FACE ILLUMINATION NORMALIZATION WITH SHADOW CONSIDERATION

By

Avinash B. Baliga

SUBMITTED IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE
AT
CARNEGIE MELLON UNIVERSITY
5000 FORBES AVENUE PITTSBURGH PA 15213-3890
MAY 2004

CARNEGIE MELLON UNIVERSITY

DEPARTMENT OF

ECE

The undersigned hereby certify that they have read and recommend to the Faculty of Graduate Studies for acceptance a thesis entitled "**Face Illumination Normalization with Shadow Consideration**" by **Avinash B. Baliga** in partial fulfillment of the requirements for the degree of **Master of Science**.

Dated: May 2004

Supervisor: ______________________________
Prof. Tsuhan Chen

Readers: ______________________________
Prof. Tsuhan Chen

____
Prof. Vijayakumar Bhagavatula

CARNEGIE MELLON UNIVERSITY

Date: **May 2004**

Author: **Avinash B. Baliga**

Title: **Face Illumination Normalization with Shadow Consideration**

Department: **ECE**

Degree: **M.S.** Convocation: **May** Year: **2004**

Signature of Author

Table of Contents

Abstract

Face recognition has been pursued by many researchers, but still remains a largely unsolved problem. In this paper, we present several methods for making face recognition illumination–invariant. We first present a computer vision oriented approach, where we use Shape–From–Shading to determine the albedo–integrated normal map of an input human face image. We then frontally relight the albedo–integrated normal map, and provide this normalized image to the classifier. The novel work we present is our improved lighting model, that takes both attached and cast shadows into account. We also provide a summary of existing image–processing algorithms that have historically been used for illumination normalization. We present a general normalization approach based on the luminance equation, which removes deep shadowed and specular areas implicitly. We conclude by comparing the face recognition performance of these algorithms and show that simple image–processing algorithms can provide very good illumination invariance.

Chapter 1

Introduction

Computing a three dimensional description of a scene from one or more 2–D images is a classic computer vision problem. A major class of algorithms that seeks to recover the shape of objects in a scene from one or more images is Shape–From–Shading (SFS) [1]. Typically, SFS algorithms produce surface–normal vectors and/or depth information; a collection of surface–normals and surface–depths will be called "normal–maps" and "height–maps," respectively.

SFS techniques that operate on more than one image of a subject taken from the same viewpoint, but under different illumination conditions, are called *photometric stereo* [2] techniques. In this paper, we propose an algorithm to improve photometric stereo by taking attached and cast shadows into account. In Section 2, we describe previous approaches to photometric stereo, and our contribution.

In Section 3, we present methods to overcome the effects of varying illumination on human face images, for the purpose of improving face recognition. In the area of face recognition, there are two main approaches. The computer–vision based approach attempts to represent the human face with specific models based on its physical structure. The pattern–recognition approach attempts to represent the human face with broader statistical models, often that can apply to non-face images as well. In this paper, we show that these approaches can be combined for illumination normalization of human faces, with favorable results. One method we propose is a modification of the illumination normalization algorithm by Y. Hsu and T. Chen [3] such that shadowed pixels are considered missing data.

We present a number of classic image processing algorithms in Section 4, and use them pre–processing algorithms for face–recognition. These methods include Gain/Offset correction, Histogram Equalization, Homomorphic Filters, and the family of algorithms known as Retinex. In this section, we propose a simple image–processing algorithm that provides very good performance for face recognition. Additionally, we propose a method of multi–resolution processing that combines our proposed method with any other normalization algorithm.

Note: In this paper, scalar quantities (e.g. b, ρ) are italicized, vector quantities (e.g. $\mathbf{n}$, $\mathbf{c}$) are in bold, matrices (e.g. $\mathbf{A}$, $\mathbf{\Sigma}$) are in capitalized bold, and functions (e.g. $e(x, \mathbf{s})$) are italicized. The dimensions of a vector or matrix will often be parenthesized in its subscript to add clarity (e.g. $\mathbf{A}_{(M\times N)}$, $\mathbf{\Phi}_{(d\times 3)}$).

Chapter 2

Photometric Stereo

2.1. Prior Work: Woodham

Horn and Brooks originally proposed a surface–normal recovery technique based on the assumption that all surfaces are Lambertian [1]. Woodham [2] introduced photometric stereo, a technique that assumes all surfaces are Lambertian, and uses many images of the same surface taken under different illumination conditions to reconstruct the surface. Lambert's law can be expressed as:

$$b = \max(\rho\, \mathbf{n}^{\mathrm{T}}\mathbf{s},\, 0) \tag{2.1}$$

where b is the reflected light intensity, ρ is the surface albedo (also referred to as 'texture'), $\mathbf{n}_{(3\times1)}$ is the surface–normal vector, and $\mathbf{s}_{(3\times1)}$ is a vector that points to the light source. Since the $\max$ function makes Lambert's law non–linear, and thus less analytically tractable, the *reflectance equation* used in most SFS–related literature is:

$$b = \rho\, \mathbf{n}^{\mathrm{T}}\mathbf{s} \tag{2.2}$$

This reflectance equation can be extended to the case where $M >= 3$ input images of the scene are given, and each image is lit by a single point light–source located at infinity (each with a different lighting–direction vector). The extended reflectance equation (for a single pixel) is written as:

