Reducing Spectral Reflections through Image Inpainting

by

Joshua Runge

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

Masters of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2009

Author .....
Department of Electrical Engineering and Computer Science
February 9, 2009

Certified by .....
Sai Ravela
Research Scientist
Thesis Supervisor

Accepted by .....
Arthur C. Smith
Chairman, Department Committee on Graduate Theses

Reducing Spectral Reflections through Image Inpainting

by

Joshua Runge

Submitted to the Department of Electrical Engineering and Computer Science
on February 9, 2009, in partial fulfillment of the
requirements for the degree of
Masters of Engineering in Electrical Engineering and Computer Science

Abstract

The goal of this thesis is to produce an image processing tool that could, with limited user interaction, significantly reduce spectral reflections. The resulting process can then be used by biological researchers in the process of cataloging and recognizing individual organisms via collected images. The process involves the generation of a mask, refinements to that mask such as background segmentation and morphological dilation, and finally the inpainting of the specular regions. This method improves on existing single pixel methods by incorporating both color and texture infilling of the specular reflections. It also works on images with no special setup or preparation.

Thesis Supervisor: Sai Ravela
Title: Research Scientist

Acknowledgments

First, I would like to thank Sai Ravela, my thesis supervisor for all the help and guidance throughout this process. He has given me both directions to take my studies and insight to help narrow and contain them. He has contributed so many unique ideas to my research that it is unfortunate that I was only able to pursue a small number of them.

Second, I would like to thank the faculty and students who aided me in this process. Specifically Professor Fredo Durand's instruction on Computational Photography provided the core knowledge for the pursuit of this research. Also Chris Yang was both a welcome audience for approaches and ideas and a skilled collaborator.

Finally, I would like to thank my family and friends who listened politely to all my crazy ideas and provided the motivation and encouragement to complete this endeavor.

This work is funded in part by NSF DBI-0640529, NSF CNS-0540259, and the Edgerly Innovation Fund Award.

Contents

List of Figures

Chapter 1

Introduction

The goal of this thesis is to produce a tool that would allow researchers with limited image processing knowledge to effectively remove glare from previously collected images. This method is intended to be used as part of a collection of image processing tools focused on identification of individual biological organisms [3]. Images that will be processed using these tools have already been captured by the researching scientists, which provided the sample set. The algorithms and techniques used may be extended to other image types and research areas, however this work will limit its scope to specific image styles and organisms.

This chapter first describes the motivation for this work as a biological research tool. Furthermore, it provides some general image processing definitions and descriptions which will be useful in understanding and implementing this work.

Chapter two provides information on related image processing algorithms that were investigated for this thesis. They provide insight into the thought processes that motivate this research as well as expose some strengths and limitations of this work.

Chapter three outlines the method for glare removal. This process involves both the specularity mask generation step and the inpainting steps to fill the masked regions.

Chapter four provides results from this investigation. This includes image examples of the techniques used as well as some problems encountered. This chapter also

specifies the user inputs to the algorithms.

Chapter five draws some conclusions about the results and final design choices as well as discusses further opportunities for development of these ideas.

1.1 Motivations for Glare Reduction Research

Photographers have long battled to reduce unwanted image artifacts such as glare. Most of the current techniques involve careful preparation of the scene or camera to stop such glare from being captured in the first place. Soft flashes can reduce the harsh lighting that leads to blown out highlights. Lens filters can be used to polarize the light coming into the camera and reduce the appearance of light being directly reflected. In short, good equipment and tight environmental controls are the best physical methods for reducing glare.

Though these techniques work well for professional photography, current trends have led to the proliferation of cheap and simple cameras, allowing many less experienced users to capture photos. In most cases, these users are not able or willing to invest in expensive filters or flash setups for their cameras to reduce glare. Furthermore, this insight is only useful for future image captures. Any previously collected images may already have glare.

For this thesis in particular, the focus is on reducing glare in currently collected images for biological researchers. Biologists use cameras to catalog animals that cannot be safely or easily tagged. In order to manage the large number of photographs collected, these researchers turn to vision-based methods to automatically identify individual animals. These methods are sensitive to lighting changes and would perform more accurately if glare was removed from the photos before processing. The method discussed in this thesis will become part of a large set of image processing tools available to researchers in this field.

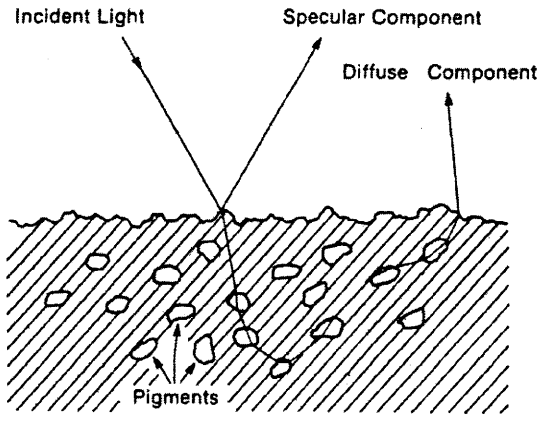

Figure 1-1: Reflections from a surface with specular and diffuse components[4]

1.2 Background Glare Definitions

Glare is more commonly referred to in the image processing world as specular reflection. To understand it, we must first characterize the interactions of light on an imaged object. As light interacts with the surface of an object, it can either be immediately reflected or absorbed into the material. This reflected component is generally very directed light and close to the color of the illuminant. The absorbed light scatters within the material and some is reflected back to the viewer. This light is generally much less directed, as the light can exit the surface in random directions. The immediately reflected light is considered the specular component, while the absorbed light is called diffuse[4].

When a camera images an object, the photosensor measures a value for three color components of light, red, green and blue. These values correspond to both the specular and diffuse light measured at each sensor location or pixel. According to the dichromatic reflection model, the color of a light observed at a pixel is a linear combination of the specular light and the diffuse light[9]. This combination is shown in equation 1.1 where S is the specular component, D is the diffuse component and w_s and w_d are the magnitudes of each component respectively. This is the central principle for most specularity reduction research. If one could separate the diffuse and specular contributions at each pixel, either could be removed without affecting

the other.

$$I(p) = w_d(p)D(p) + w_s(p)S(p) \tag{1.1}$$

Finally, some of the most important image processing operations for specularity reduction are color space transformations. Specifically in this thesis, the HSV color space is extremely useful. Instead of representing each color as a red, green and blue value, a color is described by its Hue, Saturation and Value. Figure 1-2 illustrates this color representation. The hue is a measure of the angle on the color wheel. The saturation is a measure of the difference between gray and the chosen color. The value is a measure of the intensity of the pixel, between black (no intensity) and white (full intensity). Since this color representation is usually computed from the standard RGB colors, the equations used for this transformation are shown in 1.2,1.3,1.4.[13]

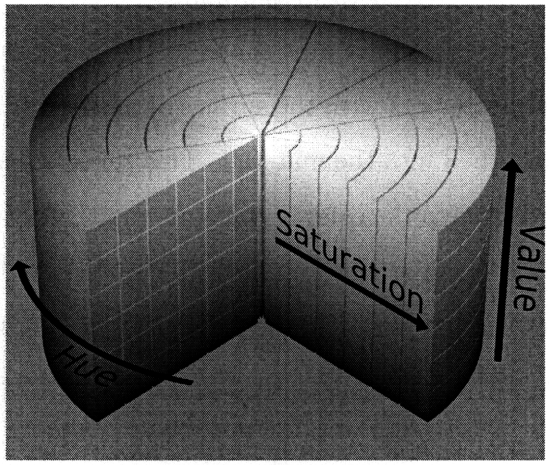

Figure 1-2: A visualization of the HSV color space [13]

$$h = \begin{cases} 0 & \text{if } \max(r,g,b) = \min(r,g,b) \\ (60° \times \frac{g-b}{\max(r,g,b)-\min(r,g,b)} + 0°) \mod 360°, & \text{if } \max(r,g,b) = r \\ 60° \times \frac{b-r}{\max(r,g,b)-\min(r,g,b)} + 120°, & \text{if } \max(r,g,b) = g \\ 60° \times \frac{r-g}{\max(r,g,b)-\min(r,g,b)} + 240°, & \text{if } \max(r,g,b) = b \end{cases} \tag{1.2}$$

