Recovering Shading and Reflectance from a Single Image

by

Marshall Friend Tappen

B.S. Computer Science
Brigham Young University, 2000

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of
Master of Science in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2002

© Massachusetts Institute of Technology 2002. All rights reserved.

Author . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

Department of Electrical Engineering and Computer Science

August 9, 2002

Certified by . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

Edward H. Adelson
Professor of Vision Sciences
Thesis Supervisor

Certified by . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

William T. Freeman
Associate Professor of Electrical Engineering and Computer Science
Thesis Supervisor

Accepted by . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

Arthur C. Smith
Chairman, Department Committee on Graduate Students
Recovering Shading and Reflectance from a Single Image

by

Marshall Friend Tappen

Submitted to the Department of Electrical Engineering and Computer Science on August 9, 2002, in partial fulfillment of the requirements for the degree of Master of Science in Electrical Engineering and Computer Science

Abstract

Images are the result of the interaction of many different characteristics of the scene. In this thesis, we present a system that recovers two characteristics of the scene, the shading and reflectance of the scene, from a single image. In addition, we show that this system is able to achieve good results on real images.

The shading and reflectance of a scene are recovered by decomposing the image into two images, one image representing the scene as if every point had the same reflectance and a second image representing just the reflectance at every point.

The shading and reflectance images are recovered by classifying every derivative in the image as being caused by either shading or a reflectance change. After every derivative has been assigned a cause, the shading image is reconstructed from the derivatives labelled as shading derivatives. The reflectance image is recovered in the same manner.

The classifications are made using both color and gray-scale information in the image. We also show how to use machine learning techniques to find the classifier which uses gray-scale image information. The classifications in ambiguous areas of the image are improved by using the Generalized Belief Propagation algorithm to propagate information along image contours.

Thesis Supervisor: Edward H. Adelson
Title: Professor of Vision Sciences

Thesis Supervisor: William T. Freeman
Title: Associate Professor of Electrical Engineering and Computer Science
Acknowledgments

I would like to thank my advisors, Edward Adelson and William Freeman, for their guidance, insights, and for all of their help in editing this thesis. Thanks to them, I have learned an incredible amount in the time that I have been at MIT.

Many thanks are in order to Ron Dror for letting me bounce many a half-formed idea off of him. I would also like to thank Antonio Torralba for many helpful discussions as I was writing this thesis.

The work described here was begun while I was an intern at the Mitsubishi Electric Research Laboratory and their support is gratefully acknowledged. Support from a Department of Defense NDSEG fellowship and NTT is also gratefully acknowledged.

I would like to thank my parents, John and Melanie Tappen, and my mother-in-law, Grace Chan, for their support and encouragement.

Most importantly, I would like to thank my dear wife, Joy, and my little son, John, for all of their love, patience, and support over the last two years. I will always cherish the interruptions caused by an insistent 18-month old walking up, with book in hand, and loudly demanding, "Dadoo, READ!"
## Contents

1 Introduction  
1.1 Shading and Reflectance .................................. 16  
1.2 Recovering Shading and Reflectance Images ............... 18  
1.3 Overview of System and Thesis Outline .................... 20

2 Previous Work  
2.1 Generative Approaches .................................... 21  
2.1.1 A Workshop Metaphor for Analysis by Synthesis ........ 22  
2.1.2 Combining Local Evidence with Global Analysis ........ 23  
2.1.3 Reconstructing the Surface from Scene Patches .......... 24  
2.2 Discriminative Approaches ................................ 26  
2.2.1 Retinex ................................................. 27  
2.2.2 Using Color to Find Shading ............................ 29  
2.2.3 Separating Shape and Illumination Changes in Real Images .......................... 30  
2.2.4 Using Multiple Images ................................. 31  
2.3 Generative Versus Discriminative Approaches .............. 32

3 Classifying Derivatives  
3.1 Classifying with Gray-scale Information .................. 33  
3.1.1 Learning the Threshold ................................ 34  
3.1.2 Retinex Performance ................................... 36  
3.2 Learning a More Advanced Gray-Scale Classifier .......... 38  
3.2.1 Choosing Features .................................... 38
3.2.2 Learning a Classifier ........................................ 40
3.2.3 Examining the Filters Learned .............................. 42
3.2.4 Results Using the AdaBoost Classifier .................... 44
3.3 Using Color Information .................................... 46
3.4 Combining Color and Gray-Scale Information .............. 47

4 Propagating Evidence ........................................ 51
4.1 Propagating Information .................................... 52
4.2 Probability Model for Propagating Information ............ 54
4.3 Learning the Compatibility Functions ....................... 56
4.4 Inferring the Correct Labelling ............................. 57

5 Recovering Shading and Reflectance Images .................. 59
5.1 Recovering the Images ....................................... 60
5.2 Results ....................................................... 60
5.2.1 Computational Details .................................... 61
5.2.2 Results on Painted Sphere ................................. 61
5.2.3 Results on Graffiti Image ................................. 62
5.2.4 Results on Pillow Image .................................. 64
5.2.5 Results on Vegetable Image ............................... 67
5.2.6 Results on Image of a Girl ............................... 70
5.2.7 Results on Sheet Image .................................... 72
5.2.8 Results on Toy Image ..................................... 74
5.2.9 Results on Raft Image ..................................... 76

6 Conclusion ..................................................... 79
6.1 Contributions ................................................ 79
6.1.1 Classifying Binary Derivatives ............................ 80
6.1.2 The Use of Multiple Cues ................................. 80
6.1.3 Probabilistic Framework ................................. 80
6.1.4 A Learning Based Classifier ............................. 80
6.2 Future Work ................................................................. 81
   6.2.1 Estimating Derivatives .......................................... 81
   6.2.2 Improved Color Classifier ...................................... 81
   6.2.3 Improved Training Set .......................................... 81
List of Figures

1-1 A wall which has been painted by spray-paint. It is trivial for humans to distinguish the shading of the surface from the paint, but difficult for computers. ................................. 15
1-2 An image of a painted sphere, along with its shading and reflectance images 16
1-3 The derivatives of the painted sphere from Figure 1-2(b) ............................... 19
1-4 Estimates of the derivatives of the reflectance image found by classification. 19

2-1 An example of the analysis by synthesis approach. ................................. 23
2-2 A painted polyhedron similar to those used in [18] ................................. 24
2-3 An example of using image changes to recover intrinsic images. ............. 26
2-4 An image from the Mondrian world, along with the reflectance and illumination patterns used to create it. ................................. 28
2-5 A comparison of lines .................................................. 28

3-1 Examples of shading from the training set ............................................. 34
3-2 Examples of reflectance changes from the training set .......................... 35
3-3 The results of applying the Retinex classifier to the sphere from Chapter 1. 36
3-4 The results of applying the Retinex classifier to a real image of a pillow. 37
3-5 The results of applying the Retinex classifier to a real image of a painted wall. .................................................. 37
3-6 An example of the filters found by AdaBoost for classifying vertical derivatives when the illumination is from the top of the image. 43
3-7 The results of applying the AdaBoost classifier to the sphere from Chapter 1. 44
3-8 Comparison of the outputs of the Retinex and AdaBoost classifiers on the
vertical derivatives of the graffitti image. .......................... 45
3-9 Comparison of the outputs of the Retinex and AdaBoost classifiers on the
vertical derivatives of the pillow image. .......................... 46
3-10 The results of applying the color classifier to the pillow image. .......... 47
3-11 The classifications of the vertical derivatives of the pillow image obtained
from the four classifiers discussed. .......................... 50

4-1 Comparison of the mouth from the pillow and an ellipse from the training
set. .................................................. 51
4-2 The classification of ambiguous areas of the pillow image. ............. 52
4-3 A simple image of an edge for demonstrating how to propagate information. 53
4-4 An example of using the joint probability distribution of the derivatives to
fill in ambiguous derivatives. .................................. 54
4-5 The derivative image with variables and lines relating the dependencies
between variables. .................................. 55
4-6 The angles used in calculating the compatibility of two nodes in the Markov
Random Field. .................................. 56
4-7 Comparison of results using Belief Propagation and Generalized Belief
Propagation .................................. 58

5-1 Results on Painted Sphere .................................. 62
5-2 Results on graffitti image using the gray-scale classifier. ............. 62
5-3 Results on graffitti image using the gray-scale classifier. ............. 63
5-4 Results on the pillow image from Chapter 3 using color and gray-scale
information combined. .................................. 64
5-5 Results on the pillow image from Chapter 3 using only gray-scale informa-
tion. .................................. 65
5-6 Results on the pillow image from Chapter 3 using only color information. 65
5-7 Results on an image of vegetables using color and gray-scale information
combined. .................................. 67
5-8 Results on an image of vegetables using only gray-scale information.  68
5-9 Results on an image of vegetables using only color information.  68
5-10 Results on an image of a girl using color and gray-scale information combined.  70
5-11 Results on an image of a girl using only gray-scale information.  71
5-12 Results on an image of a girl using only color information.  71
5-13 Results on an image of a bed sheet using color and gray-scale information combined.  72
5-14 Results on an image of a bed sheet using only gray-scale information.  73
5-15 Results on an image of a bed sheet using only color information.  73
5-16 Results on an image of a toy using color and gray-scale information combined.  74
5-17 Results on an image of a toy using only gray-scale information.  75
5-18 Results on an image of a toy using only color information.  75
5-19 Results on an image of a raft using color and gray-scale information combined.  76
5-20 Results on an image of a raft using only gray-scale information.  77
5-21 Results on an image of a raft using only color information.  77
Chapter 1

Introduction

Human beings possess the remarkable ability to interpret the scene in an image. Viewing the image in Figure 1-1, it seems clear that the image is a picture of wall that has been painted. Making this judgment requires that the viewer understand that the large black lines are not caused by the shape of the wall, but by the spray paint. The viewer effectively divides the patterns in the image into two distinct visual processes with one process representing the surface of the wall and one representing the paint.