$$\mathbf{b}^{\mathrm{T}} = \rho\,\mathbf{n}^{\mathrm{T}}\mathbf{S} \tag{2.3}$$

$$\mathbf{b}^{\mathrm{T}} = \mathbf{r}^{\mathrm{T}}\mathbf{S} \tag{2.4}$$

where $\mathbf{b}_{(M\times 1)}$ is the vector containing the light intensities of a single location in the scene under each illumination, ρ is the surface albedo (constant across illumination) , $\mathbf{n}_{(3\times 1)}$ is the surface–normal vector (constant across illumination), and $\mathbf{S}_{3\times M}$ is the matrix containing the illumination–direction of each image in its columns. Since both ρ and $\mathbf{n}$ are independent of the illumination, they can be combined into a single term $\mathbf{r}_{(3\times 1)}$, the *albedo–integrated normal*. The least–squares solution can be shown to be:

$$\mathbf{r}^{\mathrm{T}} = \mathbf{b}^{\mathrm{T}}\mathbf{S}^{\dagger} \tag{2.5}$$

where $\mathbf{S}^{\dagger}$ is the pseudo-inverse of $\mathbf{S}$:

$$\mathbf{S}^{\dagger} = \mathbf{S}^{\mathrm{T}}\left(\mathbf{S}\mathbf{S}^{\mathrm{T}}\right)^{-1} \tag{2.6}$$

Computing $\mathbf{r}^{\mathrm{T}}$ for every point in the image yields an albedo–integrated normal–map; this albedo–integrated normal–map can be relit synthetically to yield a scene under arbitrary illumination conditions.

2.2. An Improved Illumination Model

All of the above discussion of SFS assumed the Lambertian reflectance equation — Eqn. (2.2) — holds for all input images. However, three important phenomena can upset the results determined from the Lambertian model significantly: ambient light, shadows and speculars. In this paper, we focus on ways to detect and eliminate the effects of shadows in SFS. This section explains the effects of shadows on the albedo–integrated normal–map, and our method of handling shadows when we are provided with multiple input images (and their corresponding light–direction vectors).

There are two types of shadows – attached and cast (see Fig. 2.1). Attached shadows occur when the surface normal points away from the light source; cast shadows occur when one surface prevents light from reaching another. Now consider the LS estimation of the surface–normal of a given pixel. The surface normal will be “steered towards” the light–direction vectors corresponding to the brightest pixels, and “pushed

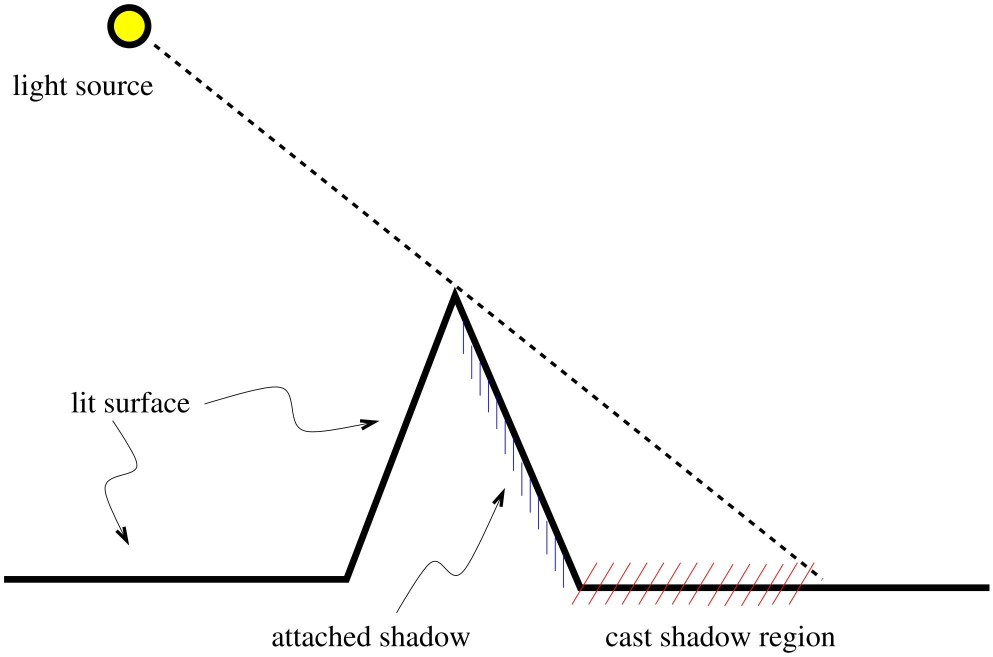

Figure 2.1: Illustration of Shadow Types

away" from the light–direction vectors corresponding to the darker pixels. When pixels under attached shadow (for a given illumination direction) have a non–zero intensity in an image (often caused by inter–reflections), then the surface–normal will be incorrectly steered closer to that light–direction. When pixels under cast shadows occur, then the surface–normal will be incorrectly pushed away from the corresponding light–directions.

To determine any shadow information, we first generate the albedo–integrated normal–map using the LS estimate (Eqn. 2.5). Now we can find some of the attached shadows in the input images simply from Lambert's law, rewritten here:

$$b = \max(\rho\,\mathbf{n}^{\mathrm{T}}\mathbf{s},\, 0) \tag{2.7}$$

Whenever $\mathbf{n}^{\mathrm{T}}\mathbf{s} < 0$, the surface normal points away from the light source, so we have found an attached shadow. Note that this is an imperfect estimate of all the attached shadows, since the albedo–integrated normal–map estimate was derived from the reflectance equation 2.2 which does not take into account any shadows.

