
jhet2 = 'jhet2.jpg';      % John mole 2
jhet3 = 'jhet3.jpg';      % John mole 3
jhet3 = 'jhet3a.jpg';     % John mole 3  cropped
jhet4 = 'jhet4.jpg';      % John mole 4 - cropped
jhet8 = 'jhet8.jpg';      % John mole normal looking
round = 'roundmel.jpg';   % round melanoma
blurry = 'blurrymel.jpg'; % blurry faint image
uneven = 'unevenmel2.jpg'; % very irregular border
mole1 = 'mole1.jpg';      %
mole2 = 'mole2.jpg';      %
mole3 = 'mole3.jpg';      %
mel1 = 'mel1.jpg';        % blurry melanoma
mel2 = 'mel2.jpg';        % partially occluded, very red image
mel3 = 'mel3.jpg';        % partially occluded, very black image
mel4 = 'SqCCa001.jpg';    % proturbuerant lesion-light red,black hairs
mel6 = 'mel6.jpg';        % raised bump, red and black
mel7 = 'mel7a.jpg';       % freckle-looking
mel8 = 'mel8.jpg';        % mole looking
mel10 = 'mel10.jpg';      %
mel11 = 'mel11.jpg';      %
mel12 = 'mel12.jpg';      %
mel13 = 'mel13.jpg';      %
mel14 = 'mel14.jpg';      %
mel15 = 'mel15.jpg';      %
mel16 = 'mel16c.jpg';     % dark cloverleaf
mel17 = 'mel17.jpg';      % dark teardrop shape - massaged bkgrd
drkmel = 'dark_mel.jpg';  % dark small - massaged bkgrd
pywr = 'pywr2.jpg';       % round lesion..trunk
```

```
wart = 'wart2.jpg';       %
irnev = 'irreg_nevus.jpg'; % irregular mole
congnev = 'cong_nevus.jpg'; % congenital mole
cmpdnev = 'cmpd_nevus.jpg'; % compound   mole
dysnev1 = 'dysnev1.jpg';
dysnev2 = 'dysnev2.jpg';
dysnev3 = 'dysnev3.jpg';
%
%  read the basal image into matrix B
%
file = dysnev1;
[B,map] = imread(file,'jpg');
thresh = .005;
B2 = rgb2gray(B);          %%  DEBUG  5/6/2002
Yfix = imcomplement(B2);
%figure,subplot(3,3,1),imshow(Yfix),title '1plac - no imfilter'

%
%  create structuring element, perform close operation
%
 SE = strel('disk',7);
 Yfix2 = imclose(Yfix,SE);
% about to comment out image show 11/23
%subplot(3,3,2),imshow(Yfix2),title 'disk sd, close operation  ;
%
%  do it again, perform close operation
%
 Yfix2 = imclose(Yfix2,SE);
%subplot(3,3,3),imshow(Yfix2),title 'disk sd, close again  ;
%
% now try maxing  it...
%
 Yfix3 = imopen(Yfix2,SE);
%subplot(3,3,4),imshow(Yfix3),title 'disk sd, open operation  ;
%
% do it again...
%
 for i = 1:10
  Yfix4 = imdilate(Yfix3,SE);
 end;
%
%subplot(3,3,5),imshow(Yfix4),title 'after dilate';
%

 SE = strel('disk',3);
 for i = 1:3
   Yfix5 = imopen( Yfix4,SE);
 end;
%subplot(3,3,6),imshow(Yfix5),title 'disk  se, after 3  opens  ;
```

```
%-----------------------------------------------------------------
%  try filling in the cell
%
%BWdfill = imfill(Xfix3,'holes );
%subplot(3,3,6), imshow(BWdfill),title('binary image with filled holes');
%BWnobord = imclearborder(BWdfill, 4);
%subplot(3,3,7),imshow(BWnobord),title('cleared border image');
%
% comment out for now....
%
%seD = strel('diamond',1);
%BWfinal = imerode(BWnobord,seD);
%BWfinal = imerode(BWfinal,seD);
%subplot(3,3,7),imshow(BWfinal),title('segmented image');
%BWoutline = bwperim(BWfinal);
%subplot(3,3,8),imshow(BWoutline),title('outline of cell');
%Segout = imadd(BW1, immultiply(BWoutline, 255));
%figure,imshow(Segout),title('outlined original image');
```

```
%*****************************************************************
%*   Clear edges of the images - arbitrary edge sizes            *
%*   Paula Yandow-Reilly                                         *
%    February  2002                                                *
%    Image 2 version  uses Yfix* variable names
%      need to  modify this to do percentages instead of hard
%     coded values
%*****************************************************************
[maxY,maxX] = size(Yfix5);
Yfix7 = Yfix5;
pix_size = maxY * maxX;
% divide the length and width in half, then
% increment or decrement the coordinates to
% find a color change indicating the distance to
% the region of interest
%
minDistH = floor(maxX * .10);
minDistV = floor(maxY * .10);
midX = floor(maxX/2);
midY = floor(maxY/2);
midVal = 0;
begVal = 0;
%
%  figure out what the color difference should be....
R = B(midY,midX,1);
R = double(R);
G = B(midY,midX,2);
G = double(G);
BB = B(midY,midX,3);
BB = double(BB);
midVal = R + (G + 255) + (BB + 510);
R = B(1,1,1);
R = double(R);
G = B(1,1,2);
G = double(G);
BB = B(1,1,3);
BB = double(BB);
begVal = R + (G + 255) + (BB + 510);
color_dist_horizontal = abs(floor(begVal - midVal));
color_dist_vertical   = abs(floor(begVal - midVal))
%-------------------------------------------------------
%  north distance
%-------------------------------------------------------

maxY = double(maxY);
maxX = double(maxX);
%
% starting point:
%
```

```
cntr = 0;
pval = 0;
pvalDiff = 0;
p_pct = 0;
avgpval = 0;
totpval = 0;
firstflag = 0;
upperY = 0;
edgepixY = 0;
maxNorth = 0;
j = maxY;
%
% move toward center looking for pixel color change
%
for i = midY:maxY
   j = j - 1;
   R = B(j,midX,1);
   R = double(R);
   G = B(j,midX,2);
   G = double(G);
   BB = B(j,midX,3);
   BB = double(BB);
   pval = R + (G + 255) + (BB + 510);
   if firstflag == 0
      avgpval = pval;
      firstflag = 1;
   end;
%
   pvalDiff = abs(pval - avgpval);
   p_pct = pvalDiff/color_dist_vertical;
   if (pvalDiff > color_dist_vertical & (cntr > minDistV ) & p_pct > .30)
   pvalDiff
   cntr
   minDistV
   p_pct
      edgepixY = j;
      break;
   end;
   cntr = cntr + 1;
   totpval = totpval + pval;
   avgpval = floor(totpval/cntr);
end;
if  edgepixY == 0
   maxNorth = maxY;
else
   %  find the distance
   upperY = maxY - edgepixY;
   maxNorth = floor(upperY/2);
   maxNorth = maxY - maxNorth
```