$$s = \begin{cases} 0, & \text{if } \max(r,g,b) = 0 \\ \frac{\max(r,g,b)-\min(r,g,b)}{\max(r,g,b)} = 1 - \frac{\min(r,g,b)}{\max(r,g,b)}, & \text{otherwise} \end{cases} \tag{1.3}$$

$$v = \max(r,g,b) \tag{1.4}$$

Chapter 2

Related Work

As stated in section 1.2, a camera measures light reflected from an object, which can be decomposed into a linear combination of specular and diffuse light. There are many research papers detailing methods for this separation, but the most simplistic use multiple images to extract information about the proportions of each kind of light. Nayar et al. [8] use polarizing filters on images of colored objects to compute the specular component of the reflected light. Since specular reflections never enter the object, they tend to be highly polarized. By varying the polarization angle of the filter, it is possible to determine the intensity contribution of the specular reflection. Lee and Bajcsy[5, 6] use a different approach involving color histograms to extract similar information. Their method relies on the idea that the color of diffuse reflections is not dependent on the location of the illuminant. By changing viewing positions, they vary the specular reflections while preserving the diffuse information.

Single image methods use various image properties to perform the specular diffuse separation without multiple samples. Below, I examine two approaches which use the chromaticity of the image to reduce specular reflections. Tan [11] proposes to create a diffuse only image by iteratively comparing local pixel chromaticity. Shen et al. [10] use the same chromaticity idea to determine the diffuse color of each specular pixel and then a least-squares method to compute the magnitudes w_s and w_d of each component S and D.

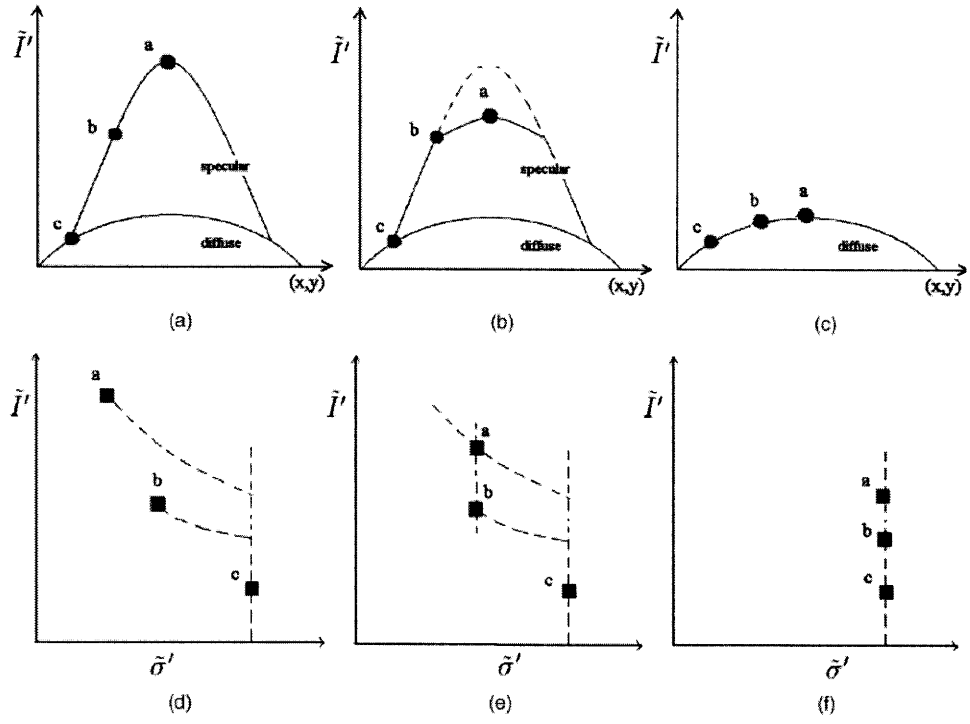

Figure 2-1: Basic principles of the intensity shifting operation. Top row is in the spatial-intensity space; the bottom row, the chromaticity intensity space. [11]

2.1 Tan's Chromaticity Based Specular Reflection Algorithm

Tan [11] proposes a method iteratively comparing the intensity difference between a specular free image, based on the chromaticity, and input image. The operation was done locally, through two pixel operations described in figure 2-1. Assuming a uniformly colored surface, pixel a is the brightest followed by pixel b and finally pixel c which is diffuse. Pixel a is transformed in the chromaticity space until its maximum chromaticity is equal to pixel b. This process is then repeated for pixel b and again for pixel a until all the maximum chromaticities are equal. This has the effect in the intensity space of reducing the pixel intensities of pixel a and pixel b to the diffuse curve of pixel c.

2.1.1 Reflection Model and Chromaticity

Tan's [11] method begins with a few definitions based on the dichromatic reflection model described in equation 2.1. The first part of the right hand side represents the diffuse component and the second part the specular component. From this, the chromaticity can be defined as in equation 2.2. It will also be useful to define the diffuse chromaticity ($w_s = 0$) in equation 2.3 and the specular or illuminant chromaticity in equation 2.4.

$$I_i(x) = w_d(x)D_i(x) + w_s(x)S_i(x), i = (r, g, b) \tag{2.1}$$

$$\sigma_i(x) = \frac{I_i(x)}{I_r(x) + I_g(x) + I_b(x)} \tag{2.2}$$

$$\Lambda_i(x) = \frac{D_i(x)}{D_r(x) + D_g(x) + D_b(x)} \tag{2.3}$$

$$\Gamma_i(x) = \frac{S_i(x)}{S_r(x) + S_g(x) + S_b(x)} \tag{2.4}$$

From these equations, we can rewrite 2.1 as:

$$I_i(x) = m_d(x)\Lambda_i(x) + m_s(x)\Gamma_i(x) \tag{2.5}$$

where

$$m_d(x) = w_d(x)[D_r(x) + D_g(x) + D_b(x)] \tag{2.6}$$

$$m_s(x) = w_s(x)[S_r(x) + S_g(x) + S_b(x)] \tag{2.7}$$

In order for this and many other specular separation algorithms to proceed, it is necessary to normalize the specular component to be pure white. First, we must estimate the color of the scene illuminant. There are several method to perform this estimation robustly, such as the one proposed by Tan [12], so it will not be discussed in detail here. Tan's estimation computes $\Gamma_{est,i}$ such that $[\frac{\Gamma_r}{\Gamma_{est,r}}, \frac{\Gamma_g}{\Gamma_{est,g}}, \frac{\Gamma_b}{\Gamma_{est,b}}] = [1, 1, 1]$.

From this estimation method, a normalized image can be derived as in equation 2.8 where $m'_d(x)$, $\Lambda'_i(x)$, $m'_s(x)$ are as in 2.9 2.10 2.11. This normalization makes the specular reflection component a scalar value.

$$I'_i(x) = m'_d(x)\Lambda'_i(x) + m'_s(x)1/3 \tag{2.8}$$

$$m'_d(x) = m_d(x)[\frac{\Lambda_r(x)}{\Gamma_{est,r}} + \frac{\Lambda_g(x)}{\Gamma_{est,g}} + \frac{\Lambda_b(x)}{\Gamma_{est,b}}] \tag{2.9}$$

$$\Lambda'_i(x) = \frac{\Lambda_i(x)}{[\frac{\Lambda_r(x)}{\Gamma_{est,r}} + \frac{\Lambda_g(x)}{\Gamma_{est,b}} + \frac{\Lambda_b(x)}{\Gamma_{est,g}}]} \tag{2.10}$$

$$m'_s(x) = 3m_s(x) \tag{2.11}$$

2.1.2 Specular/Diffuse Separation

$$\tilde{\sigma}(x) = \frac{max(I'_r(x), I'_g(x), I'_b(x))}{I'_r(x) + I'_g(x) + I'_b(x)} \tag{2.12}$$

$$\Lambda_{max,i}(x) = \frac{max(D_r(x), D_g(x), D_b(x))}{D_r(x) + D_g(x) + D_b(x)} \tag{2.13}$$

Step 1: In order to separate the diffuse and specular components of the image, first this method generates a specular free (SF) image $I_{SF}(x)$. This is done by finding the maximum chromaticity as defined in equation 2.12 and setting the diffuse maximum chromaticity 2.13 to some arbitrary scalar value. This generates an image with no specular reflections, but also incorrect diffuse color. Below is an example of an image and the generated SF image in figure 2-2. The SF image preserves the hue of the image and thus its geometric structure while setting the saturation to be constant. Below is an example of an image and the generated SF image. This will be a useful insight later in showing that specularity is directly related to the saturation and intensity of the image.