To determine cast shadows, we employ ray–tracing from the surface of the face towards the light–source. Thus, there is an inherent assumption that the surface of the subject is smooth and that occluding surfaces are visible in the input images, which is a reasonable assumption for many useful tasks, such as the surface recovery of human faces, manufactured objects, or landscapes. To perform the ray–tracing, we propose the use of a fast DDA approach (Differential Drawing Algorithm), which can be thought of as an extension of Bresenham's line algorithm [22]. Many people have used DDA approaches to ray–tracing height–fields for the purposes of rendering; see [23] for a similar approach. Essentially, we trace a "light–ray" from the pixel in question towards the light source, updating its height at every pixel we traverse; simultaneously, we integrate along the surface normals to find the heights of the pixels that the light–ray traverses. If the height of the surface at any pixel is greater than the height of the light–ray, then the pixel in question is under a cast shadow. An alternate approach to this method is to pre–integrate the normal–map (yielding a height–map), rather than integrating along the normal–map when ray–tracing. Since the LS estimate of the surface–normals does not guarantee integrability, better results are obtained from pre–integrating the surface using a stable approach; in this paper, we generated surfaces by integrating along 256 directions and averaged these surfaces together for the final height–map.

Now that we are able to determine attached and cast shadows, we use an iterative process to improve the initial LS estimate of the albedo–integrated normal–map. In each iteration, the shadowed pixels are detected, and are subsequently excluded from the LS estimate in the next iteration. The overall approach is outlined below:

1. Estimate the lighting direction vectors for each input image.
2. Estimate the albedo–integrated normal–map of the image $\mathbf{R}$ from input images $\mathbf{B}_1 \ldots \mathbf{B}_M$
3. For each input image $\mathbf{B}_i$
   - For each pixel $\mathbf{B}_{i,j}$, determine whether that pixel is in either attached or cast shadow in this iteration; attached shadows are determined by comparing the current surface normal with the light direction vector, and cast shadows are determined by performing ray–tracing
   - If pixel $\mathbf{B}_{i,j}$ is in shadow, then mark this pixel as non-Lambertian in this iteration
4. Recompute the LS estimate for the albedo–integrated normal–map $\mathbf{R}$, using only pixels not marked as non-Lambertian for this iteration
5. If solution has not converged, goto 3

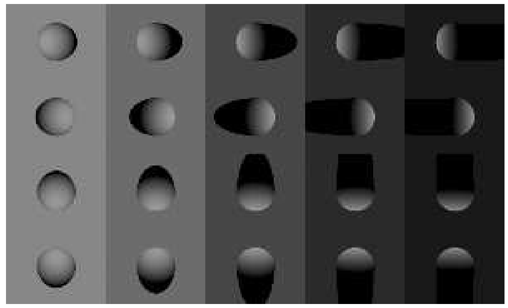

Figure 2.2: Synthetic Input Images

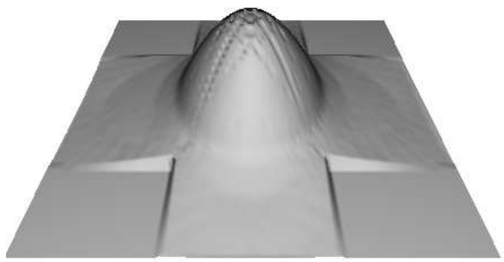

Figure 2.3: Result from original SFS algorithm – Eqn. (2.5)

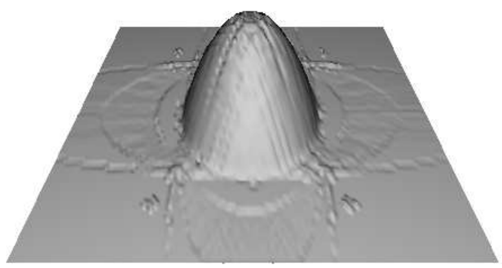

Figure 2.4: Result from proposed method; notice the improved areas around the hemisphere.

We have found that this approach works well on synthetic and human face images. The synthetic example in Figures 2.2, 2.3, 2.4 illustrates our algorithm's effect nicely. Figure 2.2 is a 5×4 grid of images of a hemisphere protruding from a plane, rendered with Povray [24] (an additional image taken with frontal illumination, not shown, was also used in the experiment). Figure 2.3 shows the height–map that was recovered by the pixel–wise LS estimate (Eqn. 2.5). Notice the ramps surrounding the hemisphere – these are caused by the cast shadows in the input images. In contrast, Figure 2.4 shows the result of our algorithm; the regions surrounding the hemisphere are obviously much more accurate. Human face results can be seen in Section 5.