```
end;
%-------------------------------------------------------
%  south distance
%-------------------------------------------------------
%
% starting point:
%
cntr = 0;
pval = 0;
pvalDiff = 0;
avgpval = 0;
totpval = 0;
firstflag = 0;
lowerY = 0;
edgepixY = 0;
minSouth = 0;
j = midX;
firstflag = 0;
%
% move toward center looking for pixel color change
%
for i = 1:midY
   R = B(i,j,1);
   R = double(R);
   G = B(i,j,2);
   G = double(G);
   BB = B(i,j,3);
   BB = double(BB);
   pval = R + (G + 255) + (BB + 510);
   if firstflag == 0
      avgpval = pval;
      firstflag = 1;
   end;
%
   pvalDiff = abs(pval - avgpval);
   if pvalDiff > color_dist_vertical         % edge found..
      edgepixY = i;
      pval
      avgpval
      break;
   end;
   cntr = cntr + 1;
   totpval = totpval + pval;
   avgpval = floor(totpval/cntr);
end;
if  edgepixY ==0
   minSouth = 0;
else
   %  find the distance
```

```
   minSouth = floor(edgepixY/2)
end;
%-------------------------------------------------------
%  east distance
%-------------------------------------------------------

maxY = double(maxY);
maxX = double(maxX);
%
% starting point:
%
cntr = 0;
pval = 0;
pvalDiff = 0;
avgpval = 0;
totpval = 0;
firstflag = 0;
upperX = 0;
edgepixX = 0;
maxEast = 0;
j = maxX;
firstflag = 0;
%
% move toward center looking for pixel color change
%
for i = midX:maxX
   j = j - 1;
   R = B(midY,j,1);
   R = double(R);
   G = B(midY,j,2);
   G = double(G);
   BB = B(midY,j,3);
   BB = double(BB);
   pval = R + (G + 255) + (BB + 510);
   if firstflag == 0
      avgpval = pval;
      firstflag = 1;
   end;
%
   pvalDiff = abs(pval - avgpval);
   if pvalDiff > color_dist_horizontal          % edge found..
      edgepixX = j;
      break;
   end;
   cntr = cntr + 1;
   totpval = totpval + pval;
   avgpval = floor(totpval/cntr);
end;
if  edgepixX ==0
```

```
   maxEast = maxX;
else
   %  find the distance
   upperX = maxX - edgepixX;
   maxEast = floor(upperX/2);
   maxEast = maxX - maxEast
end;
%-------------------------------------------------------
%  west distance
%-------------------------------------------------------
%
% starting point:
%
cntr = 0;
pval = 0;
pvalDiff = 0;
avgpval = 0;
totpval = 0;
firstflag = 0;
lowerY = 0;
edgepixX = 0;
minWest = 0;
j = midY;
firstflag = 0;
%
% move toward center looking for pixel color change
%
for i = 1:midX
   R = B(j,i,1);
   R = double(R);
   G = B(j,i,2);
   G = double(G);
   BB = B(j,i,3);
   BB = double(BB);
   pval = R + (G + 255) + (BB + 510);
   if firstflag == 0
      avgpval = pval;
      firstflag = 1;
   end;
%
   pvalDiff = abs(pval - avgpval);
   if pvalDiff > color_dist_horizontal         % edge found..
      edgepixX = i;
%     break;
   end;
   cntr = cntr + 1;
   totpval = totpval + pval;
   avgpval = floor(totpval/cntr);
end;
```

```
if  edgepixX ==0
   minWest = 0
else
   %  find the distance
   minWest = floor(edgepixX/2)
end;
%-------------------------------------------------
% comment out images 11/23 - to work form home
%figure,subplot(1,3,1),imshow(Yfix5), title 'input image  ;
%
for i = 1:maxY
 for j = 1:maxX
  if (i < minSouth) | (j < minWest) | ( i > maxNorth) | (j > maxEast)
     Yfix7(i,j) = 255;
   else
     Yfix7(i,j) = Yfix5(i,j);
   end;
%
 end;   % for j
end;   % for i
%-------------------------
% for smaller images...don't bother trimming the edge...
%
tot_size = maxX * maxY;
if tot_size < 45000
   Yfix7 = Yfix5;
end;
%subplot(1,3,2),imshow(Yfix7),title 'cleared border';
se = strel('disk',4);
Yfix7a = imdilate(Yfix7,se);
%subplot(1,3,3),imshow(Yfix7a),title   final dilate';
```

```
%*****************************************************************
%*   Reverse a black and while image
%*   Paula Yandow-Reilly
%    February  2002
%   rvrs2.m
%   Can't remember the matlab function (imcomplement!!)
%   Assumes you've run test221.m and fixedges2.m first
%   uses massaged global image variable Xfix7a
%
%*****************************************************************
%
%%%%%%%%%Yfix9 = imcomplement(Yfix7a);
%-----------------------------
split_val = 175;
max_val = max(max(Yfix7a));
high_val = max(max(Yfix7a));
high_val = double(high_val);
high_val = floor(high_val * .8)
if max_val < 175
   split__val = high_val
else
   split_val = 175;
end;
Yfix8 = Yfix7a;
[maxLen,maxWid] = size(Yfix8);
for i = 1:maxLen
 for j = 1:maxWid
  if Yfix8(i,j) <= 115
     Yfix9(i,j) = 255;
  else
     if Yfix8(i,j) > 115
        Yfix9(i,j) = 0;
     end;
  end;
 end;
end;
%-----------------------------
% comment out image show to work from home 11/23
%
se7 = strel('disk',3);
Yfix9a = imdilate(Yfix9,se7);
%
%figure,imshow(Yfix9a);
```

```
%*****************************************************************
%*   Clear edges of the images - arbitrary edge sizes            *
%*   Paula Yandow-Reilly                                         *
%    February  2002                                                *
%    Image 2 version  uses Yfix* variable names
%                                                                 *
%*****************************************************************
[maxYlen,maxYwid] = size(Yfix9a);
% comment out images to work from home 11/23/022
%figure,subplot(1,3,1),imshow(Yfix9a), title 'input image ';
%
% increment the min and max values slightly to avoid introducing
% and edge artifact
%
minSouth = minSouth + 5;
minWest  = minWest + 5;
maxEast  = maxEast - 5;
maxNorth = maxNorth - 5;
Yfix9b = Yfix9a;
for i = 1:maxYlen
 for j = 1:maxYwid
if (i < minSouth) | (j < minWest) | ( i > maxNorth ) | (j > maxEast)
     Yfix9b(i,j) = 255;
   else
     Yfix9b(i,j) = Yfix9a(i,j);
   end;
%
 end;    % for j
end;    % for i
% just suing this first subplot
%subplot(1,3,2),imshow(Yfix9b),title 'cleared border';
% se = strel('disk',4);
%%Yfix9c = imdilate(Yfix9b,se);
%subplot(1,3,3),imshow(Yfix9c),title   final dilate';
```