The image in Figure 1-1 could also be interpreted in terms of the visual characteristics at each point in the scene. The three pertinent characteristics in Figure 1-1 are the illumination of each point, the orientation of the surface at each point, and how each point reflects light. With these three characteristics of the scene separated, many useful tasks would be considerably easier. The actual three-dimensional surface of the wall could be recovered.

Figure 1-1: It is trivial for humans to distinguish the shading of the surface from the paint, but difficult for computers.
Figure 1-2: An image of a painted sphere, along with its shading and reflectance images from the orientation and illumination at each point. The paint could be segmented from the rest of image by finding the locations where there are changes in how the surface reflects light.

While it would be easy for humans to interpret Figure 1-1 in terms of the characteristics of the scene, it is difficult for computers. Understanding the image requires strong prior knowledge of the statistics of surfaces and how they appear when illuminated. In this thesis, we present a system that is able to recover two specific characteristics, shading and reflectance, of a scene from a single image. The system relies on both color and grayscale information to differentiate between the effects shading and reflectance on the image, then reconstructs the image as it would appear with the effects of one of the characteristics removed.

1.1 Shading and Reflectance

Our system is interested in two specific intrinsic characteristics of the scene, the shading and reflectance of every point in the scene. We use the term shading to denote the interaction of the shape of the surface and the illumination of the scene. Figure 1-2(a) shows an example of an image containing only shading. The image is of a sphere resting on a flat surface and every change in the image is caused by shading. Areas where the surface
normal of the sphere points toward the illumination are bright and areas where the surface normal points away from the illumination are dark.

Now imagine modifying this sphere by painting three gray dots on the top of the sphere. The sphere’s new appearance is shown in Figure 1-2(b). There are now two kinds of changes in the image: changes due to shading, manifested all over the sphere, and changes due to paint, manifested around the dots. The paint changes the sphere’s appearance because it reduces the amount of light reflected by the sphere. The characteristics of how a surface reflects light are known as the surface’s reflectance. The light to dark transition around the border of each dot is caused because the reflectance of the sphere changes at the border of the dots. The reflectance image of the scene in Figure 1-2(b) can be created by only rendering the reflectance at each point and ignoring the shading of the sphere. The reflectance image for the painted sphere is shown in Figure 1-2(c).

The goal of our system is to decompose an image, such as Figure 1-2(b), and recover the shading image, Figure 1-2(a), and the reflectance image, Figure 1-2(c). The reflectance image can be thought of as being created by painting the scene, but only including the color of each point on the object. The shading image can be viewed as being created by rendering the scene as if every surface in the scene is made of a single material, such as plaster.

The ability to decompose an image into shading and reflectance images is valuable because most computer vision tasks require that the two be treated separately. Systems interested in the shape of a surface must be able to ignore any reflectance changes that occur. Likewise, segmenting an image is often best accomplished by finding areas where it is likely a reflectance change occurs. The ability to filter out the effects of unwanted characteristics of the scene would greatly simplify these tasks.

Images created by isolating a specific characteristic of the scene are known as intrinsic images because each image contains a single intrinsic characteristic of the scene [2]. The shading image recovered in our system differs slightly from the notion of intrinsic images first presented by Barrow and Tenenbaum. The set of intrinsic images first proposed included an image for each physical characteristic of the scene that could be measured. In this representation, a separate intrinsic image is used to describe each source of illumination in the scene. The orientation of each point in the scene is also explicitly represented.
The shading image recovered by our system is the combination of the illumination and orientation of every point in the scene. While recovering intrinsic images representing the illumination of the scene and orientation of points in the scene would describe the scene more accurately, treating illumination and orientation separately greatly increases the number of parameters in the scene that must be estimated. This led us to treat illumination and orientation of the surfaces in the scene together by recovering shading instead.

1.2 Recovering Shading and Reflectance Images

Our system recovers the shading image by isolating the reflectance changes in the image, then re-rendering the image with the reflectance changes removed. Similarly, the reflectance image is found by removing all of the changes in the image caused by shading. To remove image changes caused by shading or reflectance changes, we assume that the input image, \( I(x, y) \), is the product of the shading image, \( S(x, y) \), and the reflectance image, \( R(x, y) \):

\[
I(x, y) = S(x, y) \times R(x, y)
\]  

Considering the images in the log domain instead, the input image becomes the sum of the shading and reflectance images. Any shading or reflectance changes will cause image derivatives in \( S(x, y) \) and \( R(x, y) \), respectively. Computing derivatives is a linear operation, so the derivatives of the input image are the sum of the derivatives of the shading image and the derivatives of the reflectance image. It is unlikely that significant shading boundaries and reflectance edges occur at the same point, so we make the simplifying assumption that every image derivative is either caused by shading or reflectance. This reduces the problem of specifying the shading and reflectance derivatives to that of binary classification of the image's \( x \) and \( y \) derivatives. Using binary classification allows us to focus on discriminating between image changes caused by shading and reflectance changes rather than trying to model the scene in the image. The advantages of discriminative approaches are discussed in Section 2.3.

18
Figure 1-3: The derivatives of the painted sphere from Figure 1-2(b).

Figure 1-4: Estimates of the derivatives of the reflectance image found by classification.

This process can also be viewed as using binary classification to estimate the derivatives of the shading and reflectance images. Figure 1-3 shows the horizontal and vertical derivatives of the painted sphere from Figure 1-2(b). In order to recover an estimate of the derivatives of the reflectance image, we set all derivatives not caused by a change in the reflectance image to zero. The leads to the estimated derivatives of the reflectance image shown in Figure 1-4.

After classifying the image derivatives, the shading image can be reconstructed by setting every derivative classified as being caused by a reflectance change to zero, then reconstructing the image from the derivatives classified as shading. Reconstructing the images is discussed in Chapter 5. The reflectance image can be recovered in a similar fashion.

This method of classifying derivatives, then reconstructing the desired image is sim-
ilar to that used in [4], which was developed independently of this work. In that work, Finlayson, et al. removed shadows from images by eliminating derivatives thought to be caused by shadows, then reconstructing the image from the modified derivative image.

1.3 Overview of System and Thesis Outline

There are three major steps in the system to be described in this thesis:

1. Compute the horizontal and vertical derivatives of the image.

2. Classify each derivative as being caused by shading or a reflectance change.

3. To reconstruct the reflectance image, set all derivatives not caused by a reflectance change to 0. This forms an estimate of the horizontal and vertical derivatives of the reflectance image. From this estimate, recover the reflectance image. Recover the shading image in a similar fashion.

In this thesis, Chapter 2 reviews previous work related to recovering shading and reflectance images.

Chapters 3 and 4 focus on step 2, classifying the image derivatives. Chapter 3 discusses the design of the classifiers used to separate derivatives caused by shading from those caused by reflectance changes. Classifiers using both color and gray-scale information are discussed. In addition, a method for combining both color and gray-scale information is presented.

Chapter 4 discusses how to handle areas where the image information is ambiguous. A method for propagating information from areas of the image with a clear classification into ambiguous areas of the image is presented.

Chapter 5 explains how the shading and reflectance images are recovered from the derivative images. In addition, it presents results on real images.

Chapter 6 reviews the contributions of this thesis and suggests topics for further research.
Chapter 2

Previous Work

Much of the previous work on decomposing an image into shading and reflectance images is based on the study of lightness perception. Under certain conditions, the human visual system attempts to remove the effects of illumination in order to accurately judge the reflectance of a surface. This is essentially the same task as finding the shading and reflectance of the scene.

The previous approaches described here can be roughly divided into generative and discriminative approaches. The primary difference between the two lies in how the world is modeled. A generative approach uses a model that describes the likelihood of the possible scenes that could have created the image. Discriminative approaches only model the world enough to find features that distinguish shading from reflectance changes.

2.1 Generative Approaches

A generative approach analyzes an image by finding the scene that most likely produced the image. In this thesis, scene of an image refers to the surface shown in the image and the reflectance of every surface. The unique feature of generative approaches is that the world is explicitly modelled through two probability distributions. First, a generative approach requires a distribution specifying the probability of the possible scenes. We denote this as \( P(S) \), where \( S \) represents the a possible scene. To find the scene that most likely produced some image, denoted as \( I \), a generative approach also requires a conditional distribution
specifying the probability that the scene $S$ will produce the image $I$. This is denoted as $P(I|S)$. Using Bayes’ Rule, $P(I|S)$, the probability that scene $S$ caused the image $I$, can be calculated for every candidate scene. The best surface and reflectance pattern will be the scene that maximizes $P(I|S)$. It is important to note that in addition to $P(S)$ and $P(S|I)$, generative approaches require some method of generating candidate values of $S$, the scene being recovered.

Generative approaches can also be posed in terms of cost functions. Instead of finding the scene with the highest probability, the goal is to find the surface and reflectance pattern with the smallest cost. Similar to probabilistic methods, there are two cost functions, one assigning a cost to each candidate scene and one assigning a cost to each scene based on how well it describes the image. Again, some method of generating candidate values of $S$ is required. It is important to note that probabilistic systems and systems based on cost functions are roughly equivalent. Probabilities can be rewritten as costs and vice versa.

Figure 2-1 illustrates the generative strategy when used to find the shading and reflectance images from an image of a painted sphere similar to that used in Chapter 1. First, a model of the world is used to generate possible shapes of the surface shown in the input image. The candidate shapes are shown in the right panel of Figure 2-1. Each possible shape has an associated reflectance pattern, which can be combined with a rendered image of the surface to produce the input image. The model of the world is then used to choose the most likely scene from the candidate scenes found. The reflectance pattern can have any appearance, so the likelihood of the scene will be largely based on the probability of the surface’s shape. In the example in Figure 2-1, suppose that the most likely surface is the half-sphere. Once the most likely shape and reflectance pattern has been found, the shading intrinsic image is produced by rendering the surface. The reflectance image can then be computed by dividing the shading image from the input image.