Chapter 3

Face Recognition

3.1. Prior Work: Sim and Kanade

Sim and Kanade's goal was illumination invariance for face–recognition; for this task, typically only one input image is provided for a given subject. To deal with this limitation, they used a bootstrap set of images to model the statistics of the albedo–integrated surface–normal $\mathbf{r}$ and the fitting error e at each pixel

$$e = b - \mathbf{r}^{\mathrm{T}}\mathrm{s} \tag{3.1}$$

as independent Gaussian distributions. Assuming that the face images were all aligned and scaled, their model assumed that the albedo–integrated surface–normal at a given point on a human face fell within a Gaussian distribution, independent of the neighboring pixels. They computed the statistics of $\mathbf{r}$ and e for each pixel using the PIE database [25], and used the statistics of $\mathbf{r}$ to compute a Maximum-A-Posteriori estimate of the albedo–integrated surface–normals of an input image. Sim and Kanade claimed that the reconstruction error term e encapsulated some of the non–Lambertian effects on images (shadows, speculars), so they used the statistics of this term to emulate these effects on reconstructed images. Once the albedo–integrated normal–map for the input image was computed, Sim and Kanade relit it with a large number of illumination conditions, in order to span the illumination space of the image. In this way, many images of each subject would be presented to the classifier for the gallery (training set).

3.2. Illumination Normalization Using PCA

In contrast with Sim and Kanade's pixel–wise approach to collecting statistics, the way Y. Hsu collected the statistics of human faces was by generating a PCA space of the albedo–integrated surface–normals [3]. These statistics are used to compute the albedo–integrated normal–map of the input image, so that the image can be synthetically relit with frontal illumination. Then only frontally–lit images are provided to the classifier, for both the gallery and probe (test) sets.

To describe this method, we begin by defining a column vector containing the d–pixel input image $\mathbf{b}_{test\ (d\times 1)}$ and the albedo–integrated normal–map for an entire image as:

$$\mathbf{R}_{(d\times 3)} = \begin{bmatrix} (\rho n_{1x}) & (\rho n_{1y}) & (\rho n_{1z}) \\ (\rho n_{2x}) & (\rho n_{2y}) & (\rho n_{2z}) \\ & \vdots & \\ (\rho n_{dx}) & (\rho n_{dy}) & (\rho n_{dz}) \end{bmatrix}_{(d\times 3)} \tag{3.2}$$

We define the x– and y–directions to be in the image plane, and the z–direction as the height–direction for surface points. For notational convenience, $\mathbf{R}$ is put into vector form $\hat{\mathbf{r}}$, which is defined as:

$$\hat{\mathbf{r}}_{(3d\times 1)} = \begin{bmatrix} (\rho n_{1x}) \\ (\rho n_{1y}) \\ (\rho n_{1z}) \\ (\rho n_{2x}) \\ (\rho n_{2y}) \\ (\rho n_{2z}) \\ \vdots \\ (\rho n_{dx}) \\ (\rho n_{dy}) \\ (\rho n_{dz}) \end{bmatrix}_{(3d\times 1)} \tag{3.3}$$

The albedo–integrated normal–maps are assumed to be Gaussianly distributed, i.e.

$$\hat{\mathbf{r}} \sim N(\boldsymbol{\mu}_{\hat{r}}, \boldsymbol{\Sigma}_{\hat{r}}) \tag{3.4}$$

This means we can write:

$$\hat{\mathbf{r}} = \boldsymbol{\mu}_{\hat{r}} + \mathbf{H}_{\hat{r}}\mathbf{c} \tag{3.5}$$

where $\boldsymbol{\mu}_{\hat{r}\ (3d\times 1)}$ is the mean albedo–integrated normal–map, $\mathbf{H}_{\hat{r}\ (3d\times q)}$ is the matrix containing the eigenvectors corresponding to the q largest eigenvalues of $\boldsymbol{\Sigma}_{\hat{r}\ (3d\times 3d)}$ in its columns, and $\mathbf{c}_{(q\times 1)}$ is the vector of PCA coefficients (also known as the *spectrum* of the sample). In other words, $\mathbf{H}_{\hat{r}\ (3d\times q)}$ is a PCA space for faces. The statistics $\boldsymbol{\mu}_{\hat{r}}$ and $\mathbf{H}_{\hat{r}}$ are collected from a (preferably very large, representative) training set.

Since the goal is face illumination normalization on single input images, we wish to reconstruct the image's albedo–integrated normal–map and project it into the PCA space. However, the LS solution in Equation (2.5) is not useful in the single image case, because the matrix $\mathbf{S}\mathbf{S}^{\mathrm{T}}$ will be non–invertible. Thus, the PCA space is used to compute a LS estimate of $\mathbf{c}$. This is accomplished by first estimating the light–direction vector $\mathbf{s}_{est}$. Lighting directions were estimated by pixel–wise comparing the input image to training images and computing a weighted sum; this light direction estimation technique was used by Sim and Kanade in [10]. When we ran this algorithm, we found that the estimated lighting direction vectors did not vary from the original direction vectors by more than 10% in the worst cases. Once the light–direction vector has been estimated, the normal–map can be estimated as well. Let us define the matrix $\mathbf{S}_{est}$ as the Krönecker product of $\mathbf{s}_{est}$ and the identity matrix $\mathbf{I}_{(d\times d)}$:

$$\mathbf{S}_{est} = \begin{bmatrix} \mathbf{s}_{est} & & & \mathbf{0} \\ & \mathbf{s}_{est} & & \\ & & \ddots & \\ \mathbf{0} & & & \mathbf{s}_{est} \end{bmatrix}_{(3d\times d)} \tag{3.6}$$