```
%*****************************************************************
%*   rvrs3.m
%*   Paula Yandow-Reilly
%    February  2003
%*****************************************************************
%
%  get image dimensions
%
[maxLen,maxWid] = size(Yfix9b);
%
%  reverse black and white pixels
%
Yfix18 = Yfix9b;
Yfix19 = Yfix9b;
 for i = 1:maxLen
  for j = 1:maxWid
   if Yfix18(i,j) <= 175
      Yfix19(i,j) = 255;
   else
      if Yfix18(i,j) >= 176
         Yfix19(i,j) = 0;
      end;
   end;
  end;
end;
```

```
%*****************************************************************
%*   Edge Marker
%*   Paula Yandow-Reilly
%    April 2002
%    The function expects to be passed the image edge and
%    the original image
%
%    The purpose of this routine is to scan the original image
%    and using the edge image mark the edge on the original
%
%*****************************************************************
function BE = backelim(edgepix,origpix)
%
edgefound  = 0;
%
%dilate the edge to avoid gaps
se = strel('disk',1);
edgepix = imdilate(edgepix,se);

%
% combine the edge image with the original image
%
%  read the original image, using the edge image mark the object
%  edge on the original
%
newpix = origpix;
%
% Make sure the original image doesn't have any values of 255
%
[Yval,Xval,Kval] = size(origpix);
%
for i = 1:Yval
 for j = 1:Xval
  for k = 1:3
     if origpix(i,j,k) == 255
        newpix(i,j,k) = 254;
     end;
  end;
 end;
end;
%
[Ymax,Xmax] = size(edgepix);
%
for i = 1:Ymax
 for j = 1:Xmax
  for k = 1:3
     if edgepix(i,j) == 1
        newpix(i,j,k) = 255;
     end;
```

```
  end;
 end;
end;
%
% comment out to work from home 11.23.02
%figure,imshow(newpix);
BE = newpix;
return;
```

```
%*****************************************************************
%*   Use imdilate to grow edges together
%*   Paula Yandow-Reilly
%    February  2002
%   This script will grow a region until the labelling function, bwlabel
%   returns a value of 1.  This may not result in connection of all edges
%   (i.e. 2 c shaped regions - once one edge connects they are
%   now one blob
%
%*****************************************************************
%
%
%
%  check to see if you even need to run this....
%
count19 = 0;
[m,n] = size(Yfix19);
% comment out so work from home 11/23/02
%figure,imshow(Yfix19);
lab9 = bwlabel(Yfix19,4);
reg_check = max(max(lab9));
se77 = strel('disk',7);
%
Yfix10 = Yfix19;
while reg_check > 1
%
  Yfix10 = imdilate(Yfix10,se77);
  lab9 = bwlabel(Yfix10,4);
  reg_check = max(max(lab9));
%
end;

Yfix10 = bwfill(Yfix10,'holes',4);
%
%  check to see if filling holes filled in the cell you are trying to find...
%
for i = 1:m
  for j = 1:n
      if Yfix10(i,j) ~= 1
         count19 = count19 + 1;
      end;
  end;
end;
count19
if count19 < 100
   Yfix10 = Yfix19;
end;
%
%figure,imshow(Yfix10);
```

```
%*****************************************************************
%*    Compactness Counter
%*    Paula Yandow-Reilly
%     April 2002
%     The function expects to be passed the image edge and
%     the detected object (prior to edge boundary detection
%
%     The purpose of this routine is to count the pixels
%     that make up an object edge. Then the pixels that make up
%     the object are counted (i.e. the area of the object)
%     This information is then used to calculate compactness
%     compactness = (edge boundary ) 2 / area
%
%*****************************************************************
function EC = edgecntr(edgepix,objpix)
%
edgecnt  = 0;
objcnt   = 0;
objComp  = 0;
%
% get EDGE image dimensions
[Ymax,Xmax] = size(edgepix);
%
%  read the edge image, count all white edge pixels
%
for i = 1:Ymax
 for k = 1:Xmax
     if edgepix(i,k) == 1
      edgecnt = edgecnt + 1;
     end;
 end;
end;
%
% get OBJECT image dimensions
[Ymax,Xmax] = size(objpix);
%
%  read the image, convert object pixels to value = 1
%
for i = 1:Ymax
 for k = 1:Xmax
     if objpix(i,k) == 255
        objpixb(i,k) = 1;
     else
        objpixb(i,k) = objpix(i,k);
     end;
 end;
end;
%
%  read the image, with the object detected  count all white object  pixels
```

```
%
for i = 1:Ymax
 for k = 1:Xmax
     if objpixb(i,k) == 1
      objcnt = objcnt + 1;
     end;
 end;
end;
edgecnt
objcnt
objComp = (edgecnt * edgecnt) / objcnt;
EC = objComp
return;
```

```
%*****************************************************************
%    Shape Analysis
%*   Paula Yandow-Reilly
%    July  2002
%    edgeArray.m
%    The function expects to be passed the image edge
%
%    The purpose of this routine is to use the edge pixels to plot the
%    edge and then use convhull to determine the indentation
%    This data is then used to determine the difference between
%    the original shape and the smoothed convexhull shape.  If
%    there are many indentations and protrusions this ration will
%    be low.   If the shape is fairly smooth, this ratio will be
%     be close to 1.
%
%*****************************************************************
function [CH_Ratio] = edgeArray(edgepix)
%
edgecnt  = 0;
objcnt   = 0;
objComp  = 0;
Ysize = 0;
Xsize = 0;
%
% get EDGE image dimensions
[Ymax,Xmax] = size(edgepix);
%edgeArrayY[Ymax];
%edgeArrayX[Xmax];
j = 0;
%
%  read the edge image, store all the white pixel locations
%
for i = 1:Ymax
 for k = 1:Xmax

     if edgepix(i,k) == 1
      j = j+1;
      edgeArrayY(j) = i;
      edgeArrayX(j) = k;
     end;
 end;
end;
%
%  call order array, so the area can be calculated
%
[ordered_X,ordered_Y] = orderArray(edgeArrayX,edgeArrayY);
edgeArrayX = ordered_X;
edgeArrayY = ordered_Y;
```

```
%figure,plot(edgeArrayY,edgeArrayX,'ro');
[k,ch_ar] = convhull(edgeArrayY,edgeArrayX);
%figure,plot(edgeArrayY(k),edgeArrayX(k),'r-');
%figure,plot(edgeArrayY(k),edgeArrayX(k),'r- ,edgeArrayY,edgeArrayX,'b*');
or_area = polyarea(edgeArrayY,edgeArrayX);