Step 2: In this step, the algorithm determines if the normalized image contains

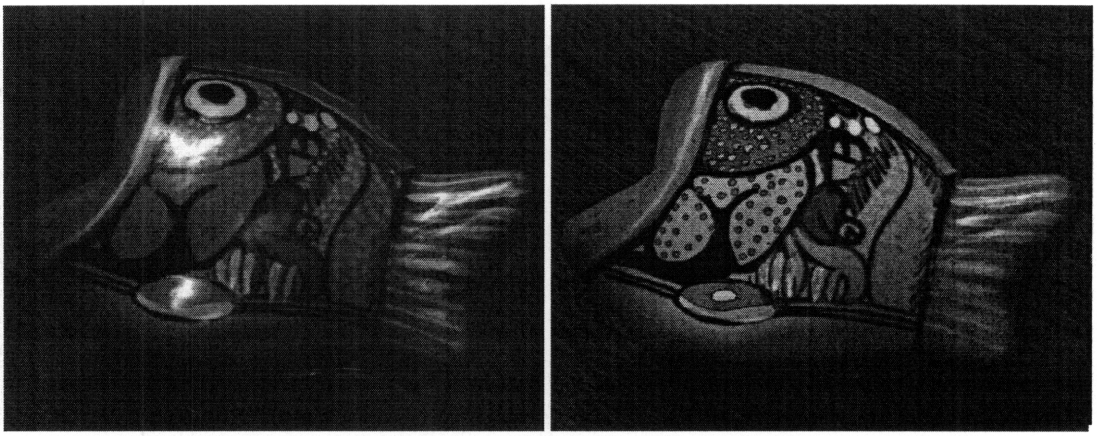

Figure 2-2: Original image and generated SF image [11]

only diffuse only pixels. If so, the process terminates, if not, the intensity of the specular pixels will be decreased. This process continues iteratively until there are only diffuse only pixels.

Tan [11] shows that the differential logarithmic of the normalized image and the differential logarithmic of the SF image are exactly identical for diffuse pixels as shown in equation 2.14. The differential logarithmic is computed by taking the derivative of the image with respect to each pixel in the natural log space. Next, $\Delta(x)$ is computed from the differential logarithmic of the normalized image and the SF image. Based on this, it is possible to determine if two neighboring pixels are diffuse as shown in 2.16. Finally, we can determine if the pixels are specular or a color discontinuity based on simple thresholds as in 2.17.

$$\frac{d}{dx}log(I'(x)) = \frac{d}{dx}log(I_{sf}(x)) \tag{2.14}$$

$$\Delta(x) = \frac{d}{dx}log(I'(x)) - \frac{d}{dx}log(I_{sf}(x)) \tag{2.15}$$

$$\Delta(x) \begin{cases} = 0: & \text{diffuse} \\ \neq 0: & \text{specular or color discontinuity} \end{cases} \tag{2.16}$$

$$(\Delta r > th_R \text{ and } \Delta g > th_R) = \begin{cases} true: & \text{color discontinuity} \\ false: & \text{otherwise} \end{cases} \tag{2.17}$$

Finally, the specular pixels intensities are reduced until there are only diffuse only pixels as described above. For a more detailed implementation description, please reference Tan's [11] paper.

2.2 Shen et al's Chromaticity Based Specular Reflection Algorithm

Shen et al. [10] use the chromaticity of the image to select an appropriate body color for each pixel. This is done by first classifying all as diffuse or combined(specular and diffuse), then matching combined pixels to diffuse pixels having sufficiently close chromaticity. Once this is done, the percentage of specular and diffuse colors are determined by a least-squares solution, since the specular color is assumed to be white and the diffuse color is drawn from the matched pixel. This method is less computationally intensive than other approaches because it does not rely on image segmentation or local interactions.

The first step of this process is to generate a specular free (SF) image and chromaticity. This SF image preserves the geometry of the original, but can affect the colors. It is created by subtracting the minimum RGB value at each pixel from all three channels of the original image as is equation 2.18. This SF image is then made more noise robust by adding in the scalar minimum average RGB ($\bar{I}_{min}$) value as in equation 2.20. This is denoted as the modified specular free (MSF) image. Finally, the chromaticity image is found by dividing the MSF by the sum of the MSF across RGB channels at each pixel as in equation 2.21. The MSF image is useful for the initial classification of the diffuse and combined pixels while the chromaticity (c_{msf}) is used to match diffuse and combined pixels.

$$I_{sf,i}(p) = I_i(p) - min(I_r(p), I_r(p), I_b(p)), i = (r, g, b) \tag{2.18}$$

$$\bar{I}_{min} = \underset{p}{mean}(min(I_1(p), I_2(p), I_3(p)) \tag{2.19}$$

$$I_{msf,i}(p) = I_i(p) - min(I_1(p), I_2(p), I_3(p)) + \bar{I}_{min} \tag{2.20}$$

$$c_{msf,i}(p) = \frac{I_{msf,i}(p)}{\sum_i I_{msf,i}(p)} \tag{2.21}$$

The MSF image I_{MSF} is next used to determine pixels that can be used for reference body color. First the pixels are roughly classified into purely diffuse and combined candidates using equation 2.22. The threshold value th_1 is set to be $\bar{I}_{min}$. This allows the algorithm to have a baseline set of pixels with body color already determined.

$$\text{candidate} = \begin{cases} \text{diffuse} & \text{if } I_i(p) - I_{msf,i}(p) \leq th_1 \text{ for all } i \\ \text{combined} & \text{otherwise} \end{cases} \tag{2.22}$$

Finally an iterative process is undertaken to determine the percentage of specular and diffuse color at each pixel.

Step 1: Iterate while there are any unprocessed diffuse candidates. Find the pixel p with largest $I_{max}(p) = max_i I_i(p)$ amongst the unprocessed diffuse candidates and set the color of p to to be the body color I_b. For each unprocessed candidate q satisfying 2.23:

$$\sum_i \|c_{msf,i}(q) - c_{msf,i}(p)\| \leq th_2 \tag{2.23}$$

where th_2 is a chromaticity threshold, compute its reflection coefficients under the least squares criterion:

$$\begin{bmatrix} \alpha(q) \\ \beta(q) \end{bmatrix} = \begin{bmatrix} I_b(p) I_s \end{bmatrix}^{\#} I(q) \tag{2.24}$$

where the superscript "#" denotes pseudo-inverse. If $\beta(q) < 0$ recompute $\alpha(q)$ as:

$$\alpha(q) = I_b(p)^{\#} I(q) \tag{2.25}$$

after obtaining the diffuse coefficient $\alpha(q)$ from equation 2.24 or 2.25, compute the diffuse reflection component $I_{df}(q)$ and specular reflection component $I_{sp}(q)$:

$$I_{df}(q) = \alpha(q) I_b(p) \tag{2.26}$$

$$I_{sp}(q) = I(q) - \alpha(q) I_b(p) \tag{2.27}$$

finally, q is labelled as processed.