### 2.1.1 A Workshop Metaphor for Analysis by Synthesis

In [1], Adelson and Pentland use a “workshop metaphor” to describe a model of a world consisting of images of planar surfaces. This metaphor describes the scene as being con-
structured from sheet metal that can be painted and shaped. In addition, lights can be placed at arbitrary locations. The scenes are created by three specialists: a lighting specialist, which places lights in the scene, sheet metal worker, which changes the shape of the surface in the scene, and a painter, who changes the reflectance of the surface. Given an image, each specialist tries to recreate the scene using as little assistance as possible from the other specialists. This scenario also includes a supervisor, who is able to use all three specialists.

The different possible scenes are evaluated according to the cost of creating each scene. Each action performed by a specialist to recreate the scene has a cost assigned to it, with more complicated actions having a higher cost. For instance, painting a rectangle is $5, while painting a general polygon is $5 per side. The most likely scene is computed by finding the scene with the lowest cost.

2.1.2 Combining Local Evidence with Global Analysis

Later, Sinha and Adelson [18] improved results on images of planar surfaces by using a combination of global and local analysis. The objects in the input image are limited to polyhedra which have their faces painted different shades of gray. For these surfaces, every shading or reflectance change is manifested as a line in the image. Junctions of lines will occur at each vertex of the polyhedron and at some areas with reflectance changes. Figure 2-2(a) shows an example of a painted polyhedron with junctions caused by reflectance and
shading marked.

The algorithm begins by labelling each junction as either being caused by a reflectance change or shading. Shading creates specific types of junctions, so Sinha and Adelson were able to create a catalogue that indicates the cause of the different types of junctions. However, local junction analysis is not enough. The authors point out that the junctions of the polyhedron in Figure 2-2(b) will be classified as illumination edges. However, examining the whole image shows that this is impossible.

To overcome the limitations of local analysis, the next step is to find the most likely shape and lighting of the scene using global analysis. The authors find a set of perceptually likely shapes consistent with the input pattern, then verify that the rendered shapes could reproduce the input pattern. The final output is the most likely shape, the associated reflectance pattern, and the possible directions of the light source.

### 2.1.3 Reconstructing the Surface from Scene Patches

Instead of explicitly modeling the surfaces in the scene, Freeman et al. [5] proposed constructing the shading image from a mosaic of image patches taken from example images. The shading image is reconstructed by dividing the input image into patches, then for each image patch, choosing the shading and reflectance patches that best match the image patch.
These patches are mosaiced together to form the shading and reflectance images. The algorithm considers a discrete set of candidate sets of patches for each image patch. The probability of each candidate set of patches is influenced by two factors: how well the patches explain the image and how compatible the patch is with neighboring patches. In addition, there is a penalty based on the flatness of the shading patch applied to the log probability of each candidate. This is important because any image can be explained as a reflectance pattern painted onto a flat surface.

Considering compatibility of neighboring patches is necessary to create smooth shading image. The compatibility between two neighboring patches is measured by examining how well the borders of the patches match. The patches are spaced such that each patch overlaps the other by a small amount. If two neighboring patches are compatible, then their appearance should be similar in the overlapping area. If the neighboring patches are at indices $j$ and $k$, $d_j$ is the overlapping area in patch $j$, and $d_k$ is the corresponding overlapping area in patch $k$, the compatibility between the two patches, $\Psi(x_j, x_k)$ is

$$\Psi(x_j, x_k) = e^{-\left(\sum |d_k - d_j|^2\right)}$$  \hspace{1cm} (2.1)

The algorithm operates by first finding a set of candidate shading patches for each image patch and calculating the probability of each patch explaining the input image patch. The algorithm then evaluates the compatibility between every possible combination of neighboring patches. With this information, the algorithm can now find the most likely explanation for the whole image by formulating the problem as a Markov Random Field. Each node in the MRF represents the shading patch that best fits a patch of input image. The interpretation of each image patch is Markov in that a patch’s likelihood only depends on the neighboring patches. If $x_j$ is the shape and reflectance explanation at patch $j$, $x_k$ is the most likely explanation at patch $k$, $y_i$ is the input image information at patch $i$, and $\Phi(x_i, y_i)$ is the compatibility between the scene and reflectance patch at location $i$ and the image at location $i$, then the probability of the explanation of the whole image is

$$P = \prod_i \Phi(x_i, y_i) \prod_{(j,k)} \Psi(x_j, x_k)$$  \hspace{1cm} (2.2)
Maximizing this equation for all $x_i$ has been shown to be NP-complete. This makes it intractable for large images. Freeman et al. solve this by using an approximate inference techniques to find a likely explanation for the image. While it is not guaranteed to find the globally optimal explanation, it often finds a good approximation.

2.2 Discriminative Approaches

The alternate strategy for finding shading and reflectance images is to distinguish effects of shading and reflectance in the image, then eliminate one or the other. Figure 2-3 illustrates this strategy. Given the image of the sphere, the algorithm first identifies the changes in the image due to shading and those due to a reflectance change. In this case, the gradient from the center to the edge of the sphere is attributed to the shape of the surface and the line running down the middle of the sphere is attributed to a change in reflectance. The shape image can then be recovered by rendering the image without the line down the middle.

There are three key aspects to discriminative approaches. The first is the type of transformation used to isolate image changes. An obvious choice is the derivative operation because every change in the image will cause a non-zero derivative at that point. Other possible transformations include wavelet transforms or the output of an edge-detector.

The second aspect is a mechanism for classifying the changes. The classifier analyzes
the intensity patterns in the image to decide which of the changes identified in the first step were caused by the shape of the object and which were caused by a change in how the surface reflects light. In order to accomplish this, the classification mechanism must have some knowledge of the statistics of surfaces in the world in order to separate the shape and reflectance changes. Typically, the classifier and the image transformation used to identify changes are chosen together in order to make classification as simple as possible.

The third component is a mechanism for using the output of the classifier to recover the intrinsic images. This mechanism must be able to use the input image and the classified image changes to return the image as if it had a single uniform reflectance. The relative reflectance at every point can then be estimated using the input image and the shape image.

2.2.1 Retinex

The first algorithm to attempt to recover the reflectance of a scene was the Retinex algorithm of Land and McCann[13]. The Retinex algorithm was originally designed operate in the world of illuminated Mondrian images. Figure 2-4(a) shows an example of a Mondrian image. The Mondrian world consists of images of a flat surface covered with patches of varying reflectance. Figure 2-4(b) shows an example of such a surface. In addition, the surface is lit with a slowly varying illumination, such as that shown in Figure 2-4(c). Mondrian images were chosen because their simplicity made distinguishing between reflectance and illumination changes much simpler.

This simple model of the world makes designing the classifier simple. Figure 2-5 shows the logarithmic image intensities of the illumination and reflectance images along one row of the image in Figure 2-4(a). The reflectance intensities are marked by abrupt, strong changes where one patch overlaps another. On the other hand, the shape intensities grow in a slow, monotonic fashion. Examining the derivative of the two lines makes the difference between the statistics of shape and reflectance even more obvious. In Figure 2-5(b), the derivatives of the shape image are all near zero, while changes in the reflectance intensities are marked by strong derivatives.

The different magnitudes of shape and reflectance derivatives allows shape and re-
Figure 2-4: An image from the Mondrian world, along with the reflectance and illumination patterns used to create it.

Figure 2-5: A comparison of lines
reflectance changes to be separated using a simple threshold test. Each change in the image will have a corresponding non-zero derivative associated with it. Derivatives in the image with an absolute value above some threshold are most likely caused by a change in the reflectance image and small derivatives indicate an illumination change.

The next step is to determine how to use the classifier to recover the shape and illumination images. The algorithm is operating on logarithmic images, so the input image is the sum of the shape and reflectance images. The shape image can be recovered from the derivative of the input image by subtracting the derivative of the reflectance image and integrating the result. Since the derivatives of the shape image are always near zero, this can be approximated by simply setting any derivatives caused by reflectance changes to zero, then integrating to obtain the log shape image.

This process for separating illumination and reflectance images only works along one dimension. Land and McCann extended it to two-dimensional images by choosing many random lines from the image, then running the one-dimensional Retinex process on each line. If enough lines are processed, then there will be enough data to interpret the image.

Horn [12] later extended the Retinex model to work more naturally on images. Because edges can occur with any orientation, Horn chose to use a rotationally symmetric operator. In particular, he used the Laplacian operator:

\[ \nabla^2 I = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2} \]  

(2.3)

because the Laplacian is a rotationally symmetric operator. The classification is Horn's scheme is the same as in Retinex, large coefficients are due to a change in reflectance and small coefficients are due to a change in shape. The intrinsic shape image can be recovered by setting any coefficients due to reflectance changes to zero and inverting the transform.

2.2.2 Using Color to Find Shading

Instead of thresholding based on the intensity of the gradient, Funt et al. [9] proposed thresholding based on color information in the image. Color is used to find reflectance changes, but the output of the algorithm is a gray-scale image that represents the shading
image of the color image if it were transformed into a gray-scale image. The shading image is found in a way similar to that used by Horn. Given a gray-scale input image, I, the algorithm computes x and y derivatives of I. The color image is then used to threshold away derivatives due to a change in reflectance. These changes are found by computing a two-dimensional chromaticity value at each pixel in the color image. The thresholding decision about a gray-scale derivative is made by examining the derivatives of the two chromaticity images. If the derivative in either chromaticity image is large, then the gray-scale derivatives are thresholded away as a reflectance derivative. The shading image is then found differentiating the thresholded x and y derivative images a second time and adding the images to create the Laplacian of the shading image. This can then be inverted in a method similar to Horn’s.

2.2.3 Separating Shape and Illumination Changes in Real Images

Retinex is successful in the limited domain of images it was designed for because the statistics of reflectance and illumination are simple and quite different in Mondrian images. Unfortunately, these simple statistics do not generally hold for the surfaces and reflectance patterns that make up real images. An algorithm designed to operate on real images must take into account that the shape of the surface will likely not be flat, leading to image derivatives as strong as those caused by reflectance changes.