Now Eqn. (2.5) can be rewritten as:

$$
\begin{aligned}
\mathbf{b}_{test} &= \mathbf{S}_{est}^{\mathrm{T}}\hat{\mathbf{r}} &(3.7)\\
&= \begin{bmatrix} \mathbf{s}_{est}^{\mathrm{T}}\hat{\mathbf{r}}_{1..3} \\ \mathbf{s}_{est}^{\mathrm{T}}\hat{\mathbf{r}}_{4..6} \\ \mathbf{s}_{est}^{\mathrm{T}}\hat{\mathbf{r}}_{7..9} \\ \vdots \\ \mathbf{s}_{est}^{\mathrm{T}}\hat{\mathbf{r}}_{(3d-2)..(3d)} \end{bmatrix} &(3.8)
\end{aligned}
$$

Substituting Eqn. (3.5) into Eqn. (3.7) results in:

$$
\begin{aligned}
\mathbf{b}_{test} &= \mathbf{S}_{est}^{\mathrm{T}}\boldsymbol{\mu}_{\hat{r}} + \left(\mathbf{S}_{est}^{\mathrm{T}}\mathbf{H}_{\hat{r}}\right)\mathbf{c} &(3.9)\\
\left(\mathbf{b}_{test} - \mathbf{S}_{est}^{\mathrm{T}}\boldsymbol{\mu}_{\hat{r}}\right) &= \left(\mathbf{S}_{est}^{\mathrm{T}}\mathbf{H}_{\hat{r}}\right)\mathbf{c} &(3.10)
\end{aligned}
$$

Now we can take the LS estimate of $\mathbf{c}$, which yields:

$$
\begin{aligned}
\mathbf{c} &= \left(\mathbf{S}_{est}^{\mathrm{T}}\mathbf{H}_{\hat{r}}\right)^{\dagger}\left(\mathbf{b}_{test} - \mathbf{S}_{est}^{\mathrm{T}}\boldsymbol{\mu}_{\hat{r}}\right) &(3.11)\\
&= \left(\mathbf{H}_{\hat{r}}^{\mathrm{T}}\mathbf{S}_{est}\mathbf{S}_{est}^{\mathrm{T}}\mathbf{H}_{\hat{r}}\right)^{-1}\left(\mathbf{H}_{\hat{r}}^{\mathrm{T}}\mathbf{S}_{est}\right)\left(\mathbf{b}_{test} - \mathbf{S}_{est}^{\mathrm{T}}\boldsymbol{\mu}_{\hat{r}}\right) &(3.12)
\end{aligned}
$$

It can be shown [3] that the least–squares solution for $\mathbf{c}$ is equivalent to the Maximum–A–Priori solution.

Now that a method of generating $\mathbf{c}$ from a single face image has been described, $\mathbf{c}$ can be plugged into Eqn. (3.5) to generate the albedo–integrated normal–map $\hat{\mathbf{r}}$ for the input image. Each input image is relit frontally before passing the image to the classifier.

In Figure 3.1 we show images of the Z–components of each albedo–integrated normal–vector in the mean and first 20 eigenvectors; the images have been biased (to 128), so that negative values are viewable as dark values. Since we are generating frontally lit images of input faces, we simply take the Z–component of each surface–normal of $\hat{\mathbf{r}}$ to get the pixel values; this means that the final output images are effectively linear combinations of these Z–component images. For the purpose of visualization, we also show the 3–D surface of the mean and the first seven eigenvectors in Figure 3.2. Note that the surfaces constructed from the eigenvectors do not have a physical meaning themselves; only linear combinations of the eigenvectors (which are albedo–integrated normal–maps) are used to reconstruct the surfaces of input images.

3.3. Further Improving Face Recognition by Considering Shadows

In this section we explain our expanded approach to normalizing single input images. The approach is a combination of the methods described in Sections 3.2 and 2.2 — that is, we find the shadowed pixels and disclude them from the estimate of the albedo–integrated surface–normals. However, since we only have a single input image, we consider all the shadowed pixels to be missing data. We can generate a weight vector $\mathbf{w}$ containing 0's in the elements corresponding to the pixels where shadows are found, and 1's in the remaining elements. Thus, we are motivated to use a weighted–SVD approach (see [26] for a thorough discussion). The way we use weighted–SVD is: given an existing estimate for the albedo–integrated normal–map $\hat{\mathbf{r}}$ and a weight vector $\mathbf{w}_{3d\times 1}$ which indicates the presence of shadows (with zero values), we wish to find $\mathbf{c}_M$ by solving the minimization problem:

$$\arg\min_{\mathbf{c}_M} \; J(\mathbf{c}_M) = \parallel \mathbf{W}(\hat{\mathbf{r}} - (\boldsymbol{\mu}_{\hat{r}} + \mathbf{H}_{\hat{r}}\mathbf{c}_M)) \parallel \tag{3.13}$$

where $\mathbf{c}_M$ is the vector of PCA coefficients we wish to obtain, $\mathbf{W}_{3d\times 3d} = diag(\mathbf{w})$ is the matrix containing the elements of $\mathbf{w}$ in its diagonal, and the remaining terms are the same as in Eqn. (3.12). The LS solution is:

$$\mathbf{c}_M = (\mathbf{H}_{\hat{r}}^{\mathrm{T}}\mathbf{W}^2\mathbf{H}_{\hat{r}})^{-1}\,\mathbf{H}_{\hat{r}}^{\mathrm{T}}\,(\mathbf{W}^2)\,\;(\hat{\mathbf{r}} - \boldsymbol{\mu}_{\hat{r}}) \tag{3.14}$$

We begin the illumination normalization by estimating the albedo–integrated normal–map of the input image using the method described in Section 3.2; that is, we start by computing the LS estimate of the albedo–integrated normal–map $\hat{\mathbf{r}}$ without considering shadows. We then use this initial albedo–integrated normal–map to generate a height–map, and locate attached and cast shadows using the methods described in Section 2.2. The current albedo–integrated normal–map and the weight–vector are used with Eqn. (3.14) to compute a new albedo–integrated normal–map. The process of finding shadows and using that information to refine the estimate of the albedo–integrated normal–map is repeated until the solution converges (i.e. when the change in reconstruction error between iterations reaches a threshold).