Step 2: Iterate while there are any unprocessed combined candidates. Take an unprocessed combined candidate r in a sequential manner, and find the already processed pixel p^* whose chromaticity is closest to that of pixel r:

$$p^* = \underset{i}{\operatorname{argmin}} \| c_{msf}(p) - c_{msf}(r) \| \tag{2.28}$$

For each unprocessed candidate q satisfying 2.23:

$$\sum_i \| c_{msf,i}(q) - c_{msf,i}(p^*) \| \leq th_2 \tag{2.29}$$

compute its reflection coefficients under the least squares criterion:

$$\begin{bmatrix} \alpha(q) \\ \beta(q) \end{bmatrix} = \begin{bmatrix} I_b(p^*) I_s \end{bmatrix}^{\#} I(q) \tag{2.30}$$

Note the change from p to p^* in equation 2.30.After obtaining the diffuse coefficient $\alpha(q)$ from equation 2.30 compute the diffuse reflection component $I_{df}(q)$ and specular reflection component $I_{sp}(q)$:

$$I_{df}(q) = \alpha(q) I_b(p^*) \tag{2.31}$$

$$I_{sp}(q) = I(q) - \alpha(q) I_b(p^*) \tag{2.32}$$

finally, q is labelled as processed.

In the first set of iterations, combined candidates are matched with diffuse pixels to determine their body color. If any combined pixels are left after this set of iterations, they are matched to already processed pixels. The chromaticity threshold th_2 determined to be best in this paper was 0.05.

2.3 Results and Conclusions from Chromaticity

Though both of these methods initially showed promise, neither produced acceptable results. A few of the test images can be seen below. This did however lead to a few important observations and conclusions.

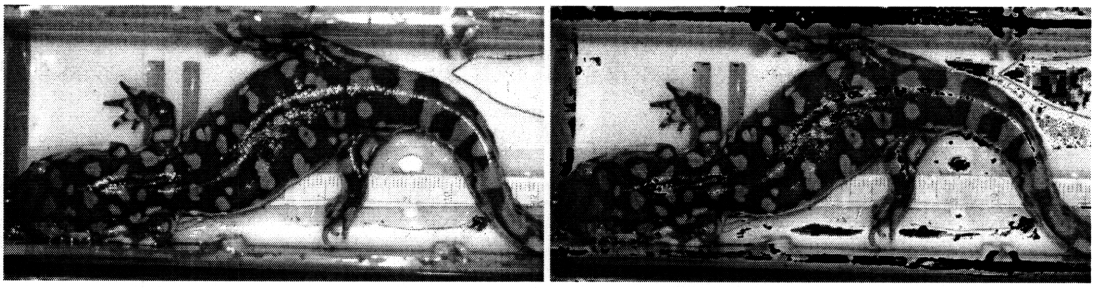

Figure 2-3: Results of Shen Chromaticity Algorithm on Tiger Salamander

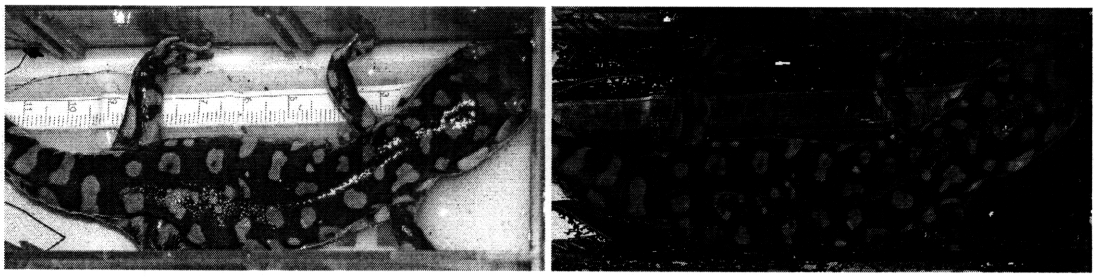

Figure 2-4: Results of Tan Chromaticity Algorithm on Tiger Salamander

Some classes of images cannot be computed under the assumptions of the above algorithms, or those used in this thesis. Images where the object is black, white or gray will fail. This is due to the fact that the objects will have no color information other than the illuminant. Since virtually all single image algorithms rely on color

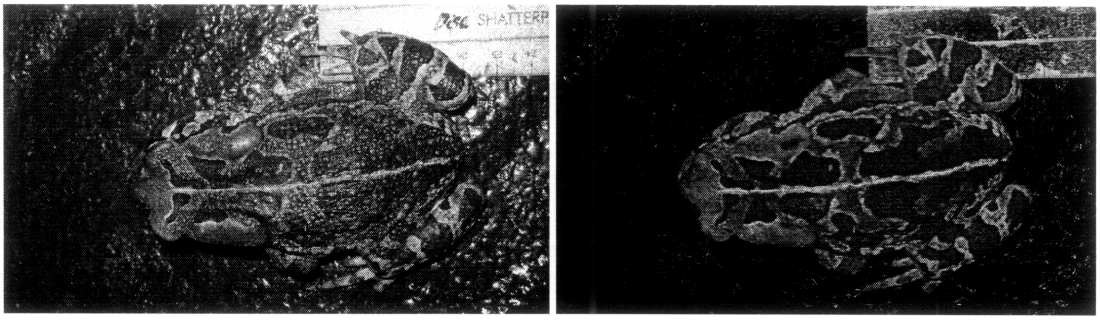

Figure 2-5: Results of Shen Chromaticity Algorithm on Toad

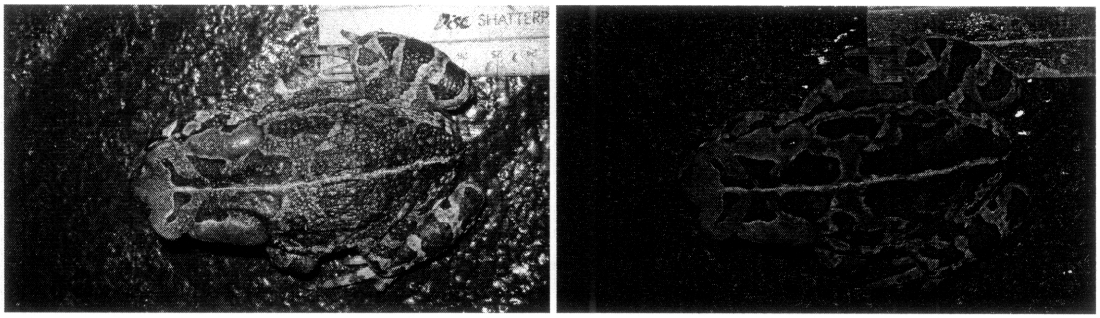

Figure 2-6: Results of Tan Chromaticity Algorithm on Toad

information from the object to separate specularity, these images cannot be used. Two examples are shown below.

Saturated pixels cannot be computed with these algorithms. A saturated pixel is defined as one with each R,G, and B channel at its maximum value. Since these pixels are basically pure white, they have no diffuse component and no chromaticity. Both algorithms described above fail to compute any useful value for these pixels.