However, the key idea behind Retinex, that illumination and reflectance changes typically have different statistics, still holds when the surface is not flat. As explained in Section 2.1.2, Sinha and Adelson were able to use the patterns of each junction to decide whether they were caused by a reflectance change or shading. Freeman and Viola also considered more complex surfaces and found that a simple prior computed from image intensities could predict whether a human viewed the image as a picture of an illuminated surface or as pattern that had been painted [6].

These two results motivated Bell and Freeman to attempt to use the local image information around a change in the image to classify the cause of that change [3]. Using machine learning techniques, they attempted to learn image patterns that could be used
to label multi-resolution wavelet coefficients as being caused by a reflectance change or shading. To identify image patterns, Bell and Freeman used a classifier based on cascaded, non-linear filters[14]. Cascaded, non-linear filters are a series of linear filters, where the output of each filter undergoes non-linear transformation and is filtered by the next filter in the cascade. [For more information on cascades of non-linear filters, see section 3.2.1] After choosing a set of useful filters, their algorithm uses a mixture of factors analyzer [10] to fit a probability distribution to the outputs of the filters when applied to image patches surrounding illumination changes. The algorithm then fits another distribution to the filters' outputs on patches surrounding reflectance changes.

Each wavelet coefficient is then classified by first computing the response of the chosen filters on the image patch surrounding the change. Using the distributions of the filter responses to shading and reflectance changes, the classification is then found using likelihood ratio test.

2.2.4 Using Multiple Images

Weiss [20] took a different approach by using multiple images to find the reflectance image. Weiss' method requires multiple images of a fixed scene and changing illumination. The output of a webcam fixed on an outdoor scene is a good example of this type of input. As the sun moves through the sky, the illumination of the scene changes, while the scene itself remains relatively constant.

The true reflectance of the scene is recovered by calculating the image derivatives in the horizontal and vertical directions for each image, then estimating the image derivatives for the reflectance of the scene. By assuming that the distribution of the derivatives of the illumination image are sparse, Weiss shows that each derivative of the reflectance image is optimally recovered from the derivatives of the input images by computing the median of the derivatives.

Once the horizontal and vertical derivatives of the reflectance image have been computed, the reflectance image is recovered by finding the pseudo-inverse of the linear system represented by the image derivatives. The illumination of each of the input images can then
be found by dividing out the reflectance image.

2.3 Generative Versus Discriminative Approaches

The primary weakness of generative approaches is that a model of how scenes and images are created must be constructed. Creating a model complex enough model to recover meaningful results on real images is prohibitive. In addition, as the model of the world becomes more complex, it becomes more difficult to generate candidate scenes that match the image.

On the other hand, a discriminative approach only requires some feature that is able to distinguish between shading and reflectance changes. This feature will likely be much simpler to compute than finding the scene that the image represents. However, some knowledge of the world is still necessary in order to find the best feature.

In order to produce a system that could generalize to real images, we chose to take a discriminative approach. This enabled us to avoid issues such as the best strategy for searching through the space of possible scenes or trying to deduce the geometry of the scene from a single image.
Chapter 3

Classifying Derivatives

Our system recovers the shading and reflectance images by estimating their derivatives, then recovering the images from the derivatives. As explained in Chapter 1, we estimate the derivatives by classifying each horizontal and vertical derivative as being caused by shading or a reflectance change. This makes the quality of the recovered shading and reflectance images dependent on the quality of the classifications. Classifying the image derivatives is the most important step for recovering the shading and reflectance images.

This chapter discusses how to classify derivatives using two types of information, color and gray-scale. The gray-scale classifier is explained in Sections 3.1 and 3.2; beginning with a simple heuristic in Section 3.1. A more general classifier is then developed using machine learning techniques in Section 3.2. We then develop a second classifier which relies on color information to distinguish between shading and reflectance changes. Finally, Section 3.4 discusses how to combine the results of the two classifiers.

3.1 Classifying with Gray-scale Information

We begin by evaluating one of the most well known heuristics for classifying image derivatives, Retinex. Originally designed to separate reflectance changes from the effects of varying illumination, Retinex classifies an image derivative based on its magnitude. Derivatives with large magnitudes are labelled as reflectance changes and those with small magnitudes are classified as being caused by illumination. While originally designed to operate on
Mondrian images, we evaluated how well this classification heuristic generalizes to real images.

### 3.1.1 Learning the Threshold

Before using Retinex, the threshold that divides shading and reflectance changes must be chosen. Ideally, this threshold would be found by finding the threshold that best classifies the derivatives in a set of real images where each derivative is labelled. Unfortunately, creating a set of real images that have been correctly labelled is a very difficult task. Instead, we learn the threshold from a set of synthetic images. The Retinex threshold is found by setting it to the value that correctly classifies the largest percentage of the training set. This same training set will also be used to train the more advanced classifier in Section 3.2.

In order to generate a training set that captures the statistics of shading, the examples of shading are created with three methods. Approximately 63% of the set is generated by rendering random ellipsoids with Lambertian shading. The surfaces are created by randomly placing ellipsoids throughout the image. At points where ellipsoids overlap, the maximum of the height of every ellipsoid at that point is used. Before being rendered, the surface was smoothed with a Gaussian filter. An example image is shown in Figure 3-1(a).

The remaining 37% of the examples of shading was taken from the set created by Bell.
and Freeman for their work on producing intrinsic images. Half of the Bell and Freeman training set was also created by randomly placing ellipses in the image, except the ellipses are rendered using linear shading, an approximation to the true Lambertian shading when the angle of the illumination is oblique to the surface [16]. An example of this part of the training set is shown in Figure 3-1(b). The rest of the Bell and Freeman training set comes from rendered fractal surfaces. Each example image is created by using the mid-point displacement method to create a surface, then rendering it using linear shading.

Every shading image was lit from the same direction. In the training set, the illumination comes from the top of the image. We assume that for every input image, the direction of illumination is roughly known. When classifying an input image, we rotate the image so that the dominant illumination in the image also comes from the top side of the image.

The examples of reflectance changes were generated in two fashions. Approximately 37% of the reflectance examples were generated by randomly placing ellipses in the image, then rendering them as reflectance changes. Figure 3-2(a) shows an example of an image produced in this fashion. The remaining 63% of the training set is generated by randomly rendering lines. In order to create images with corners and anti-aliased lines, we use ellipsoid images similar to those used as shading examples. To create the lines, we mask the darkest portions of the ellipsoid image, then set the rest of the image to white. The intensity of each line is then set randomly. An examples of an image of rendered lines is shown in

Figure 3-2: Examples of reflectance changes from the training set.
Derivatives caused by reflectance changes are marked in white. (a) The input image (b) The classification of each vertical derivative.

3.1.2 Retinex Performance

When trained to classify vertical derivatives, Retinex performed poorly on the training set, classifying only 60% correctly.

Figure 3-3 shows the results on the painted sphere from Chapter 1. On this image, Retinex performs well, the three painted spots are marked as reflectance changes and the rest of the sphere is labelled as shading, except for the border. The strength of the gradient also causes Retinex to mistakenly label the borders of the sphere as reflectance changes also. In this scene, the sphere has the same reflectance as the surface it is resting on, so the border should be classified as a shading change. In general, occluding edges are not considered in our model, so derivatives around occluding edges may have an unclear classification.

Figures 3-4, and 3-5 show the results applied to two real images. In Figure 3-4(b), the vertical derivatives of the pillow image shown in 3-4(a) are marked in white. The Retinex classifier locates the reflectance changes in the image quite well, all of the deriva-
Figure 3-4: The results of applying the Retinex classifier to a real image of a pillow. Derivatives caused by reflectance changes are marked in white. (a) The input image (b) The classification of each vertical derivative.

Figure 3-5: The results of applying the Retinex classifier to a real image of a painted wall. Derivatives caused by reflectance changes are marked in white. (a) The input image (b) The classification of each vertical derivative.

tives around the face on the pillow are marked as reflectance changes. However, Retinex is also prone to misclassifying derivatives caused by shading as reflectance changes. This is most noticeable around the ridge in the upper left corner of the pillow and the ripples on the left side of the pillow.

This tendency to mistake shading for reflectance changes is even more noticeable in Figure 3-5. Here, the variations on the rock surface cause many false positives because the roughness of the surface causes many of the image derivatives to have relatively large magnitudes.
3.2 Learning a More Advanced Gray-Scale Classifier

Retinex, extended beyond the Mondrian world for which it was designed, is limited by its reliance on a very simple statistic and by not considering local information. The image of the wall in Figure 3-5 is an excellent example of these limitations. At a single point in the image, the roughness of the wall is able to cause a derivative with the same magnitude as the derivatives caused by the paint. However, examining an area around each derivative makes it clear whether there is paint present or the derivative is caused by the natural variations in the surface of the wall. In this section, we show how to find a more advanced gray-scale classifier.

The limitations of Retinex are due to it being designed for a simplified world. Focusing on the simplified world of Mondrian images made choosing the classifier relatively simple. However, choosing the sufficient statistics of shading and reflectance changes for more general types of images is difficult. In order to build a more general classifier, we use machine learning techniques to isolate the patterns indicative of shading and reflectance changes. If these statistics can be isolated, then they can be used to separate derivatives caused by a reflectance change from those caused by shading.

3.2.1 Choosing Features

In order to learn the statistics that distinguish shading from reflectance changes, a set of features must be chosen that isolate those statistics. These features can be found by first choosing a family of features, then choosing the best features from that family.