Ysize = size(edgeArrayY);
Xsize = size(edgeArrayX);
max(edgeArrayY(k));
min(edgeArrayY(k));
max(edgeArrayX(k));
min(edgeArrayX(k));
size(edgeArrayY(k));
size(edgeArrayX(k));
ch_area = polyarea(edgeArrayY(k),edgeArrayX(k));
x = 'the convex hull area is ';
 ch_area;
x = 'the original edge area is  ;
 or_area;
area_diff = ch_area - or_area;
CH_Ratio = or_area/ch_area;
%
%  try counting pixels inside of the polygons...
origIn = inpolygon(Xsize,Ysize,edgeArrayY,edgeArrayX);
EX = edgeArrayX(k);
EY = edgeArrayY(k);
OX = edgeArrayX;
OY = edgeArrayY;
```

```
%*****************************************************************
%*  orderArray
%*
%    November  2002
%    This function passes back the edge coordinates ordered so they
%    will plot correctly
%
%*****************************************************************
function [CX,CY] = orderArray(Xarray,Yarray)
%
% start with the first coords in the arrays, then find the closest pixel,
% add that to the newly ordered array.  The first coords will also be the last
% to close the polygon
%
[dimx,lenx] = size(Xarray);
[dimy,leny] = size(Yarray);
%
minDist = 1000;
diff = 0;
diffX = 0;
diffY = 0;
nineFill = 9999999;
newCount = 1;
nextPos = 0;
newlen = lenx + 1;
% assign the  starting coordinates:
%
% make sure to remove last memeber if it equals the first
%
if Xarray(1) == Xarray(leny) & Yarray(1) == Yarray(leny)
   leny = leny - 1;
end;
newX = zeros(1,newlen);
newY = zeros(1,newlen);
% seed loop with first coordinate points
newX(1) = Xarray(1);
newY(1) = Yarray(1);
Xarray(1) = nineFill;
Yarray(1) = nineFill;
current = 1;
%
%  loop through  number of coordinates to assign to new array
%
for i = 1:leny
newCount = newCount + 1;
minDist = 1000;
nextPos = 1;
%
%  loop through x and y arrays and look for next closest pixel
```

```
    for j = 1:leny
      if Xarray(j) ~= nineFill
       diffX = abs(newX(current) - Xarray(j));
       diffY = abs(newY(current) - Yarray(j));
       diff = diffX + diffY;
       if diff <= minDist
          nextPos = j;
          minDist = diff;
       end;
      end;    % if Xarray(j)...
    end;      % for j loop...
%
%----------------ok should have the next closest one now....
% assign closest pixel to the new arrays
   if nextPos ~= nineFill & nextPos > 0 & Xarray(nextPos) ~= nineFill
        current = current + 1;
        newX(current) = Xarray(nextPos);
        newY(current) = Yarray(nextPos);
%         flag the  pixel values so it won't be used again
        Xarray(nextPos) = nineFill;
%       Yarray(nextPos)  = nineFill;
   end;  % if nextPos...
%
end;  % for i  continue looping through the arrays
%
%  on completion, add coords to close the loop...
%
    newX(current + 1) = newX(1);
    newY(current + 1) = newY(1);

CX = newX;
CY = newY;
```

```
%*****************************************************************
%*    Background Eliminator
%*    Paula Yandow-Reilly
%     April 2002
%     The function expects to be passed the image edge and
%     the original image
%
%     The purpose of this routine is to scan the original image
%     and using the edge remove the background from the original
%
%*****************************************************************
function BE = backelim(edgepix)
%
% now that you have the original image with the edge marked, remove the background
% outside of the edge
%
[Ymax,Xmax,Kmax] = size(edgepix);

pixnobg = edgepix;                    % the working image
edgefound = 0;                        % flag to indicate when the edge is encountered
checker = 0;                          % debug
checker2 = 0;                         % debug
prevpixel = 0;                        % flag to det if you are entering or leaving edge
jnext = 0;                            % start position to look ahead
%
for i = 1:Ymax
 for j = 1:Xmax
  for k = 1:3
%
% debug --------------------
%
%     checker = checker + 1;
%     checker2 = checker2 + 1;
%     if checker > 1000
%        checker2
%        checker = 0;
%     end;
%---------------------------
     if jnext <  Xmax
       jnext = j + 1;
     else
       jnext = j;
     end;

%
     if edgepix(i,j,k) == 255 & edgefound == 0 & edgepix(i,jnext,k) ~=255
        %
        %  check to see if you are just grazing the edge of the edge or if you
        %  are going to encounter an edge on the other side
```

```
        %   don't turn on the flag if you don't find another edge pixel..
            for jj = jnext:Xmax
              if edgepix(i,jj,k) ==  255
                edgefound = 1;
               end;
            end;
     else
        if edgepix(i,j,k) ~=255 & edgefound == 0
           pixnobg(i,j,k) = 0;
        else
           if edgepix(i,j,k) == 255 & edgefound == 1 & edgepix(i,jnext,k) ~= 255
              edgefound = 0;
           end;
        end;
     end;

   end;
  end;
end;
% comment out to work from home 11.23.02
%
%figure,imshow(pixnobg);
BE = pixnobg;
return;
```

```
%*****************************************************************
%*   color_text_prep.m
%*   Paula Yandow-Reilly
%    January 2003
%
%    The purpose of this routine is to use the edge pixels to
%    determine which pixels are in the region of interest
%    The function expects to be passed the edge and the color
%    segmented image.
%
%   Once these pixels are identified, routines are run to evaluate
%     COLOR
%
%    These pixela are then examined for relationships in the red,
%    green, and blue components to determine percent of red and
%    black in the lesion.  Another component in accessing if
%    a lesion is cancerous is the evidence of regression. In most
%    caucasion skin, the regression will be a pink color.  However,
%    the regressed area may not be segmented out with the lesion.
%    So this metric (#pink pixels/#total pixels) gives some measure
%    that there might be regression present.
%
%  and TEXTURE

%    prepares a matrix for the co-occurrence matrix evaluations
%    the function also simplifies the data in the sub matrix to contain
%    fewer grey levels to make computation reasonable.
%
%
%*****************************************************************
function [red_pct,black_pct,pink_pct,no_colors,t,x,y] = color_text_prep(edgepix,segpix)
%
% this variable controls the size of the texture sample
%
txtSize  = 129;
%
edgecnt  = 0;
objcnt   = 0;
objComp  = 0;
Ysize = 0;
Xsize = 0;
Ystart = 0;
Xstart = 0;

%
% turn the segmented RGB image grayscale:
% (this is used to find the lesion pixels)
%
M = rgb2gray(segpix);
```

```
M = double(M);
M = mat2gray(M);
%
% get EDGE image dimensions
[Ymax,Xmax] = size(edgepix);
%
% analyze the  pixels inside of the polygons...
%
j = 0;
%
%  read the edge image, store all the white pixel locations
%
for i = 1:Ymax
 for k = 1:Xmax
     if edgepix(i,k) == 1
      j = j+1;
      edgeArrayY(j) = i;
      edgeArrayX(j) = k;
     end;
 end;
end;
%%
%
j = 0;
cntr = 0;
fullX = 0;
fullY = 0;
cntrM = 0;
i = 0;
l = 0;
Xline = [1:Xmax];
fullX = [1:Xmax];
datestr(now)
Yend = Ymax - 1;
for j = 1:Yend
    fullX = [fullX Xline];
end;
%
display 'finished creating fullX'
size(fullX)
datestr(now)
fullY = [];
%-------------------------
for i = 1:Ymax
 fullY = [fullY repmat(i,[1,Xmax])];
end;
%_____________________________________
size(fullY)
datestr(now)
```