2.4 Image Inpainting

Filling saturated regions with diffuse color require a different approach than the chromaticity methods described above. This observation lead to research in the areas of image inpainting. Bertalmio et al. [1] were the first to suggest inpainting as a method for replacing areas that were damaged or obscured in an image. Their approach uses diffusion and flow to spread color into the unknown regions of an image. This method performs well with linear structures, such as lines and contours, but produces blurry

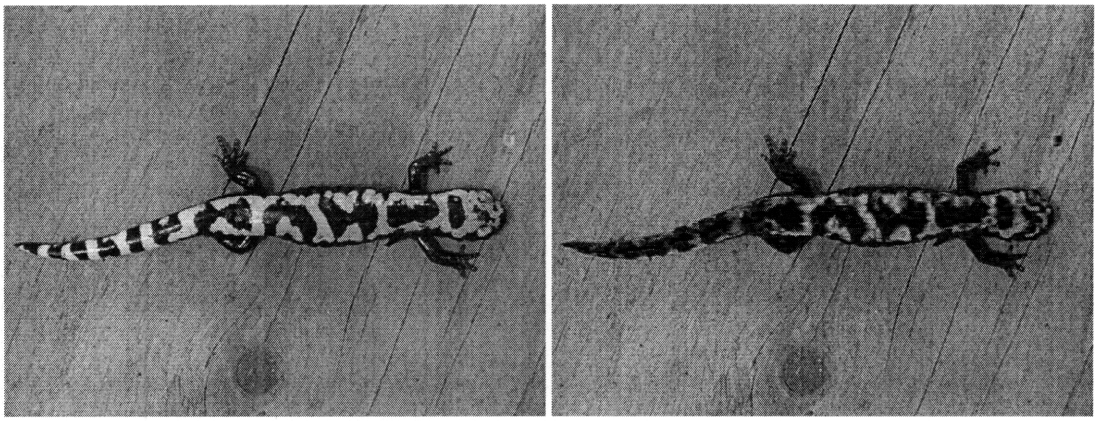

Figure 2-7: Results of Shen Chromaticity Algorithm on Marbled Salamander Showing Poor Results for Grayscale Object

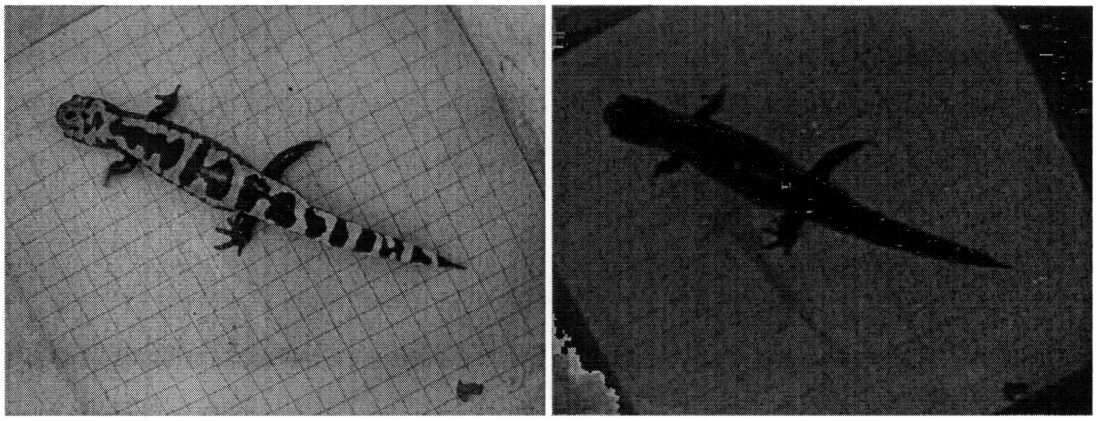

Figure 2-8: Results of Tan Chromaticity Algorithm on Marbled Salamander Showing Poor Results for Grayscale Object

results when filling larger regions.

Exemplar based inpainting, proposed by Criminisi and Prez [2], is the synthesis of two classes of algorithms for filling holes in digital images: those which synthesize texture to fill large regions, and inpainting techniques for filling small gaps. The former works well when there is a consistent repeating texture that can plausibly fill the target region and inpainting helps to maintain non-uniform structure. Ideally, this approach combines the best of both methods and works best on a large variety of images and target regions.

Chapter 3

Glare Reduction Method

Following the ideas and conclusions from the previous chapter, this chapter outlines a process for automatically identifying areas of glare using the intensity and saturation of the image. This generates a rough mask that must be modified to improve accuracy and limit processing time. After these areas are identified, they are filled with patches and texture from the diffuse parts of the image using exemplar based inpainting. [2]

3.1 Mask Generation

Followed the idea of Tan [11] that the saturation and intensity of the image characterizes the specular component of the image, I consulted several papers which further investigated this approach. First I examined a paper by Zhang et al. [14] that focused on the elimination of specular reflections assuming controlled light sources and metallic objects. While their research was insightful, I was interested in expanding it to address unknown light sources and regions of more complex texture. For a more thorough background in specularity, I consulted Shafer's work on using color to separate reflection components [9]. From this point, I was able to construct the following process of identifying regions of specular reflection

The first step required in processing the image saturation and intensity values was to covert the image to the HSV color space. This transformation produces 3 channels: Hue, Saturation, and Value. Next, two sets of pixels were identified, those in the value

channel greater than the intesity threshold I_{thresh} and those in the saturation channel less than the saturation threshold S_{thresh}. The intersection of these two sets was then used to generate a starting specularity mask. This is explained in equations 3.1 through 3.4. To facilitate further modifications to the mask, the resulting binary image was saved as a png file for later use.

$$[H, S, V] = hsv(I) \tag{3.1}$$

$$A = \{p : V(p) > I_{thresh}\} \tag{3.2}$$

$$B = \{p : S(p) < S_{thresh}\} \tag{3.3}$$

$$\text{mask pixels } = A \cap B \tag{3.4}$$

3.2 Background Removal

In this step, all identified specular regions in the background of the image were removed. This process was designed to both reduce the computation time for each individual image and remove background regions mistakenly identified as specular. For the purposes of these experiments, all modifications were completed in Adobe Photoshop on the generated pngs.

3.3 Algorithmic Mask Modifications

On certain images, the generated mask did not sufficiently cover the specular components of the image, so further masking was required. Since the mask generated in the previous steps was binary, the morphological dilation operator was the logical choice for expanding the mask. Dilation of a region A by a structuring element B is defined as the set of points covered by B as B moves inside A. A structuring element

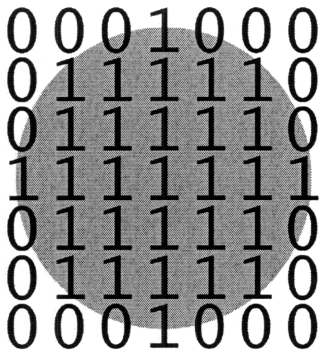

Figure 3-1: Disk structuring element with 3 pixel radius

is a binary matrix such as the in figure 3-1. The result of morphological dilation is generally the expansion of all non-zero regions in the image as shown in figure 3-2. Structuring elements can be chosen based on the size and shape of the region not covered by the mask. For this step, the modified png from the previous step was imported and dilated then saved for further processing.

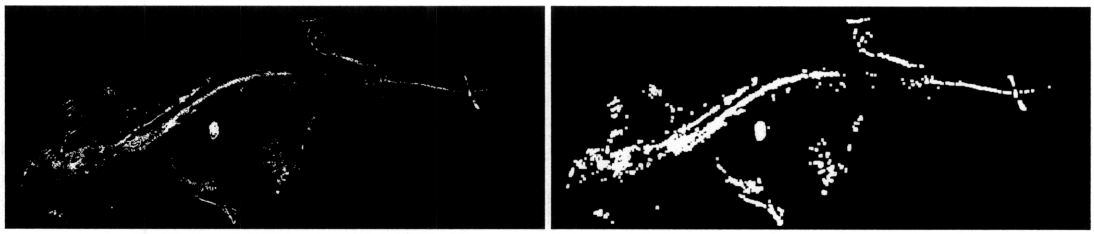

Figure 3-2: Original and dilated mask

3.4 Exemplar Based Inpainting

In order to understand the region filling process[2] , some notation is described graphically in figure 3-3. The user selects a target region, noted as Ω and by providing the contour around it $\delta\Omega$. The source region, Φ, provides the samples used throughout the process and may be defined as the image minus the source region $I - \Omega$ or another region defined by the user or a collection of images. Next the template window Ψ must be specified. The default size for this window is 9x9 pixels, but this can be set by the user. Once these parameters are defined, the algorithm proceeds automatically.