The basic feature of our classifier is the absolute value of the response of a linear filter. The output of a feature \( F \) given an input patch \( I_p \) is

\[
F = |I_p \ast w|
\]

(3.1)

where \( \ast \) is convolution and \( w \) is a linear filter. The filter, \( w \) is the same size as the image patch, \( I \), and we only consider the response at the center of \( I_p \). This makes the feature a function from a patch of image data to a scalar response. This feature could also be viewed
as the absolute value of the dot product of $I_p$ and $w$. In the rest of the text, we will refer to a feature computed in this manner as a **non-linear filter**.

We chose to use the responses of linear filters as the basis for our feature, in part, because they have been used successfully for characterizing [15] and synthesizing [11] images of textured surfaces. Another advantage of using linear filters is that once specific filters are chosen as features for the classifier, it should be relatively simple to interpret what image patterns the feature is responding to. Using the absolute value of the filter response is necessary because we are only interested in the magnitude of the response, not the sign. Changing an edge from a dark to light transition into a light to dark transition should not change its classification.

We also evaluated the features used by Freeman and Bell to classify wavelet coefficients [3]. Bell and Freeman used cascaded, non-linear filters as the basic feature for their classifier. A cascaded, non-linear filter consists of a non-linear filter, described above, with a second filtering and absolute value operation added. In addition, the output of the first filtering stage is down-sampled before it is filtered again. If $F$ is the output of the non-linear filter described in equation 3.1, then the output of a cascaded, non-linear filter, $F_c$ is

$$F_c = |F_{12} \ast w_2|$$

where $F_{12}$ denotes that $F$ is down-sampled by a factor of 2 before being convolved, and $w_2$ is a second linear filter.

After evaluating both classifiers, we found that using a cascaded, non-linear filter, rather than just a non-linear filter, did not improve the performance of the classifier. In addition, taking the absolute value between filtering steps makes it more difficult to interpret the features chosen; leading us to only use a single filtering step followed by an absolute value operation. We also found that removing the down-sampling step did not affect the classifier's performance on the training set.
3.2.2 Learning a Classifier

Training the classifier actually involves two tasks, choosing the set of features and training a classifier based on those features.

In [3], Bell and Freeman perform feature selection and train the classifier in two distinct steps. Features are chosen by selecting the linear filters for the non-linear cascade from a large set of candidate filters. The large number of possible filters, their total set included 1875 possible values for the linear filters, led Bell and Freeman to take a greedy approach to finding the best set of filters. After computing the correlation coefficient between each possible pair of filters, the pair with the highest correlation coefficient is chosen and one of the two filters from the pair is eliminated. The feature selection algorithm continues to eliminate filters in this fashion until only the desired number of filters remained.

Once the filters have been chosen, the classifier is trained by fitting probability distributions to the outputs of the filters. The distributions are found using mixture of factor analyzers [10]. A single factor analyzer models a high-dimensional jointly Gaussian random vector as a linear transformation of a much lower-dimensional jointly Gaussian random vector. Once distributions have been fit to the outputs of the filters, the wavelet coefficient can be classified using a likelihood ratio test.

Alternatively, training and feature selection can occur simultaneously. Tieu and Viola [19] accomplish this using the AdaBoost algorithm [7]. The AdaBoost algorithm provides a way to combine a collection of weak classifiers, drawn sequentially, into a single strong classifier. The AdaBoost algorithm iteratively chooses a weak classifier, calculates the weight that should be given to the weak classifier, then reweights the training set to place more emphasis on examples misclassified by the weak classifier. The output of final strong classifier is a weighted combination of the outputs of the weak classifiers. The weak classifiers used in Tieu and Viola’s method consist of a threshold test on the output of a single non-linear cascade. The weak classifier used for each iteration is chosen by finding the weak classifier that performs the best on the reweighted training set. Effectively, Tieu and Viola’s algorithm performs greedy feature selection and learns the classifier at the same time.
We chose to use a method similar to that used by Tieu and Viola because the AdaBoost algorithm is simple, efficient, and the classifiers trained using the AdaBoost algorithm have been found to generalize well to new data.

### Training Using AdaBoost

For learning purposes, all derivatives caused by shading are assigned the label 1 and derivatives caused by a reflectance change are labelled -1. In this system, each weak classifier consists of a single non-linear filter and a threshold. An example is classified by examining whether the response of filter is above or below the threshold.

At each iteration $t$, the AdaBoost algorithm maintains a distribution, $D_t$, of the training samples. $D_t(i)$ can also be thought of as the weight of training example $i$.

The first step in the algorithm is to find a weak classifier that classifies the reweighted training set with better than 50% accuracy. For iteration $t$, the goal is to find a classifier $h_t$ such that

$$
\Pr_{i \sim D_t} [h_t(x_i) = y_i] > 0.5
$$

3.3

where $x_i$ is training example $i$ and $y_i$ is its true label. The notation $i \sim D_t$ denotes that $i$ is chosen according to the probability distribution $D_t$.

When choosing a classifier, we actually choose the classifier that has the highest probability of being correct on the weighted training set. Because each weak classifier consists of a single non-linear filter and a threshold, greedily choosing the weak classifier is equivalent to greedily choosing the best filters.

The next step is to calculate the weight to assign to the new classifier. This weight, $\alpha_t$, is

$$
\alpha_t = \frac{1}{2} \ln \left( \frac{1 - \epsilon_t}{\epsilon_t} \right)
$$

3.4

where $\epsilon_t$ is the probability of error, $\Pr_{i \sim D_t} [h_t(x_i) \neq y_i]$, when the training examples are drawn with probability distribution $D_t$. Essentially, the errors are weighted according to $D_t$.

Once $\alpha_t$ has been computed, the weight for each training example in the next iteration,
\( D_{t+1} \) is computed as

\[
D_{t+1}(i) = \frac{D_{t}(i) \exp(\alpha_{t} y_{t} h_{t}(x_{i}))}{Z_{t}}
\]  

(3.5)

where \( Z_{t} \) is a normalization factor needed to make \( D_{t} \) a valid distribution.

Once the desired number of iterations has run, a new sample is classified by having each of the weak classifiers independently classify the sample. Given a sample, \( x \), each weak classifier casts a vote to classify the sample as -1 or 1. The final classification output of the combined classifier, \( H(x) \) is the weighted average of the votes

\[
H(x) = \text{sign} \left( \sum_{t=1}^{N} \alpha_{t} h_{t}(x) \right)
\]  

(3.6)

where \( N \) is the number of weak classifiers used.

Filters used for Training

To maximize the number of filters evaluated as possible weak classifiers, each candidate filter was formed from the combination of two smaller filters taken from a set of 9 × 9 filters. This set consists of nine derivative of Gaussian filters and nine second-derivative of Gaussian filters oriented every 22.5 degrees between 0 and 180 degrees. In addition, the set included an impulse filter and four Gaussian filters with different widths.

The actual candidate set of filters that the classifiers are chosen from consists of every possible combination of the smaller filters, except for combinations without at least one derivative filter.

3.2.3 Examining the Filters Learned

We used the training set described in Section 3.1.1 to learn the filters and thresholds for the AdaBoost classifier. This training set was also used to find the threshold for the Retinex classifier. The size of the linear filter in each weak classifier is 17 × 17.

Figure 3-6 shows the filter associated with each weak classifiers chosen by the AdaBoost algorithm. Each column contains the filter for one weak classifier. The classifiers
Figure 3-6: An example of the filters selected by AdaBoost for classifying vertical derivatives when the illumination is from the top of the image. Each column contains the filters for one cascade.

are trained for vertical derivatives when the illumination is from the top of the image. For all but the final filter, a response with a large magnitude signals a reflectance change.

The percentage of weight allocated to each classifier is shown beneath each weak classifier in Figure 3-6. The AdaBoost classifier operates by using a voting scheme to combine the results of each of weak classifiers in Figure 3-6. Given an input patch, each weak classifiers checks whether the absolute value of the response of its filter is above a certain threshold. For all but the final weak classifier, if the response is above the weak classifier's threshold, the weak classifier votes to classify the derivative as a reflectance change. As explained in Section 3.2.2 each weak classifier’s vote is assigned a weight.

It is interesting to note that the filter with the greatest weight is very similar to Retinex. This filter is essentially applying Retinex to vertical derivatives of a smoothed image. This smoothing is necessary to filter noise and other high-frequency variations that do not indicate a reflectance change, such as the rough wall in the graffiti image. Assigning this filter the greatest weight shows that smoothing the input image before using the Retinex classifier would increase the Retinex classifier's performance. The AdaBoost classifier is advantageous because it takes advantage of the strengths of the Retinex classifier and is able to use the response of other filters in addition.

Another interesting aspect of the AdaBoost classifier is that seven of the ten filters chosen to classify vertical derivatives are horizontally oriented. This is related to the fact that in the training set, each image is illuminated from the top of the image. The choice of these filters indicates that it is unlikely that edges caused by shading will be oriented in the same direction as the illumination.
3.2.4 Results Using the AdaBoost Classifier

The AdaBoost classifier is able to classify 87% of the training set correctly. This is a significant improvement over the 60% that the Retinex classifier is able to achieve. We do not expect the classifier to attain 100% correct classification on the training set because the classifier relies on local evidence. In Chapter 4, we show how the classification of some derivatives, given only local evidence, is ambiguous. These ambiguous derivatives will prevent the classifier from attain 100% accuracy on the training set.

Figure 3-7 shows the results of the AdaBoost classifier on the painted sphere from Chapter 1. The results are similar to those obtained by the Retinex classifier, but the lines of pixels classified a reflectance changes are much thicker in the results from the AdaBoost classifier. This is likely due to the smoothing effect of the filters used. Retinex uses only the pixel and its neighbor to classify. The AdaBoost classifier uses filters that sum over a window around the pixel, making it likely that the neighbors of a pixel with a large response will also have a large response. Most of the weak classifiers chosen by the AdaBoost system interpret a large response as a reflectance change. If the filter response is large enough to make the weak classifiers vote for a reflectance change, it is likely that the response will also be large enough to signal a reflectance change.