```
% order the array...
[ordered_X,ordered_Y] = orderArray(edgeArrayX,edgeArrayY);
%
origIn = inpolygon(fullY,fullX,ordered_Y,ordered_X);
%
size(origIn)
m = 0;
l = 0;
i = 0;
j = 0;
%
%  assign the inside pixels to the output matrix
%
%-------------------------------------------------------------
[Ymax,Xmax] = size(M);
Yin = 0;
Xin = 0;
C = 0;
k = 0;
for i = 1:Ymax
 for j = 1:Xmax
   C = C+1;
   if origIn(C) == 1
      k = k+1;
      Xin(k) = j;
      Yin(k) = i;
   end;
 end;
end;
%-----------------------------------
%

% now put use the X and Y arrays to locate the lesion pixels and do analysis
%
red_cnt = 0;
black_cnt = 0;
[q,totPixels] = size(Xin);
 for i= 1:totPixels
   r1 =  segpix(Yin(i),Xin(i),1);          % red component
   g1 =  segpix(Yin(i),Xin(i),2);          % green component
   b1 =  segpix(Yin(i),Xin(i),3);          % blue component
   r1 = double(r1);
   g1 = double(g1);
   b1 = double(b1);
   if g1 ~= 0 & r1 ~= 0
      rgDiff = g1/r1;
   else
      rgDiff = 0;
   end;
```

```
   if b1 ~= 0 & r1 ~= 0
      rbDiff = b1/r1;
   else
      rbDiff = 0;
   end;
    if rgDiff < .49 & rbDiff < .49
       red_cnt = red_cnt + 1;
    end;
    if r1<.4 & g1<.4 & b1< .4
       black_cnt = black_cnt + 1;
    end;
 end;
%--------------------------------------------------
%
% initialize the bin value to the max value
for i = 1:1525
  bin(i) = 0;
end;
binval = 0;
color_count = 0;
R = 0;
G = 0;
B = 0;

% Bin the image colors by 16 possible color  ranges

 for i = 1:totPixels
    R =  segpix(Yin(i),Xin(i),1);
    R = double(R);
    G =  segpix(Yin(i),Xin(i),2);
    G = double(G);
    B =  segpix(Yin(i),Xin(i),3);
    B = double(B);
%
%  separate the various color values by adding 255 to move them into different number ranges to distinguish
the combinations
    G = G + 255;
    B = B + 510;
    binval = R + G + B;
    if binval ~= 0 & binval <= 1525
       bin(binval) = bin(binval) + 1;
    end;
 end;
 [q,maxbin] = size(bin)
%
  for i = 1:maxbin
            if bin(i) > 0
               color_count = color_count + 1;
            end;
```

```
        end;
%
%  check for pink pixels to check for regression
%
pink_cnt = 0;
  for i = 1:totPixels
    R = segpix(Yin(i),Xin(i),1);
    R = double(R);
    G = segpix(Yin(i),Xin(i),2);
    G = double(G);
    B = segpix(Yin(i),Xin(i),3);
    B = double(B);
    if R ~= 0 & G ~= 0
       gvsred = G / R;
    else
       gvsred = 0;
    end;
    if R ~= 0 & B ~= 0
       bvsred = B / R;
    else
       bvsred = 0;
    end;

  if R > 195  & gvsred > .86 & gvsred < .90 & bvsred > .86 & bvsred < .90
     pink_cnt = pink_cnt + 1;
  end;
  end;
%-------------------------------------------------------------------------
%-----------------------------TEXTURE ROUTINE -----------------------------
%-------------------------------------------------------------------------
%

% now put x and y together to make the output matrix of the lesion pixels
% if necessary, reduce the size of the output matrix - all output should
% be the same size, to make comparisons possible
% in addition, the matrix must be square to allow the texture calcs to leverage
% matlab's powerful functions
%
maxX = (max(Xin) - min(Xin));
maxY = (max(Yin) - min(Yin));
if maxX > maxY
   maxY = maxX;
else
   maxX = maxY;
end;
for i=1:txtSize
 for j = 1:txtSize
  txtmat(i,j) = 7;
 end;
```

```
end;
%
minX = min(Xin) - 1;
minY = min(Yin) - 1;
[q,maxNew] = size(Xin);
for i=1:maxNew
  newx = Xin(i) - minX;
  newy = Yin(i) - minY;
  txtmat(newx,newy) = M(Yin(i),Xin(i));
end;
%
%this step ensures that all output matrices are the same size
% (this size picked via heuristics)
%
 for i = 1: txtSize
   for j = 1:txtSize
        txtOut(i,j) = txtmat(i,j);
     end;
   end;
%
% create a textel to use to fill in areas not covered by
% original lesion
%
[Ymax,Xmax] = size(txtOut);
midPos = round(Ymax/2);
submat = imcrop(txtOut,[midPos,midPos,25,25]);
[YY,XX] = size(submat);
subsize = YY * XX;
textel = reshape(submat,1,subsize);
%
% now fill in blank areas with the Textel values
% also the white edge pixels which are not part of the
% original image
%
[q,textelMax] = size(textel);
textelMax = textelMax - 1;
%
 for i = 1:Ymax
   for j = 1:Xmax
     if txtOut(i,j) == 7 | txtOut(i,j) == 1
        k = k + 1;
        if k > textelMax
           k = 1;
        end;
        txtOut(i,j) = textel(k);
     end;
   end;
 end;
%
```

```
%  now reduce the sub matrix to 10 greylevels to make computation reasonable
%
txtOut = grayslice(txtOut,10);
q = find(txtOut == 0);
r = find(txtOut == 1);
s = find(txtOut == 2);
t = find(txtOut == 3);
u = find(txtOut == 4);
v = find(txtOut == 5);
x = find(txtOut == 6);
w = find(txtOut == 7);
y = find(txtOut == 8);
z = find(txtOut == 9);
%
% move the levels up by one (can't index matrix with zero)
%
txtOut(q) = 1;
txtOut(r) = 2;
txtOut(s) = 3;
txtOut(t) = 4;
txtOut(u) = 5;
txtOut(v) = 6;
txtOut(w) = 7;
txtOut(x) = 8;
txtOut(y) = 9;
txtOut(z) = 10;
%
%
%------------- send values back to calling routine
%
%     COLOR
totPixels = double(totPixels);
rec_cnt = double(red_cnt);
black_cnt = double(black_cnt);
pink_cnt = double(pink_cnt);
%
 red_pct = red_cnt/totPixels;
 black_pct = black_cnt/totPixels;
 pink_pct = pink_cnt/totPixels;
 no_colors = color_count;
%
%   TEXTURE
%
t = txtOut;
x = fullX;
y = fullY;
```