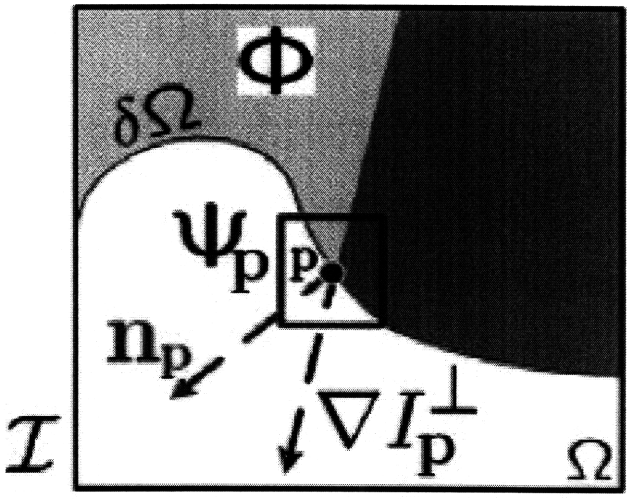

Figure 3-3: Notation diagram. Given the patch Ψ_p, n_p is the normal to the contour $\delta\Omega$ of the target region Ω and $\nabla I_p^{\perp}$ is the isophote (direction and intensity) at point p. The entire image is denoted with I. [2]

At each pixel, the algorithm must maintain a color value and a confidence value in the color value determined. Pixels along the fill front are temporarily given a priority value to determine the order they are filled. The algorithm iterates in three steps described below until all pixels are filled.

3.4.1 Step 1: Computing patch priorities

Exemplar based inpainting proceeds via a best-first filling algorithm that depends on the priority values assigned to each patch on the fill front. Given a patch Ψ_p centered at point p for some $p \in \delta\Omega$ its priority $P(p)$ is defined as:

$$P(p) = C(p)D(p) \tag{3.5}$$

where $C(p)$ is the confidence term and $D(p)$ the data term defined as:

$$C(p) = \frac{\sum_{q \in \Psi_p \cap \Omega} C(q)}{|\Psi_p|} \tag{3.6}$$

$$D(p) = \frac{|\nabla I_p^{\perp} \cdot n_p|}{\alpha} \tag{3.7}$$

where $|\Psi_p|$ is the area of Ψ_p, α is a normalization factor, and n_p is a unit vector orthogonal to the front $\delta\Omega$ at the point p. During initialization, the function $C(p)$ is set to $C(p) = 0\ \forall p \in \Omega$, and $C(p) = 1\ \forall p \in I - \Omega$.

3.4.2 Step 2: Propagating texture and structure information

Once all priorities on the fill front have been computed, the patch Ψ_p with highest priority is found. The algorithm then fills it with data extracted from the source region Φ. The source region is searched for the patch most similar to Ψ_p. Formally:

$$\Psi_q = \arg\min_{\Psi_q \in \Omega} d(\Psi_p, \Psi_q) \tag{3.8}$$

where the distance $d(\Psi_a, \Psi_a)$ between two patches is defined as the sum of squares difference (SSD) between the already filled pixels in the two patches. Having found the source exemplar Φ_q, the value of each pixel p to be filled is copied into its corresponding position in Ψ_p.

3.4.3 Step 3: Updating confidence values

After a patch Ψ_p has been filled with new pixel values, the confidence value for each newly update pixel is computed as follows:

$$\mathrm{C(q)} = \mathrm{C(p)}\ \{\forall p \in (\Psi_p \cap \Omega)\} \tag{3.9}$$

which allows the measure of relative confidence on the fill front. As filling proceeds, the confidence values will decay as the algorithm will be less sure of color values near the center. This is illustrated in figure 3-4.

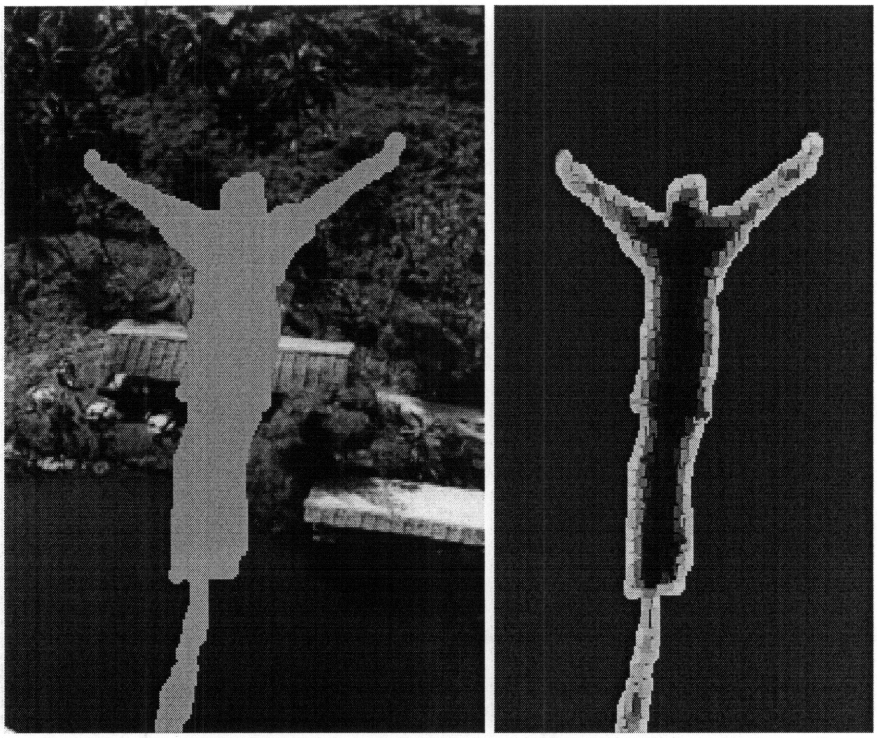

Figure 3-4: Original Image with Masked Region, Confidence Values for Filling (Red is most confident, Blue is least)

Chapter 4

Results and Discussion

This section will outline the specific results of each step of the method described above as well as some specific input choices made in the experimental process. Below is a table of chosen parameters for each tested image.

filename	dilation SE	window size	saturation threshold	intensity threshold
lizard1	3 pix disk	7x7	0.3	0.65
lizard2	3 pix disk	7x7	0.3	0.65
tigersal1	3 pix disk	7x7	0.3	0.65
tigersal2	3 pix disk	7x7	0.3	0.65
tigersal3	3 pix disk	7x7	0.3	0.65
toad1	no dilation	7x7	0.3	0.65
toad2	no dilation	7x7	0.3	0.65
toad3	nodilation	7x7	0.3	0.65

4.1 Mask Generation

The first step in producing reasonable results from the mask generation step was to decide on thresholds for intensity and saturation. Zhang et al. [14] suggest an image specific threshold, but good results were achieved using a fixed intensity threshold of 0.65 and a saturation threshold of 0.3. An example of the mask generated by this thresholding is shown in figure 4-1.

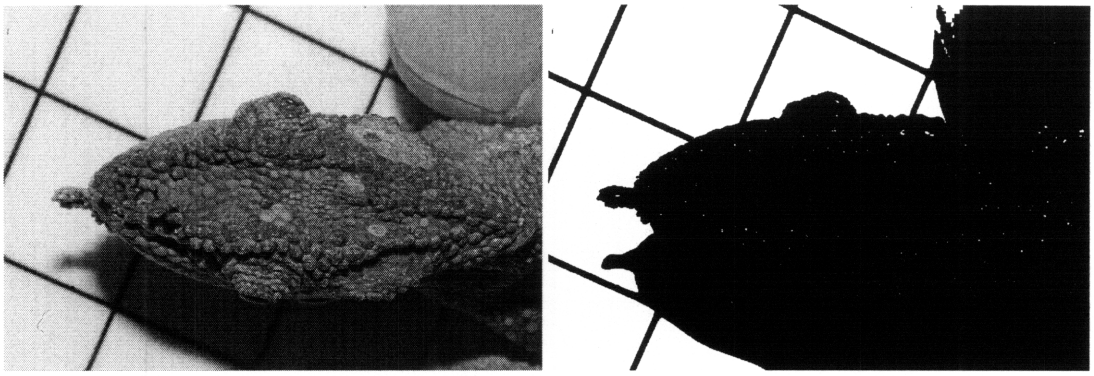

Figure 4-1: Original Image with Intensity and Saturation Mask (lizard1)