The difference in results between the AdaBoost and Retinex classifier on the real images
Figure 3-8: Comparison of the outputs of the Retinex and AdaBoost classifiers on the vertical derivatives of the graffiti image. Vertical derivatives caused by reflectance changes are marked in white. (a) The input image (b) The output of the Retinex classifier. (c) The output of the AdaBoost classifier

In Figure 3-8(c), the classification computed using the AdaBoost classifier is much less noisy than the classification computed using Retinex.

In Figure 3-9(c), the classifications computed by the Retinex and AdaBoost classifiers are similar. The similarity between the two classifications is a sign that the training set did not generalize well to this image. The AdaBoost classifier is able to perform a significant 27% better on the training set than the Retinex classifier. The fact that this improvement does not carry over to the classification of the derivatives of the pillow image signals that the training set should be improved to generalize to real images better.

The other interesting difference between the classifications is the cheek patch. The Retinex classifier classifies the top and bottom edges of the patch as reflectance changes. On the other hand, the AdaBoost classifier marks the right and left sides as reflectance changes. This is likely caused by the fact that assuming the image is illuminated from the top of the image, a horizontal edge, such as the bottom part of the cheek, is a very likely
3.3 Using Color Information

While most of the face painted on the pillow in Figure 3-9 is correctly marked, the round cheek patch is not. As explained in Section 3.2.4, this likely because the top and bottom edges of the cheek patch locally appear very similar to shading. However, in the color image of the pillow, the cheeks are pink. This color changes provides strong evidence for classifying the lines around the cheek patches as reflectance changes. In this section, we discuss creating a classifier which uses color information to classify the derivatives.

Similar to [9], our system takes advantage of color information by relying on the property that changes in color between pixels indicate a reflectance change [17]. When surfaces are diffuse, any changes in a color image due to shading should affect all three color channels proportionally. Assume two adjacent pixels in the image have values $c_1$ and $c_2$, where $c_1$ and $c_2$ are RGB triplets. If the change between the two pixels is caused by shading, then $c_2 = \alpha c_1$ for some scalar $\alpha$. Intuitively, shading should only cause the intensity of surface’s color to change, not the chromaticity.
To measure the likelihood that a color change is caused by a reflectance change, we treat each RGB triplet as a vector and normalize them to create $\hat{c}_1$ and $\hat{c}_2$. We then use the angle between $\hat{c}_1$ and $\hat{c}_2$ to find reflectance changes. When the change is caused by shading, $(\hat{c}_1 \cdot \hat{c}_2)$ equals 1. If $(\hat{c}_1 \cdot \hat{c}_2)$ is below a threshold, then the derivative associated with the two colors is classified as a reflectance derivative. We found that a threshold equivalent to limiting changes due to shading to an angle of 0.86 degrees or less worked well. In some images, there were noticeable chromaticity artifacts caused by JPEG compression. To alleviate these, the images were smoothed with a $5 \times 5$ Gaussian filter first.

The output of the color classifier on the vertical derivatives of the pillow image is shown in Figure 3-10. The color classifier correctly marks the cheek patches as reflectance changes, but fails to classify the black face markings correctly. This is because the derivatives around the black face markings are intensity changes—the color goes from black to white. Under the rules of the color classifier, this is a valid shading change.

### 3.4 Combining Color and Gray-Scale Information

Neither the color nor the gray-scale classifier is able to classify the entire pillow image correctly. However, by combining the results of the two classifiers, the quality of the classifications should be enhanced significantly. In this section, we discuss combining the
outputs of the gray-scale AdaBoost classifier and the color classifier.

Color and gray-scale information can be used simultaneously by requiring the classifiers to assign probabilities to the classification of each derivative. To combine the results of the color and gray-scale classifiers, let $D$ be the classification of some derivative. We denote $D_s$ as the event that the derivative should be labelled shading and denote $D_r$ as the event that the derivative should be labelled a reflectance change. From the color classifier, we obtain $\Pr[D_s|C]$ and $\Pr[D_r|C]$. These are the probabilities of the derivative being caused by shading or a reflectance change, given the local color information $C$. The gray-scale classifier returns $\Pr[D_s|G]$ and $\Pr[D_r|G]$, the probabilities of the derivative’s classification, given the local gray-scale information $G$.

We assume that the outputs of the color and gray-scale classifiers are statistically independent variables. This enables us to express the probability of the derivative being shading, $\Pr[D_s|G,C]$, as

$$\Pr[D_s|G,C] = \frac{\Pr[D_s|G] \Pr[D_s|C]}{(\Pr[D_s|G] \Pr[D_r|C] + \Pr[D_r|G] \Pr[D_r|C])} \tag{3.7}$$

The probability that the derivative is caused by a reflectance change is found by

$$\Pr[D_r|G,C] = \frac{\Pr[D_r|G] \Pr[D_r|C]}{(\Pr[D_s|G] \Pr[D_r|C] + \Pr[D_r|G] \Pr[D_r|C])} \tag{3.8}$$

To obtain $\Pr[D_s|G]$, the probability of a derivative being caused by shading from the gray-scale classifier, we used the method suggested by Friedman et al. to transform the output of an AdaBoost classifier into the probability of each label [8]. Each sample, $x$, classified by the AdaBoost classifier has a true label, which we will refer to as $y$. This label is either 1 or -1. Given the sample, $x$, the AdaBoost classifier, $H(x)$, returns a value between -1 and 1. The probability that the true classification, $y$, has the value 1 is well approximated by

$$\Pr[y = 1] \approx \frac{e^{H(x)}}{e^{H(x)} + e^{-H(x)}} \tag{3.9}$$

In our system, the AdaBoost classifier uses $G(i,j)$, a patch of local image information around location $(i,j)$, to classify the derivative at location $(i,j)$. Using this approximation,
the probability that the derivative is caused by shading is

\[ \Pr[D_s \mid G(i,j)] = \frac{e^{H(G(i,j))}}{e^{H(G(i,j))} + e^{-H(G(i,j))}} \] (3.10)

We obtain the probabilities of each label, given the color information, in a different fashion. The output of the color classifier is transformed into a probability by setting the probability that a derivative is caused by shading, \( \Pr[D_s \mid C(i,j)] \), to be some constant probability wherever the color classifier finds a reflectance change. For the results shown, we use the value 0.1 for the constant. The probability that a derivative is shading is set to 0.5 at locations marked by the color classifier as shading. This is because the color classifier is unable to distinguish between shading and reflectance changes where only the intensity of the color changes.

Figure 3-11 shows the results of the Retinex, AdaBoost, color, and combined classifier together. The results of combining the gray-scale and color are shown in Figure 3-11(a). By combining the two classifiers, all of the features of the pillow image are classified correctly.
Figure 3-11: The classifications of the vertical derivatives of the pillow image obtained from the four classifiers discussed in this chapter.
In Chapter 3, the classifier uses a $17 \times 17$ pixel window to classify each derivative. While this works well, there are areas in the image where the local information is not sufficient to clearly classify the derivatives. An example of an ambiguous image patch, taken from the pillow image, is shown in Figure 4-1. The $17 \times 17$ area outlined in the center of the mouth area is very similar in appearance to examples of both shading and reflectance changes. Example training samples taken from both the shading and reflectance training sets which resemble this area of the pillow are shown in Figures 4-1(b) and 4-1(c). Using local image information alone, the classification of the derivative at the center of the outlined area is ambiguous.

Ambiguous areas, such as the mouth on the pillow, are problematic because they are prone to misclassifications. While the classifier shown in Chapter 3 correctly classifies the

![Figure 4-1: Comparison of the mouth from the pillow and an ellipse from the training set.](image)
Figure 4-2: The classification of ambiguous areas of the pillow image. Reflectance changes are marked with white pixels. Notice that the center of the mouth is misclassified as shading.

mouth, the potential problems in ambiguous areas of an image can be illustrated by training a new classifier using a new training set drawn in the exact same fashion as described in Chapter 3. Figure 4-2 shows the output of this classifier on the mouth of the pillow image. This classifier misclassifies a large section on the center of the mouth. Areas with clear classifications will be assigned the same label by both the original classifier and this newly-trained classifier, but it is very uncertain whether ambiguous areas will be classified correctly.

While the center of the mouth is ambiguous, the corners of the mouth can be classified as being caused by a reflectance change with much more confidence. Since the points in the center and corners of the mouth all lie along a single image contour, they should have the same classification. A mechanism is needed to propagate information along the contour from the corners of the mouth, where the classification is clear, into areas where the local evidence is ambiguous. This will allow areas where the classification is clear to disambiguate those areas where it is not. In this chapter, we present a mechanism which propagates information along image contours to improve the classifications returned by the classifier in ambiguous areas of the image.

4.1 Propagating Information

In order to propagate information, we treat the classification of each derivative as a random binary variable. Our goal is to find the most probable setting of each random variable. The local image classifiers from Chapter 3 provide the probability of the derivative being
caused by shading or a reflectance change, given the local image information. If this is not sufficient, we must also consider the classifications of neighboring derivatives. The information about neighboring derivatives can be incorporated by also considering the joint distribution of the classifications. Given a joint distribution, some classifications of all the derivatives in the image will be more probable than others.

Consider a simple image of an edge, such as the one in Figure 4-3(a), and its horizontal derivatives, shown in Figure 4-3(b). The random variables representing the classification of each derivative are depicted as graph nodes placed over each derivative in Figure 4-3(c). In Figure 4-4(a), nodes labeled with a red circle have been classified as reflectance changes by the image classifier, while nodes labeled with an ‘?’ are ambiguous. If the joint probability distribution of the nodes reflects our intution that derivatives along an image contour should have the same classification, then the configuration in Figure 4-4(b), where all of the nodes along the contour have the same classification, will be more likely than the configuration in Figure 4-4(c), where the ambiguous nodes have been labelled as shading.
Derivative's cause is ambiguous

- Derivative caused by a reflectance change
- Derivative caused by shading

Figure 4-4: An example of using the joint probability distribution of the derivatives to fill in ambiguous derivatives. (a) The output of a classifier which uses image information. Nodes marked with a red circle are clearly reflectance changes. Nodes marked with a ‘?’ are ambiguous. (b) and (c) Two possible states of the Markov Random Field. (b) The ambiguous nodes are classified as reflectance changes. (c) The ambiguous nodes are classified as shading, denoted by a blue square.