```
%  co_occur.m  -  This function calculates the co-occurrence matrix for
%                 a given image.  It starts by creating an 8x8 matrix M
%                 initialized to zero, and then increments the count of each
%                 entry in M every time a pixel has a gray level that occurs
%                 a certain distance and orientation to another pixel.
%                 The distance can be from 1 to min(m,n).  The default is 1.
%                 The orientation can be north, south, east, or west.  The
%                 default is east.


function y = co_occur(x,d,o)

[m,n] = size(x);
M = zeros(10,10);            % Initialize co-occurrence matrix M


% error checking and setting of parameters
if nargin > 3, error('Too many arguments.'); end
if nargin < 3, o='east'; end
if nargin < 2, d=1; end
if strcmp(d,'east') | strcmp(d,'west ) | strcmp(d,'south') | ...
   strcmp(d,'north'), d=1; end
if d >= m | d >= n, error('Distance too large. ); end



if strcmp(o,'east')                          % Orientation is East
  r=1;                                       % Initial pixel, row value
  c=1;                                       % Initial pixel, column value
  while r <= m                               % Row boundary checking
     i = x(r,c);                             % Set i index into M
     c = c + d;                              % Find the pixel to compare
     j = x(r,c);                             % Set j index into M
     M(i,j) = M(i,j) + 1;                    % Increment M
     if c == n                               % Column boundary checking
        c = 1;                               % Reset the column
        r = r + 1;                           % Increment row
     else
        c=c-d+1;                             % Next pixel, column value
     end
   end
elseif strcmp(o,'west')                      % Orientation is West
  r=1;
  c=n;
  while r <= m
     i = x(r,c);
     c = c - d;
     j = x(r,c);
     M(i,j) = M(i,j) + 1;
```

```
      if c == 1
         c = n;
         r = r + 1;
      else
         c=c+d-1;
      end
    end
elseif strcmp(o,'south')                    % Orientation is South
  r=1;
  c=1;
  while c <= n
      i = x(r,c);
      r = r + d;
      j = x(r,c);
      M(i,j) = M(i,j) + 1;
      if r == m
         r = 1;
         c = c + 1;
      else
         r=r-d+1;
      end
    end
elseif strcmp(o,'north')                    % Orientation is North
  r=m;
  c=1;
  while c <=n
      i = x(r,c);
      r = r - d;
      j = x(r,c);
      M(i,j) = M(i,j) + 1;
      if r == 1
         r = m;
         c = c + 1;
      else
         r = r+d-1;
      end
    end
else
   error('Invalid direction:north, south, east, or west please')
end

Y = M;                                      % Return the matrix M - it is the
                                            % co-occurrence matrix
```

```
%*****************************************************************
%*    texture.m
%*    Paula Yandow-Reilly
%     January 2003
%     This function expects to be passed a sub matrix created by
%     texture_prep.m
%     the function sends the modified grayscale image to various texture
%     algorithms to retrieve texture measurements.
%
%*****************************************************************
function [ENERGY,ENTROPY,DISTANCE,CONTRAST] = texture(txtmatrix)
%
M = co_occur(txtmatrix);
ENERGY = energy(M);
ENTROPY = entropy(M);
DISTANCE =  distance(M);
CONTRAST = contrast(M);
```

```
%-------------------------------------------------------------------------
%  energy.m - This function computes the total energy of the co-occurrence
%             matrix for a gray level image.  Energy is a measure of image
%             homogeneity.  The higher the energy, the more homogenous it is.
%             courtesy Roxanne Canosa
%-------------------------------------------------------------------
function y = energy(x)

xsquared = x^2;
y = sum(sum(xsquared));
```

```
%  entropy.m - This project finds the entropy of the co-occurrence matrix.
%              Entropy is a measure of the amount of "disorder" in the
%              image, and roughly corresponds to texture (no pun intended).
%              First, the entries in the co-ocurrence matrix must be converted
%              from counts to probabilities (by dividing by the total number
%              of counts), then the entropy can be calculated.
%              courtesy Roxanne Canosa

function y = entropy(x)

[m,n] = size(x);
E = 0;
M = sum(sum(x));

P = x/M;                        %  convert counts to probabilities

for i=1:m
   for j=1:n
      if P(i,j) ~= 0
         E = E + P(i,j) * log2(P(i,j));
      end
   end
end

y = E;
```

```
%  distance.m - This function calculates a scalar feature of the co-occurrence
%               matrix called distance.  It is the weighted absolute average
%               distance of the gray levels from the diagonal.  A high value
%               of distance indicates a lack of smoothness in the image.
%               courtesy Roxanne Canosa

function y = distance(x)

[m,n] = size(x);
M = 0;
P = 0;

for i = 1:m
   for j = 1:n
      P = P + abs(i-j) * x(i,j);
   end
end

M = sum(sum(x));

y = 1/M * P;
```

```
%-------------------------------------------------------------------------
%  contrast.m - This function computes the contrast of the co-occurrence
%               matrix for a gray level image by summing the energy multiplied
%               by the x/y difference squared.  This difference gives informaion
%               about how much contrast there is in the image.
%-------------------------------------------------------------------
function c = contrast(x)

C = 0;
k = 0;
[m,n] = size(x);
M = sum(sum(x));

for i = 1:m
  for j = 1:n
    t1 = i + 1;
    t2 = j + 1;
    if t1 <= m
      if t2 <= n
         C = C + (abs(x(t1,t2) - x(i,j)))^2;
      end;
    end;
  end;
end;
    C = M * C;
c = C;
```

```
%*****************************************************************
%*   classify.m
%*   Paula Yandow-Reilly
%    February 2003
%    This script  assigns a Feature Warning Level value based on
%    database results for Melanoma vs Mole.  It uses the most indicative
%    vectors produced for shape, color, and texture
%    The Feature Warning Level classification has three levels:
%
%     RED   - has warning levels in all three categories: shape, color, and texture
%     AMBER - has warning levels in at least 2 categories
%     GREEN - doesn't appear to have any warning levels
%
%*****************************************************************
%
FWL = 'GREEN';
img_no_colors
img_area_ratio
img_texture_distance
img_compactness

% check for RED - 3 out of 4 will turn it RED
R = 0;
if img_no_colors > 350
   R = R + 1;
end;
if img_area_ratio < .96
   R = R + 1;
end;
if img_texture_distance > .5
   R = R + 1;
end;
if img_compactness > 11.5
   R = R + 1;
end;
Y = 0;
if img_no_colors > 350
   Y = Y + 1;
end;
if img_area_ratio < .98
   Y = Y + 1;
end;
if img_texture_distance > .5
   Y = Y + 1;
end;
if img_compactness > 11.5
   Y = Y + 1;
end;
```