An outline of our iterative approach is as follows:

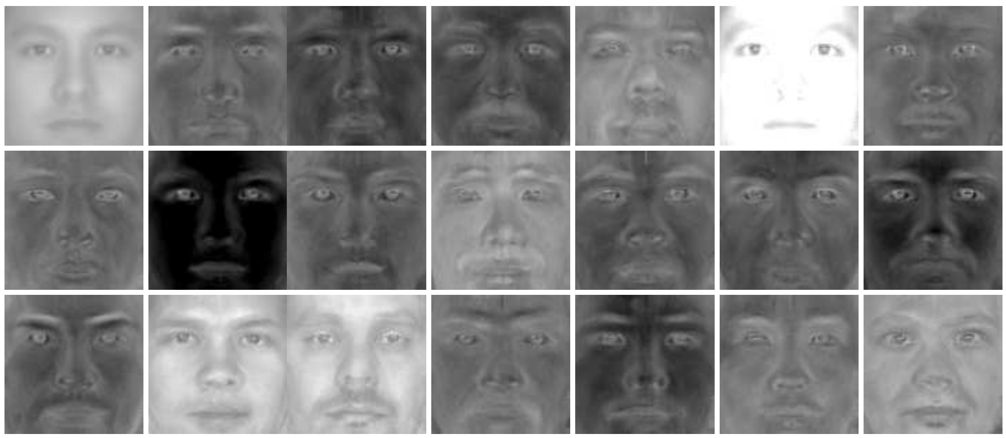

Figure 3.1: Z-Components of Mean (Upper Left) and First 20 Eigenvectors

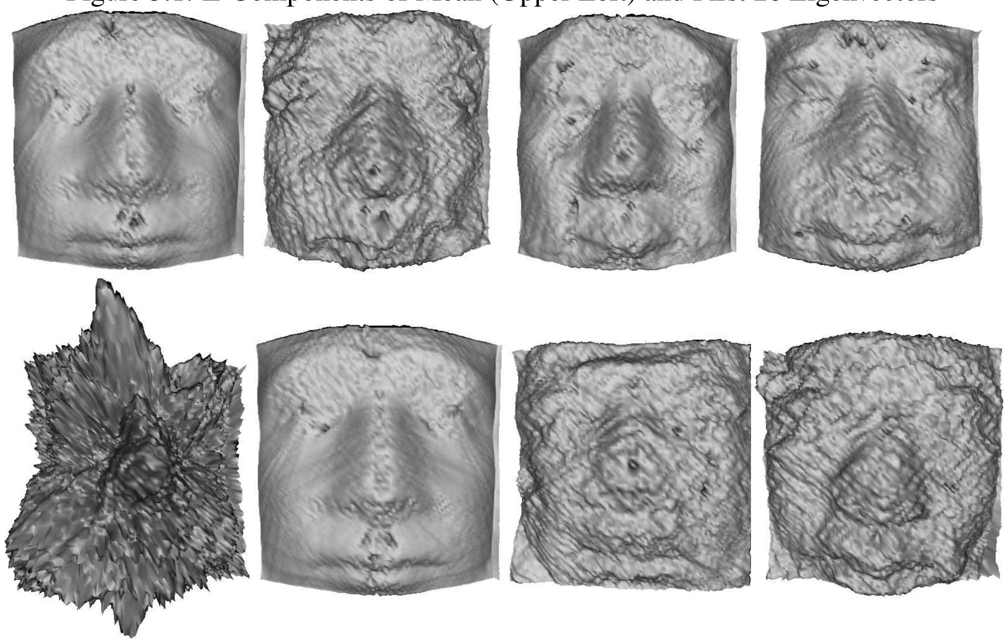

Figure 3.2: Surface of Mean (left) and First 7 Eigenvectors

1. Estimate the lighting direction vector for the input image.

2. Estimate the albedo–integrated normal–map of the input image $\hat{\mathbf{r}}$ by using Eqn. (3.12) and Eqn. (3.5); that is, use the PCA–space to construct an initial estimate of the geometry

3. Compute $\mathbf{w}$ by finding attached and cast shadows

4. Compute the LS estimate for $\mathbf{c}_M$ using Eqn. (3.14)

5. Recompute $\hat{\mathbf{r}}$ using $\mathbf{c}_M$ in Eqn. (3.5)

6. If the solution has not converged, goto 3

3.4. Improving Robustness by Considering Ambient Light

When we introduced the Lambertian model, we made the assumption that scenes were lit by a single point light source, and the algorithms described up until this point rely on the images having no ambient light. However, in practice face images are taken under a variety of illumination conditions. Therefore, we propose a simple method for removing ambient light from input images, so that input images better match the lighting model that the previous sections assume.