4.2 Probability Model for Propagating Information

To find the right joint distribution for the nodes, we model the distribution as a Markov Random Field. In a Markov Random Field, or MRF, the probability of a node’s state is only dependent on the neighboring nodes and the information from the image classifier [5]. The dependency relationships between the variables can be illustrated by drawing lines from each node to the nodes that it depends on. The MRF for the simple edge image is shown in Figure 4-5.

The probability density function of a Markov Random Field is

\[ P(x) = \frac{1}{Z} \prod_i \phi(x_i) \prod_{(j,k)} \psi(x_j, x_k) \]  

(4.1)

where \( x \) is a possible state of the MRF, \( x_i \) is the value of node \( i \) in that state, nodes \( j \) and \( k \) are the indices of neighboring nodes, and \( Z \) is a normalization constant. The function \( \phi(x_i) \) is the local evidence for node \( i \). This is the probability of each possible state of node
Figure 4-5: The derivative image with variables and lines relating the dependencies between variables. Lines depicting each node's dependency on image information are not shown.

\[ i, \text{ given the image information. In our system, this is provided by the image classifiers.} \]

The function \( \psi(x_j, x_k) \) is the compatibility function between two neighboring nodes. It relates the joint probability of each possible setting of nodes \( j \) and \( k \). It can also be thought of as the compatibility between every state of \( x_j \) and every state of \( x_k \). Setting the compatibility functions, \( \psi(\cdot) \), allows us to control how the nodes in the MRF influence each other. Setting the compatibility functions along a contour in the image to favor \( x_j \) and \( x_k \) having the same state would favor all of the nodes along the contour having the same classification.

We limit the compatibility functions to be symmetric, so they only constrain whether two neighboring nodes must have the same value or not. Thus, each potential function, \( \psi(x_i, x_j) \) has two possible values, one when \( x_i = x_j \) and one when \( x_i \neq x_j \). If nodes \( i \) and \( j \) lie along an image contour, \( \psi(\cdot) \) should be set so that \( x_i = x_j \) is the most compatible setting. This dependence of \( \psi(\cdot) \) on whether there is a contour in the image can be expressed by defining the compatibility function \( \psi_0(I) \) as the compatibility when the two nodes have a different value and \( \psi_1(I) \) to be the compatibility when the nodes have the same value. Writing \( \psi \) as a function of the input image, \( I \), makes it clear that the compatibility between the states of neighboring nodes is dependent upon whether there is a contour in the image near the location of \( \psi \).
4.3 Learning the Compatibility Functions

To simplify learning the potential functions, $\psi_1(I)$ is set to $1 - \psi_0(I)$ with $\psi_0(I)$ constrained to be between 0 and 1. This constraint is fulfilled by specifying that $\psi_0(I) = g(z(I))$, where $g(\cdot)$ is the logistic function and $z(I)$ corresponds to how necessary it is for two neighboring nodes to have the same value.

The function relating the image to the compatibility between nodes, $z(I)$, is found from training data. As input for $z(\cdot)$, we use two local image features that reflect our heuristic that derivatives along an image contour should have the same classification. The features are the magnitude of the image gradient and the difference in orientation between the gradient and the orientation of the graph edge.

The difference in orientation between a graph edge and image contour, $\hat{\theta}$, is found from the orientation of the image gradient, $\theta$, as shown in , see Figure 4-6. Assuming that $-\pi/2 \leq \theta \leq \pi/2$, the angle between a vertical edge and the image gradient, $\hat{\theta}$, is $\hat{\theta} = |\theta| - \pi/2$. For horizontal edges, $\hat{\theta} = |\theta|$.

To find the values of $z(\cdot)$ we maximize the probability of a set training examples, similar to those used in Chapter 3, over the parameters of $z(\cdot)$. This training is done with labelled data, so there is no local evidence. This makes the probability of each training sample

$$ P = \frac{1}{Z} \prod_{(i,j) ; x_i \neq x_j} \psi_0(I) \prod_{(i,j) ; x_i = x_j} \psi_1(I) $$

where all $(i, j)$ are the indices of neighboring nodes in the MRF and $Z$ is a normalization constant. We constrain $z(\cdot)$ to be a linear function of the form
\[ z(\phi, |\nabla I|) = a \cdot \phi + b \cdot |\nabla I| + c \]  \hspace{1cm} (4.3)

where \( |\nabla I| \) is magnitude of the image gradient and the constants, \( a, b, \) and \( c, \) are to be found by minimizing Equation \hspace{1cm} 4.2. In addition, both \( \hat{\phi} \) and \( |\nabla I| \) are normalized to be between 0 and 1. These measures break down in areas with a weak gradient, so we set \( z(\cdot) \) to 0 for regions of the image with a gradient magnitude less than 0.05.

The constants, \( a, b, \) and \( c, \) are found by maximizing equation 4.2 over a set of training images similar to those used to train the local classifier. In order to simplify the training process, we approximate the true probability of each MRF by assuming that \( Z \) is constant. Doing so leads to the following values for \( a, b, \) and \( c: a = -1.2, b = 1.62, c = 2.3. \)

Larger values of \( z(\cdot) \) correspond to a belief that the neighboring derivatives should have the same value, while negative values signify that the derivatives should have a different value. These values found for the constants of \( z(\cdot) \) correspond with our expected results; two derivatives are constrained to have the same value when they are along an edge in the image that has a similar orientation to the edge in the MRF connecting the two nodes.

4.4 Inferring the Correct Labelling

Given an input image, the best labelling of the derivatives in that image can be found by first applying \( z(\cdot) \) to the gray-scale image to find the compatibility functions between neighboring derivatives. With these compatibilities and the local image information provided by the classifiers, the most probable labelling of each derivative can be found. This is done by finding the labelling of all the derivatives, \( x, \) that maximizes the probability of the Markov Random Field, shown in Equation 4.1.

Finding the labelling of the MRF with the maximum likelihood has been shown to be an NP-complete problem; making it intractable on large images. However, an approximate solution can be found in reasonable time using algorithms based on Belief Propagation [5]. In the Belief Propagation algorithm, nodes in the Markov Random Field pass messages to their neighboring nodes. Information is propagated as each node passes messages to
Belief Propagation

Generalized Belief Propagation

Figure 4-7: Comparison of results using Belief Propagation and Generalized Belief Propagation. Generalized Belief Propagation correctly classifies the entire mouth as a reflectance change.

its neighbors, receives messages, then uses these messages to compute the next messages to send. We found it necessary to the Generalized Belief Propagation algorithm [21] to infer the best label of each node in the MRF because ordinary Belief Propagation performed poorly in areas with both weak local evidence and strong compatibility constraints. Generalized Belief Propagation finds a better approximation by passing messages between groups of nodes in addition to passing messages between nodes.

Pixels marked in white in Figure 4-7(a) mark vertical derivatives classified as reflectance changes after using the Belief Propagation on the the classifications shown in Figure 4-2. The Belief Propagation algorithm actually worsened the original classification by classifying an even larger area as shading changes. On the other hand, the Generalized Belief Propagation algorithm correctly fills in the ambiguous areas as reflectance changes.
Chapter 5

Recovering Shading and Reflectance Images

With the addition of the Belief Propagation step to the system, the steps of our system are:

1. Compute the horizontal and vertical derivatives of the image.

2. Classify each derivative as being caused by shading or a reflectance by combining the results of the gray-scale and color classifiers.

3. Use Belief Propagation to improve the classifications in ambiguous areas of the image.

4. Reconstruct the shading and reflectance images from the estimated derivatives of the shading and reflectance images.

In this chapter, we discuss the final step in our system, recovering the shading and reflectance images from the estimates of their derivatives. We also show the results of our system on both the images used as examples in previous chapters and additional real images.
5.1 Recovering the Images

As discussed in Section 3.2.4, our algorithm estimates the derivatives of the shading and reflectance images by classifying each image derivative as being caused by either shading or a reflectance change. The shading derivatives can then be found by setting every derivative labelled as a reflectance change to 0. This is justified by assuming that it is unlikely that significant shading boundaries and reflectance edges occur at the same point, thus we make the simplifying assumption that every image derivative is either caused by shading or reflectance.

After classification, our system has two sets of horizontal and vertical derivatives, one for the shading image and one for the reflectance image. Each derivative represents a set of linear constraints on the associated image. Using both derivative images results in an over-constrained system. We recover each intrinsic image from this over-constrained system by using the method introduced by Weiss in [20] to find the pseudo-inverse of the system of derivatives. If \( f_x \) and \( f_y \) are the filters used to compute the \( x \) and \( y \) derivatives and \( F_x \) and \( F_y \) are the estimated derivatives of shading image, then the shading image, \( S(x, y) \) is:

\[
S(x, y) = g \ast [(f_x(-x, -y) \ast F_x) + (f_y(-x, -y) \ast F_y)]
\]  

(5.1)

where \( \ast \) is convolution, \( f(-x, -y) \) is a reversed copy of \( f(x, y) \), and \( g \) is the solution of

\[
g \ast [(f_x(-x, -y) \ast f_x(x, y)) + (f_y(-x, -y) \ast f_x(x, y))] = \delta
\]  

(5.2)

The reflectance image is found in the same fashion. One nice property of this technique is that the computation can be done using the Fast Fourier Transform, making it more computationally efficient.