```
if R >= 3
   FWL = 'RED';
else
   if Y >= 2
      FWL = 'AMBER';
   end;
end;
FWL
```

```
%-------------------------------------------------------
%
% dbWrite.m
% Paula Yandow-Reilly
% February 2003
%
%  This function accepts feature vector parms in and
% writes them to the file lesion.mat.  The backup for
% this file is called lesion.bkp.
% Currently the function only appends records.
% the function sends back a status flag to indicate record
% was written ok
%-------------------------------------------------------
function [status]  = dbWrite(fi,di,ni,ri,bi,pi,ci,ari,tei,teni,tdi,tci)
%
   fi = [fi '&'];
   di = [di '#'];
   s = 0;
%
%  write the char fields record
%
   fid = fopen('lesion.mat ,'a');
   fwrite(fid,fi,'8*char');
   fwrite(fid,di,'8*char',1);
%
%  write the number fields record
%
    fwrite(fid,ni,'8*float64 );
    fwrite(fid,ri,'8*float64 );
    fwrite(fid,bi,'8*float64 );
    fwrite(fid,pi,'8*float64 );
    fwrite(fid,ci,'8*float64 );
    fwrite(fid,ari,'8*float64');
    fwrite(fid,tei,'8*float64 );
    fwrite(fid,teni,'8*float64');
    fwrite(fid,tdi,'8*float64');
s = fwrite(fid,tci,'8*float64');
%
   fclose(fid);
%
%-------------------------------------------------------------------------
status = s;
```

```
%-------------------------------------------------------
%
% dbReadMatchVote.m
% Paula Yandow-Reilly
% February 2003
%
%  This function reads records from the lesion.mat file
% database.  The function accepts image features as input
% and will loop through the data in the file checking for a
% match.  The function will return the name of the closest :
% match, or if the gap is too large will return "no match".
% The system uses a simple voting system to determine the
% closest match.

%-------------------------------------------------------
function [match] = dbio(fi,di,ni,ri,bi,pi,ci,ari,tei,teni,tdi,tci)
%
%
current_match = 0;
diff_mat = 0;
vote_pix = 0;
best_vote = 0;
%
min_ni = 999999999;
min_ri = 999999999;
min_bi = 999999999;
min_pi = 999999999;
min_ci = 999999999;
min_ari = 999999999;
min_tei = 999999999;
min_teni = 999999999;
min_tdi = 999999999;
min_tci = 999999999;
%
input_total = 0;
read_total = 0;
diff_total = 0;
min_total = 999999999999;
min_sum = 0;
input_total = ni+ci+ari+ teni + tdi;
allowed_min = input_total - (input_total * .2);
allowed_max = input_total + (input_total * .2);
first_read = 'Y';
char_cnt  = 0;
prev_pos = 0;
first_rec  = 'Y';
first_done = 'N';
first_time = 'Y';
first_time2 = 'Y';
```

```
first_time3 = 'Y';
first_time4 = 'Y';
last_done = 'N';
charlen = 35;
%
% record layout
%
rec_filename = 0;
rec_diagnosis = 0;
rec_no_colors = 0;
rec_pct_red = 0;
rec_pct_black = 0;
rec_pct_pink = 0;
rec_compactness = 0;
rec_area_ratio = 0;
rec_texture_energy = 0;
rec_texture_entropy = 0;
rec_texture_distance = 0;
rec_texture_contrast = 0;
%
%-------------------------------------------------------------------------
%
%  read records...
%
fid = fopen('lesion.mat');
pos = 0;
status = fseek(fid,pos,'bof');
while status == 0
   %
   % first get the 2 char fields:
   %
   tline = fread(fid,charlen,'8*char');
   [q,r] = size(tline);
if r ~= 0 & q ~= 0
  for i = 1:q
    char_cnt = char_cnt + 1;
       if char(tline(i)) ~= '&' & first_done == 'N'
          if first_time == 'Y'
            rec_filename = [char(tline(i))];
            first_time = 'N';
          else
            rec_filename = [rec_filename char(tline(i))];
          end;
       else
          first_done = 'Y';
       end;
       if char(tline(i)) == '#'
          last_done = 'Y';
          break;
```

```
      end;
    if char(tline(i)) ~= '#  & first_done == 'Y' & last_done == 'N' & char(tline(i)) ~= '&
      if first_time2 == 'Y'
         first_time2 = 'N';
      else
        if first_time3 == 'Y'
           rec_diagnosis = [char(tline(i))];
           first_time3 = 'N';
        else
           rec_diagnosis = [rec_diagnosis char(tline(i))];
        end;
      end;
    end;
  end;  % for
 end;   % if
 if rec_filename == 0
    break
 end;
%-------------------------------------------------------
% now read the associated feature vectors
%
nline = 0;
%if first_rec == 'Y'
  char_cnt = char_cnt - 1;
%  first_rec = 'N';
%end;
pos = prev_pos + char_cnt + 1;
char_cnt = 0;
status = fseek(fid,pos,'bof');
%
  nline = fread(fid,10,'8*float64');
  [q,r] = size(nline);
  if r ~= 0 & q ~= 0
    rec_no_colors        = nline(1);
    rec_pct_red          = nline(2);
    rec_pct_black        = nline(3);
    rec_pct_pink         = nline(4);
    rec_compactness      = nline(5);
    rec_area_ratio       = nline(6);
    rec_texture_energy   = nline(7);
    rec_texture_entropy  = nline(8);
    rec_texture_distance = nline(9);
    rec_texture_contrast = nline(10);
  end;
%-------------------------------------------------------
%     Match routine......
%
vote_pix = 0;
%
```

```
diff_amt = abs(rec_no_colors - ni);
%
if diff_amt <= min_ni
   vote_pix = vote_pix + 1;
   min_ni = diff_amt;
end;
%
diff_amt = abs(rec_pct_red - ri);
if diff_amt <= min_ri
   vote_pix = vote_pix + 1;
   min_ri = diff_amt;
end;
%
diff_amt = abs(rec_pct_black - bi);
if diff_amt <= min_bi
   vote_pix = vote_pix + 1;
   min_bi = diff_amt;
end;
%
diff_amt = abs(rec_pct_pink - pi);
if diff_amt <= min_pi
   vote_pix = vote_pix + 1;
   min_pi = diff_amt;
end;
%
diff_amt = abs(rec_compactness - ci);
if diff_amt <= min_ci
   vote_pix = vote_pix + 1;
   min_ci = diff_amt;
end;
%
diff_amt = abs(rec_area_ratio - ari);
if diff_amt <= min_ari
   vote_pix = vote_pix + 1;
   min_ari = diff_amt;
end;
%
diff_amt = abs(rec_texture_energy - tei);
if diff_amt <= min_tei
   vote_pix = vote_pix + 1;
   min_tei = diff_amt;
end;
%
diff_amt = abs(rec_texture_entropy - teni);
if diff_amt <= min_teni
   vote_pix = vote_pix + 1;
   min_teni = diff_amt;
end;
%
```