A popular model for ambient light is an additive term scaled by the surface albedo, so the new lighting model (for a given pixel i) becomes:

$$b_i = \rho_i \, s_a + \rho_i \, \mathbf{n}_i^{\mathrm{T}} \mathbf{s} \tag{3.15}$$

where s_a is the constant ambient term. Notice that s_a is assumed to be the same for every pixel in the scene.

Since we collect the prior statistics of face images in the training stage of our normalization algorithm, we can extract the mean albedo from the mean albedo–integrated normal–map of the PCA space by taking the magnitude of each albedo–integrated normal–vector. We now formulate an expression for b_i', the set of pixel values that correspond to the b_i's but do not contain ambient light.

$$\begin{aligned} b_i' &= b_i - \mu_{\rho_i} \, s_a && (3.16)\\ &= \rho_i \, s_a + \rho_i \, \mathbf{n}_i^{\mathrm{T}} \mathbf{s} - \mu_{\rho_i} \, s_a && (3.17) \end{aligned}$$

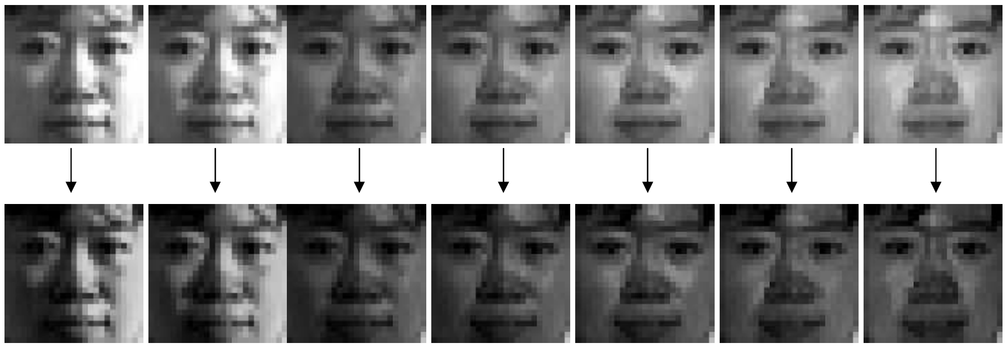

Figure 3.3: Images 2 through 8 of PIE Subject 0, applying ambient light removal to images that were illuminated with ambient light

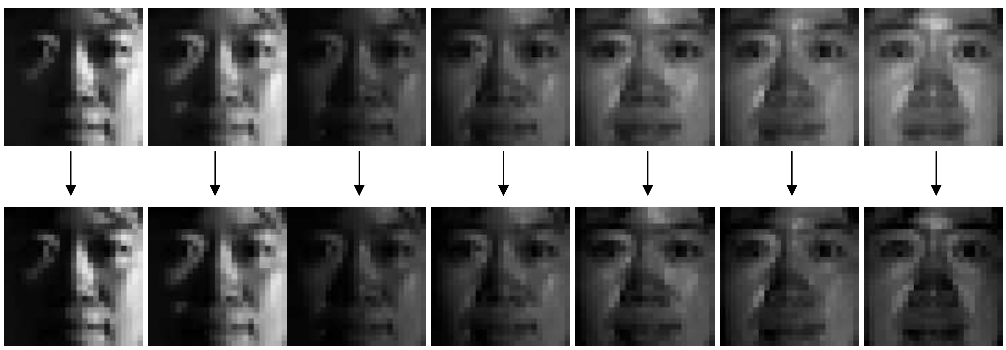

Figure 3.4: Images 2 through 8 of PIE Subject 0, applying ambient light removal to images that were taken without ambient light

where μ_{ρ_i} is the mean–albedo for the i'th pixel. Using this approximate model, we can search for a value of s_a that best represents the ambient light present in the image. Our method considers s_a optimal when a certain percentage of the b_i''s fall near zero. We experimentally determined that a good value for this threshold is close to 3%.

To show that this method works well, we have included example images of PIE subject 0. Figure 3.3 shows the result of our ambient light removal method when applied to PIE images that were taken with ambient light; Figure 3.4 shows the result when applied to images that were taken without ambient light. Notice that after processing the images with ambient light, the results look very similar to the original PIE images without ambient light. Also note that after processing the images without ambient light, the results do not differ much from the original images.

Chapter 4

Image–Processing Algorithms

In this section, we present a number of image–processing algorithms that perform contrast enhancement. Gain/offset correction and histogram equalization are classic algorithms that make the input image span the entire dynamic range of the output channel (8–bit greyscale in this paper). Retinex is an algorithm by E. Land [30], that was designed to emulate the human visual system. The remaining algorithms that we discuss (Homomorphic Filtering, Single–Scale Retinex, Multi–Scale Retinex, and the proposed method) all assume the same reflectance model, which can be described with the reflectance equation.

The reflectance equation, defined for each pixel at location (x, y), is:

$$B(x, y) = R(x, y) \cdot I(x, y) \tag{4.1}$$

where $B(x, y)$ is the image, $R(x, y)$ is the perceived reflectance, and $I(x, y)$ is the perceived illumination. Thus, to perform illumination normalization one simply divides the $I(x, y)$ term off of $B(x, y)$, the pixel value. Several proposals have been made for determining I, with varying complexity.