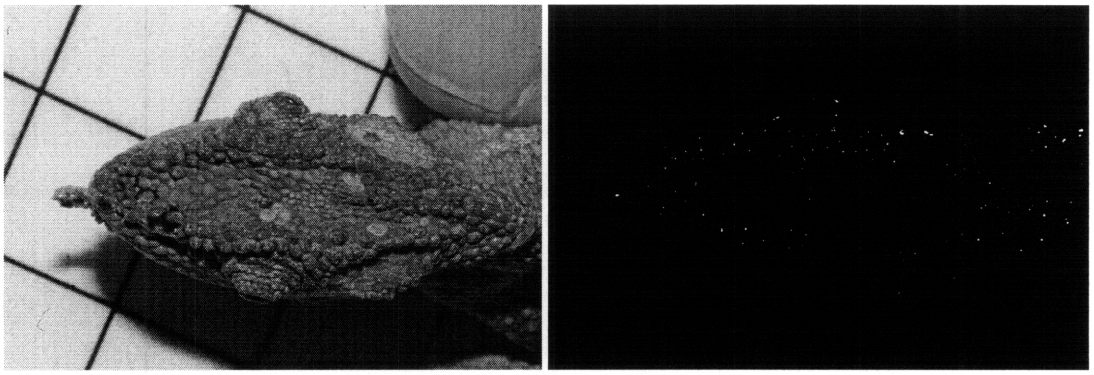

Figure 4-2: Original Image with Intensity and Saturation Mask Removing Background (lizard1)

4.2 Background Removal

In the previous example, much of the masked region is background that is not relevant to the researchers. Since this region will be ignored by the researchers and recognition tools, it is reasonable to exclude it from the mask. Figure 4-2 was worked on for about one minute in Adobe Photoshop to remove the background. This process would benefit from automatic background segmentation.

4.3 Algorithmic Mask Modifications

In this step, I experimented with several structuring element shapes and sizes for the dilation. To compare the results, it is not sufficient to examine the masks. It

Figure 4-3: Original Image, Inpainted Image from Original Mask, Inpainted Image from Dilated Mask (toad3)

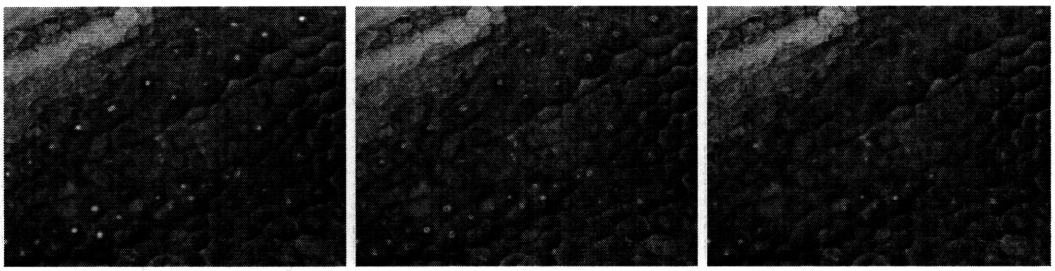

Figure 4-4: Original Image, Inpainted Image from Original Mask, Inpainted Image from Dilated Mask (lizard2)

is more important to look at the inpainted results. Some images, such as figure 4-3 worked best with no dilation. Others, such as figure 4-4 left noticeable borders around the inpainted regions without dilating. This was solved by using a three pixel disk structuring element. Also tried were a two pixel disk and three pixel diamond. The two pixel disk was not large enough to cover the entire specularity. The diamond tended to be slightly more inaccurate when considering borders as shown in figure 4-5.

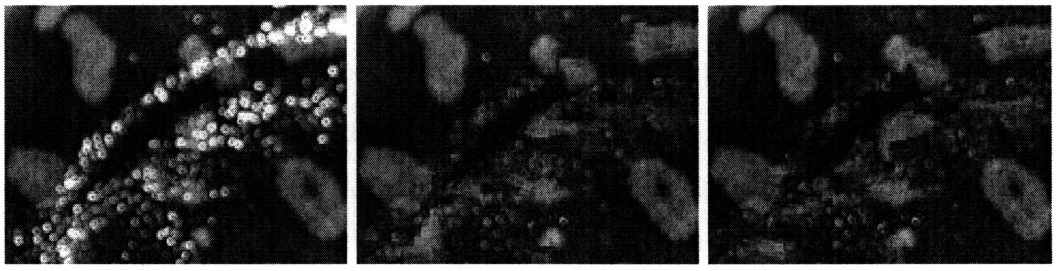

Figure 4-5: Original Image, Inpainted Image using Disk Dilation, Inpainted Image using Diamond Dilation (tigersal2)

Figure 4-6: Original Image, Inpainted Image 7x7 Window, Inpainted Image using 5x5 Window (tigersal1)

Figure 4-7: Original Image, Inpainted Image using Full Source Region, Inpainted Image using Dilated Source Region (toad3)

4.4 Exemplar Based Inpainting

Several important design choices were considered when implementing the exemplar based inpainting. First, the default 9x9 pixel window was used for all previous images, then 7x7 and 5x5 windows. Unsurprisingly, border completion accuracy tended to improve as the window size went down, but computation time increased substantially. Computing the 5x5 window inpaint took twice as long as the 7x7 and four times as long as the 9x9. Also once the 5x5 window size was reached, it tended to introduce artifacts no found in the large window sizes as shown in figure 4-6.

Second, the source region had to be chosen. Originally, this was chosen to be the entire image minus the mask, but this produced some incorrect fills as shown in figure 4-7. This was solved by choosing a source region that was a 10 pixel disk dilation of the mask region minus the mask.

Finally, with appropriately chosen mask dilations, a 7x7 window size and a dilated source region, this method produces good results as shown in figures 4-8, 4-9, and

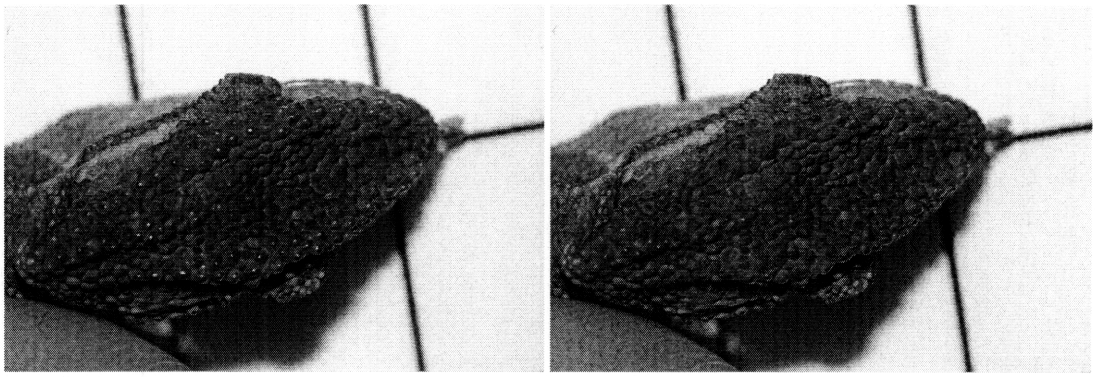

Figure 4-8: Original Image, Inpainted Image using 3 pixel disk dilation for the mask (lizard2)

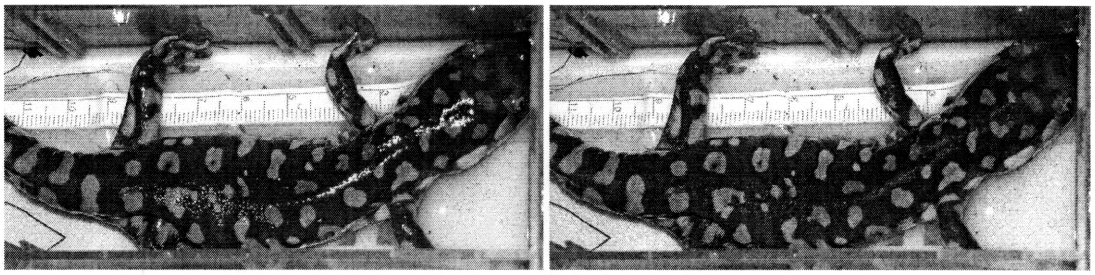

Figure 4-9: Original Image, Inpainted Image using 3 pixel disk dilation for the mask (tigersal3)

4-10.