```
diff_amt = abs(rec_texture_distance - tdi);
if diff_amt <= min_tdi
   vote_pix = vote_pix + 1;
   min_tdi = diff_amt;
end;
%
diff_amt = abs(rec_texture_contrast - tci);
if diff_amt <= min_tci
   vote_pix = vote_pix + 1;
   min_tci = diff_amt;
end;
%
%  skip the vote for the first record to avoid
%  casting all 9 votes for first image
%
if first_time4 = 'Y'
   first_time4 = 'N';
else
 if best_vote < vote_pix
    best_vote = vote_pix;
    current_match = rec_filename;
end;


%-----------------------------------------------------
 pos = ftell(fid);
prev_pos = ftell(fid);
 status = fseek(fid,pos,'bof');
%
% reset for reading char fields for next record
% clear input fields for the next record
%
first_done = 'N';
last_done = 'N';
first_time = 'Y';
first_time2 = 'Y';
first_time3 = 'Y';
rec_filename = 0;
rec_diagnosis = 0;
rec_no_colors = 0;
rec_pct_red = 0;
rec_pct_black = 0;
rec_pct_pink = 0;
rec_compactness = 0;
rec_area_ratio = 0;
rec_texture_energy = 0;
rec_texture_entropy = 0;
rec_texture_distance = 0;
rec_texture_contrast = 0;
```

```
end;     % while
%
fclose(fid);
%-----------------------------------------------------
%  determine if closest match in within a reasonable range
%  a reasonable range is determined to be the average values
%  for melanoma images for number of colors + compactness +
%  area_ratio + texture_distance...+/- 20 percent
%
if best_vote > 2
   match = current_match;
else
   match = 'unknown';
end;
%
%-----------------------------------end --------------------
```

```
%-------------------------------------------------------
%
% dbReadMatch.m
% Paula Yandow-Reilly
% February 2003
%
%  This function reads records from the lesion.mat file
% database.  The function accepts image features as input
% and will loop through the data in the file checking for a
% match.  The function will return the name of the closest :
% match, or if the gap is too large will return "no match"
% The gaps are determined by the difference between the mean values
% for melanoma images vs mean values for benign images.
%-------------------------------------------------------
function [match] = dbio(fi,di,ni,ri,bi,pi,ci,ari,tei,teni,tdi,tci)
%
% flags used to find the end of the character fields
%
input_total = 0;
read_total = 0;
diff_total = 0;
min_total = 999999999999;
min_sum = 0;
input_total = ni+ci+ari+ teni + tdi;
allowed_min = input_total - (input_total * .2);
allowed_max = input_total + (input_total * .2);
first_read = 'Y';
char_cnt  = 0;
prev_pos = 0;
first_rec  = 'Y';
first_done = 'N';
first_time = 'Y';
first_time2 = 'Y';
first_time3 = 'Y';
last_done = 'N';
charlen = 35;
%
% record layout
%
rec_filename = 0;
rec_diagnosis = 0;
rec_no_colors = 0;
rec_pct_red = 0;
rec_pct_black = 0;
rec_pct_pink = 0;
rec_compactness = 0;
rec_area_ratio = 0;
rec_texture_energy = 0;
rec_texture_entropy = 0;
```

```
rec_texture_distance = 0;
rec_texture_contrast = 0;
%
%-------------------------------------------------------------------------
%
%  read records...
%
fid = fopen('lesion.mat );
pos = 0;
status = fseek(fid,pos,'bof');
while status == 0
   %
   % first get the 2 char fields:
   %
   tline = fread(fid,charlen,'8*char');
   [q,r] = size(tline);
if r ~= 0 & q ~= 0
  for i = 1:q
    char_cnt = char_cnt + 1;
       if char(tline(i)) ~= '&' & first_done == 'N'
          if first_time == 'Y'
            rec_filename = [char(tline(i))];
            first_time = 'N';
          else
            rec_filename = [rec_filename char(tline(i))];
          end;
       else
          first_done = 'Y';
       end;
       if char(tline(i)) == '#'
          last_done = 'Y';
          break;
       end;
     if char(tline(i)) ~= '#' & first_done == 'Y' & last_done == 'N' & char(tline(i)) ~= '&'
       if first_time2 == 'Y'
          first_time2 = 'N';
       else
         if first_time3 == 'Y'
            rec_diagnosis = [char(tline(i))];
            first_time3 = 'N';
         else
            rec_diagnosis = [rec_diagnosis char(tline(i))];
         end;
       end;
     end;
  end;  % for
 end;   % if
 if rec_filename == 0
    break
```

```
 end;
%-------------------------------------------------------
% now read the associated feature vectors
%
nline = 0;
%if first_rec == 'Y'
  char_cnt = char_cnt - 1;
%  first_rec = 'N';
%end;
pos = prev_pos + char_cnt + 1;
char_cnt = 0;
status = fseek(fid,pos,'bof');
%
  nline = fread(fid,10,'8*float64 );
  [q,r] = size(nline);
  if r ~= 0 & q ~= 0
    rec_no_colors        = nline(1);
    rec_pct_red          = nline(2);
    rec_pct_black        = nline(3);
    rec_pct_pink         = nline(4);
    rec_compactness      = nline(5);
    rec_area_ratio       = nline(6);
    rec_texture_energy   = nline(7);
    rec_texture_entropy  = nline(8);
    rec_texture_distance = nline(9);
    rec_texture_contrast = nline(10);
  end;
%-------------------------------------------------------
%     Match routine......
%
read_total = rec_no_colors + rec_compactness + rec_area_ratio + rec_texture_entropy + rec_texture_distance;
diff_total = abs(read_total - input_total);
if diff_total < min_total
   min_total = diff_total;
   min_sum = read_total;
   min_file = rec_filename;
end;

%-------------------------------------------------------
 pos = ftell(fid);
prev_pos = ftell(fid);
 status = fseek(fid,pos,'bof');
%
% reset for reading char fields for next record
% clear input fields for the next record
%
first_done = 'N';
last_done = 'N';
first_time = 'Y';
```

```
first_time2 = 'Y';
first_time3 = 'Y';
rec_filename = 0;
rec_diagnosis = 0;
rec_no_colors = 0;
rec_pct_red = 0;
rec_pct_black = 0;
rec_pct_pink = 0;
rec_compactness = 0;
rec_area_ratio = 0;
rec_texture_energy = 0;
rec_texture_entropy = 0;
rec_texture_distance = 0;
rec_texture_contrast = 0;
end;     % while
%
fclose(fid);
%-------------------------------------------------------
%  determine if closest match in within a reasonable range
%  a reasonable range is determined to be the average values
%  for melanoma images for number of colors + compactness +
%  area_ratio + texture_distance...+/- 20 percent
%
if min_sum >= allowed_min & min_sum <= allowed_max
   match = min_file;
else
   match = 'unknown';
end;
%
%-----------------------------------end --------------------
```