5.2 Results

In this section, we show shading and reflectance images generated from real images.
5.2.1  Computational Details

The gray-scale classifier consisted of ten non-linear filters, trained as described in Chapter 3. Before processing, each image was normalized so that the intensity of every pixel is between 0 and 1. The horizontal and vertical derivatives were computed by convolving the image with [-1 1 0] filters. In the images where Generalized Belief Propagation was used, the number of iterations that the Generalized Belief Propagation algorithm was allowed to run was equal to half the size of the image.

On calibrated images, the processing should actually occur on the logarithm of the input image. This makes the input image the sum of the shading and reflectance images. Our test images are uncalibrated so operating on the logarithm of the input image is not correct. However, ordinary photographic tonescale is similar to a log transformation, so we apply our system to untransformed input images.

For each of the images, we must specify the direction of illumination. The possible directions are from the right, left, top, or bottom of the image. Bell and Freeman suggest choosing the illumination direction by choosing the direction where the shading image accounts for the largest portion of the changes in the image.[3] This method would also work with our system.

In images with color information available, the reflectance image is shown in color. This makes the performance of the algorithm easier to evaluate. The color reflectance images are found by independently reconstructing all three color bands using the same classifications for each color band.

5.2.2  Results on Painted Sphere

Figure 5-1 shows the results of our system on the painted sphere, first introduced in Chapter 1. These results were computed using only the gray-scale classifier. Propagation was not used because there are no ambiguous areas in the image.

The dots on the sphere are recovered correctly, although some artifacts in the shading image remain around where the dots used to be. These are artifacts of using binary classification to estimate the derivatives.
In the reflectance image, the sphere is shown to have a different reflectance than the rest of the surface. This is because, as discussed in Section 3.2.4, the borders of the sphere are marked as reflectance changes.

### 5.2.3 Results on Graffiti Image

Figure 5-2 shows the results of our system on the graffiti image from Chapter 3, with and without using the propagation step. Overall, the separation of shading and reflectance is good. The bright spots in the reflectance image are from misclassified shading derivatives. The reflectance image also contains the ridges that were misclassified by the gray-scale classifier.
classifier. These ridges are likely misclassified because the training set does not contain ridges such as these. The propagation step does not significantly reduce the number of edges that should not appear in the reflectance image, but it does reduce some.

Figure 5-3 shows an enlarged portion of the image from Figure 5-2. This image shows how the shading variations of the surface have been retained in the shading image.
5.2.4 Results on Pillow Image

Figures 5-4, 5-5, and 5-6 show the results of our system on the pillow image from Chapter 3, using both the combined output of the color and gray-scale classifiers and the output of the classifiers individually. Results are shown with and without propagating the information from the classifiers.

In Figure 5-4, the algorithm separates the shading and reflectance of the scene very well. After using Belief Propagation, the face and cheek patches are only present in the reflectance image, while the folds of the pillow are in the shading image. The reflectance image does contain some small variations on the left side of the pillow that should be in the shading image. These variations are caused by the classifier misclassifying some shading derivatives as reflectance changes.

This image is also a good example of the benefits of propagation. Without propagation,
Figure 5-5: Results on the pillow image from Chapter 3 using only gray-scale information.

Figure 5-6: Results on the pillow image from Chapter 3 using only color information.
portions of the mouth and eyebrow are recovered in the shading image. As discussed in Chapter 4, these areas are misclassified as shading because they are locally ambiguous. After propagating information from areas of the image with a clear classification, the mouth and eyebrow contours are placed entirely in the reflectance image.

Figure 5-5 shows the images recovered if only the gray-scale classifier is used. The face is recovered correctly, but the cheek patch is still present in the shading image.

Figure 5-6 shows the images recovered if only the color classifier is used. In this image, the cheek is recovered correctly, but the face is not.
Figure 5-7: Results on an image of vegetables using color and gray-scale information combined. The illumination is assumed to come from the top of the image.

5.2.5 Results on Vegetable Image

Figures 5-7, 5-8, and 5-9 show the results on an image of vegetables on a plate. In the results using both color and gray-scale information, shown in Figure 5-7, the boundaries between the carrots, broccoli, and plate are all present in the reflectance image. In addition, most of the intensity variations in the broccoli are placed in the shading image. In the reflectance image, the variations in the broccoli have been flattened.

The greatest errors are on the plate. The shading along the edge of the plate, near the broccoli, is in the reflectance image. In addition, the small bumps on the plate, on the left side of the image, are placed in the reflectance image. Examining the results from only using the color image, in Figure 5-9, shows that these bumps are classified as reflectance changes by the color classifier. This is likely because of inter-reflection. The carrots cause some of the bumps to have a slight orange tint to them. This causes the color classifier to
Figure 5-8: Results on an image of vegetables using only gray-scale information.

Figure 5-9: Results on an image of vegetables using only color information.
classify them as reflectance changes.
Figure 5-10: Results on an image of a girl using color and gray-scale information combined. The illumination is assumed to come from the right of the image.

### 5.2.6 Results on Image of a Girl

Figures 5-10, 5-11, and 5-12 show the results on an image of a girl. This classifier separates the boundary between the arm and clothes very well. This image is also another example of the value of combining gray-scale and color information. The sleeve on the right is not correctly labelled by the color classifier, but is marked correctly by the gray-scale classifier. This causes it to be marked correctly in the output of the combined classifier.
Figure 5-11: Results on an image of a girl using only gray-scale information.

Figure 5-12: Results on an image of a girl using only color information.
Figure 5-13: Results on an image of a bed sheet using color and gray-scale information combined. The illumination is assumed to come from the right of the image.

5.2.7 Results on Sheet Image

Figures 5-13, 5-14, and 5-15 show the results on an image of a bed-sheet. The patterns on the sheet are almost totally removed from the shading image. Propagation improves the performance around the lines running down the center of the sheet. After propagation, they are less noticeable.
Figure 5-14: Results on an image of a bed sheet using only gray-scale information.

Figure 5-15: Results on an image of a bed sheet using only color information.
Figure 5-16: Results on an image of a toy using color and gray-scale information combined. The illumination is assumed to come from the right of the image.

### 5.2.8 Results on Toy Image

Figures 5-16, 5-17, and 5-18 show the results on an image of a toy. The shading on the legs of the action figure is correctly placed in the shading image. The shading on the green chest area is also recovered correctly. The system makes a mistake by classifying the shading on the green arm bands as a reflectance change.
Figure 5-17: Results on an image of a toy using only gray-scale information.

Figure 5-18: Results on an image of a toy using only color information.
Figure 5-19: Results on an image of a raft using color and gray-scale information combined. The illumination is assumed to come from the right of the image.

### 5.2.9 Results on Raft Image

Figures 5-19, 5-20, and 5-21 show the results on an image of a raft. The results produced by just the gray-scale classifier, shown in Figure 5-20, are excellent. However, the color classifier attributes most of the variations in the image to reflectance changes. This could be caused by non-linearities in the image or the threshold in the color classifier being too tight.
Figure 5-20: Results on an image of a raft using only gray-scale information.

Figure 5-21: Results on an image of a raft using only color information.
Chapter 6

Conclusion

In this thesis, we have presented a system which recovers the shading and reflectance images of a scene from a single image by using the following steps:

1. Compute the horizontal and vertical derivatives of the image.

2. Classify each derivative as being caused by shading or a reflectance by combining the results of the gray-scale and color classifiers.

3. Use Belief Propagation to improve the classifications in ambiguous areas of the image.

4. Reconstruct the shading and reflectance images from the estimated derivatives of the shading and reflectance images by using Weiss’ pseudo-inverse method [20].

In this chapter, we review the contributions of this thesis and suggest future work that could be used to improve the system.

6.1 Contributions

In this thesis, we have described a system that is able to recover good results on real images. The architecture of this system also contains several significant contributions.
6.1.1 Classifying Binary Derivatives

We have shown that good results can be obtained by estimating the derivatives of the shading and reflectance images by simply classifying each derivative as belonging to either the shading or reflectance image. This is advantageous because binary classification is a considerably easier problem than estimating the actual derivatives. In addition, the problem of binary classification has been extensively studied in the machine learning community. Using binary classification allows the system to be extended by using advanced classification techniques, such as Support Vector Machines.

6.1.2 The Use of Multiple Cues

Our system is able to use multiple cues, both color and gray-scale information, to classify the derivatives. This is important because, as shown in Chapter 3, both are needed to correctly classify every derivative. We have also shown the value of propagating information along contours in the image to correctly classify ambiguous areas of the image.

6.1.3 Probabilistic Framework

Treating the classification and propagation of information in a probabilistic framework is important because it allows the system to be easily extended. The gray-scale and color classifiers can be changed independently, but the framework does not need to be changed to combine the information from the two. In addition, as long as any new classifiers and cues can be expressed probabilistically, they can be incorporated easily. The propagation step can also be easily changed by changing the how the image information is translated into compatibility functions.

6.1.4 A Learning Based Classifier

We have created the gray-scale classifier by training it using the AdaBoost algorithm. The classifier performs well on the training set and performs feature selection automatically.
6.2 Future Work

Almost every aspect of this system has interesting extensions. In this section, we outline several of the most significant areas in which this work could be improved.

6.2.1 Estimating Derivatives

While binary classification works well and is powerful, it does cause artifacts in the recovered images. Figure 5-1 is an obvious example of the artifacts. Edges appear in the shading image around the locations of the dots in the reflectance image. These occur because both shading and reflectance derivatives are located at those points. The artifacts would be improved significantly if both shading and reflectance derivatives could be present at every point.

6.2.2 Improved Color Classifier

The color-based classifier uses a very simple heuristic for separating shading and reflectance derivatives. This classifier could be extended to consider illumination effects such as specularities or inter-reflection among objects.

6.2.3 Improved Training Set

The training set used in this thesis was created synthetically. The system would generalize to real scenes much better if it were trained natural images. The greatest obstacle to this is obtaining ground-truth images. Color information has proved to be a very reliable indicator of reflectance changes. An improved color classifier applied to properly chosen images may be able to provide good enough data to train the gray-scale classifier.
Bibliography