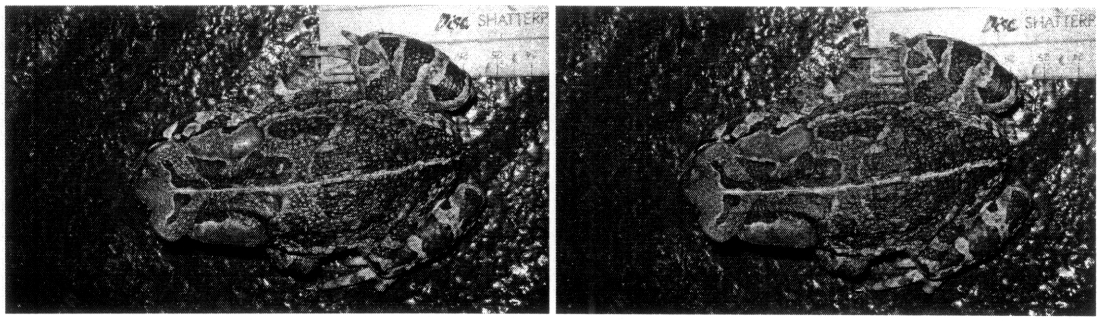

Figure 4-10: Original Image, Inpainted Image using no dilation for the mask (toad3)

Chapter 5

Conclusions and Directions for Future Work

As was shown in the previous section, this method shows significant promise in reducing specular reflections in a wide variety of images. It performs more accurately and robustly than any method I have tested assuming it is given the proper inputs. However, it is not without problems or areas for growth. In this section I will outline these problem areas, discuss my solution and offer some suggestions for future work.

5.1 Mask Generation

The first major area I see for work is in the initial mask generation. Now we generate a mask that covers all extremely specular areas, but leaves some less specular regions uncovered. It would be useful to generate a more complete mask, but the limitation of a binary mask make this nearly impossible. In order to cover all partially specular pixels, the thresholds for the mask would also include many diffuse pixels and therefore destroy much more image data. This limitation of a binary mask is imposed by the current inpainting method. If this were replaced by another inpainting tool that allowed for alpha masks, we could possibly generate a much more accurate initial mask. One paper that has touched on this approach is Mallick and Zickler's [7] PDE based approach. They use a color space transformation to determine how far each

pixel is from the illuminant color. This could be used as a alpha mask for diffusion based filling approaches.

The second major area of work for the mask would be in segmenting the foreground and background. Currently this is done manually, but could be done more automatically using established segmentation algorithms. Some images with complex backgrounds would still have to be worked on manually, but this would greatly reduce the time spent by researchers on most images.

The third major area would be to automate the selection of the dilation structuring element. Currently, the process requires trial and error to find the right size and shape. An element that works well for one image can perform extremely poorly on another. This problem is one of feature size recognition. It would be most useful to be able to extract the average feature size of the specularity and use that to produce the structuring element. Feature size selection has its own set of problems. First, one must identify relevant features. Second, one must find a way to measure their size, and third, one must translate that to the selection of the structuring element. This process alone could take up a significant amount of research.

5.2 Inpainting

Exemplar based inpainting performed very well in the cases where there was a well defined specular spot on the image that didn't cross many borders. Like all infilling methods, it had more difficulty with regions that crossed color boundaries. It performed worst when the entire specular region was not completely identified and it captured some of the specular color to inpaint. In order to solve this problem, either a better mask must be generated or a different inpainting approach must be used. If we were to use an approach based on diffusion and flow, we could also use a mask that incorporated some of the uncertainty in selecting specular regions. More infilling could occur where the algorithm was most sure that the image was specular and less where it was unsure. Unfortunately, this approach would bring the problem of blurring common to diffusion based infilling.

Another interesting problem with the exemplar based infilling was the inaccurate color copying when using the whole image. This should not have been an issue, if anything a larger source region should have allowed for more varied infilling patches an better results. This will have to be investigated in any extension of this work. If this problem could be solved, it could be useful to consider extending the available source patches. A database of images and patches would allows the algorithm a much broader pallete to inpaint the image and hopefully closer matching results. In this approach, there would certainly be some speed concerns, but more intelligent matching heuristics could be used to shorten the search time.

5.3 Conclusion

In all, this work shows promise in reducing specular reflections in a broad range of images. Through there are areas for future research and improvement, the framework is sound. Hopefully it will provide a valuable tool to researchers looking to reduce glare in their current collection of images.

Bibliography

[1] Marcelo Bertalmio, Guillermo Sapiro, Vicent Caselles, and Coloma Ballester. Image inpainting. In Kurt Akeley, editor, *Siggraph 2000, Computer Graphics Proceedings*, pages 417–424. ACM Press / ACM SIGGRAPH / Addison Wesley Longman, 2000.

[2] A. Criminisi, P. Pérez, and K. Toyama. Object removal by exemplar-based inpainting. In *2003 Conference on Computer Vision and Pattern Recognition (CVPR 2003)*, pages 721–728, June 2003.

[3] Lloyd Gamble, Sai Ravela, and Kevin McGarigal. Multi-scale features for identifying individuals in large biological databases: an application of pattern recognition technology to the marbled salamander ambystoma opacum. *Journal of Applied Ecology*, 45:170–180(11), February 2008.

[4] H.-C. Lee. Method for computing the scene-illuminant chromaticity from specular highlights. *Journal of the Optical Society of America A*, 3:1694–1699, October 1986.

[5] Sang Wook Lee. *Understanding of surface reflections in computer vision by color and multiple views.* PhD thesis, University of Pennsylvania, Philadelphia, PA, USA, 1992.

[6] Sang Wook Lee and Ruzena Bajcsy. Detection of specularity using color and multiple views. In *ECCV '92: Proceedings of the Second European Conference on Computer Vision*, pages 99–114, London, UK, 1992. Springer-Verlag.

[7] Satya P. Mallick, Todd Zickler, Peter N. Belhumeur, and David J. Kriegman. Specularity removal in images and videos: A pde approach. In Ales Leonardis, Horst Bischof, and Axel Pinz, editors, *ECCV (1)*, volume 3951 of *Lecture Notes in Computer Science*, pages 550–563. Springer, 2006.

[8] S K Nayar, X S Fang, and T Boult. Separation of reflection components using color and polarization. *International Journal of Computer Vision*, (21):163–186, 1997.

[9] S. A. Shafer. Using color to separate reflection components (A). *Journal of the Optical Society of America A*, 1:1248–+, 1984.

[10] Hui-Liang Shen, Hong-Gang Zhang, Si-Jie Shao, and John H. Xin. Chromaticity-based separation of reflection components in a single image. *Pattern Recogn.*, 41(8):2461–2469, 2008.

[11] Robby T. Tan. Separating reflection components of textured surfaces using a single image. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(2):178–193, 2005. Fellow-Katsushi Ikeuchi.

[12] T.T. Tan, K. Nishino, and K. Ikeuchi. Illumination chromaticity estimation using inverse-intensity chromaticity space. *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, 1:I–673–I–680 vol.1, June 2003.

[13] Wikipedia. Hsl and hsv — wikipedia, the free encyclopedia, 2009. [Online; accessed 23-January-2009].

[14] Zhong Zhang, Shiqing Ren, Miyake Tetsuo, Hisanaga Fujiwara, and Takashi Imamura. Reducing specular reflection components of metallic surfaces using photometric linearization. In *ICICIC '06: Proceedings of the First International Conference on Innovative Computing, Information and Control*, pages 121–125, Washington, DC, USA, 2006. IEEE Computer Society.