4.1. Gain/Offset Correction

The goal of this method is to make the signal occupy the full dynamic range of the channel. This method relies on assuming an underlying probability density function of the pixel intensities. If the underlying pdf

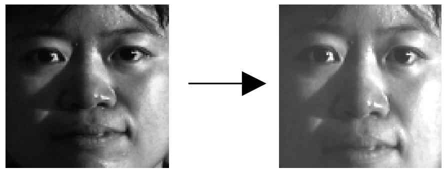

Figure 4.1: Gain/Offset Correction (Gaussian) applied to a PIE image illuminated by only a single point light source

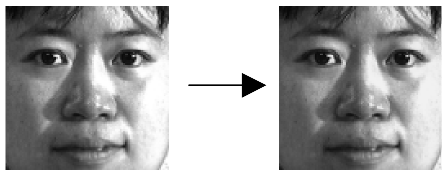

Figure 4.2: Gain/Offset Correction (Gaussian) applied to a PIE image illuminated by point light source and ambient light

is assumed to be uniform, then each output pixel $b_{out}(x, y)$ is computed as

$$b_{out}(x, y) = \frac{255}{b_{\max} - b_{\min}} \cdot b_{in}(x, y) \tag{4.2}$$

where $b_{\max}$ is the largest pixel value in the input image, $b_{\min}$ is the smallest pixel value in the input image, and b_{in} is the input image. Naturally, this method is very sensitive to noise and image content.

If the underlying pdf is assumed to be Gaussian, then the goal is to transform the image such that the mean of the pixel intensities is 128, and the standard deviation of the pixel intensities is $\frac{128}{\alpha}$. Here α is a parameter that determines the dynamic range of the output image; we chose to set $\alpha = 2$ since two standard deviations capture 95% of the pixels. The resulting expression for each pixel is

$$b_{out}(x, y) = \frac{128}{\alpha \cdot \sigma_{in}} \cdot (b_{in}(x, y) - \mu_{in}) + 128 \tag{4.3}$$

where μ_{in} and σ_{in} and the mean and standard deviation of the input image b_{in}. Figures 4.1 and 4.2 show examples of this algorithm applied to PIE images without and with ambient light.

4.2. Histogram Equalization

The goal of histogram equalization is to transform the input image in such a way that the output image has a uniform pdf. To accomplish this, the image's pdf must be approximated first, which can be done by taking a

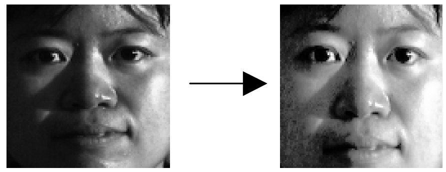

Figure 4.3: Histogram Equalization applied to a PIE image illuminated by only a single point light source

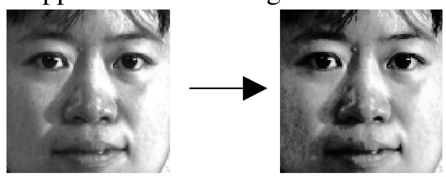

Figure 4.4: Histogram Equalization applied to a PIE image illuminated by point light source and ambient light

histogram. Once the pdf is obtained, the cdf can be computed; the inverse cdf provides the mapping between input pixel values and output pixel values. Figures 4.3 and 4.4 show examples of this algorithm applied to PIE images without and with ambient light.

4.3. Classic Retinex

Retinex (a combination of the words retina and cortex) was originally proposed by E. Land and J. McCann [30] as an algorithm that models the human visual system. Although its original purpose was for color constancy, it performs well as a contrast–enhancement algorithm too. For our experiments we used code by B. Funt et al [31], which implements the Frankle–McCann variation on the Retinex algorithm.

The Retinex algorithm works by operating on the image pixels in the log domain. An output image buffer is created and initialized to all ones. Then for each pixel, a random path is followed, with a "sequential product" computed along the way (the sequential product is intialized to 1 for each path). Four operations are performed along the path, when each neighbor is visited: *(1)* the ratio of the current input pixel (on the path) to the previous input pixel is computed, *(2)* that ratio is multiplied with the sequential product, *(3)* if the sequential product is greater than a certain threshold it is reset to 1, and *4* the sequential product is averaged with the current output pixel. Depending on the lengths of the paths and the threshold, different amounts of contrast can be achieved (global versus local). Figures 4.5 and 4.6 show examples of this algorithm applied to PIE images without and with ambient light.

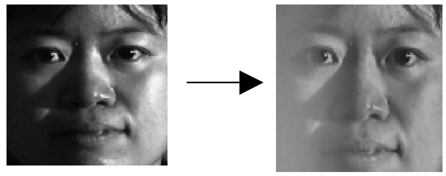

Figure 4.5: Classic Retinex applied to a PIE image illuminated by only a single point light source

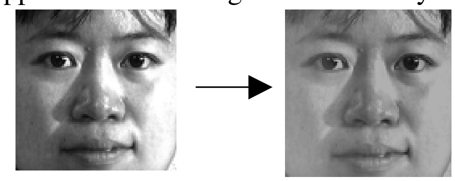

Figure 4.6: Classic Retinex applied to a PIE image illuminated by point light source and ambient light

4.4. Homomorphic Filtering

Homomorphic filtering is often used in medical imaging. Homomorphic filters start by taking the log of each pixel, to enhance the contrast in dark regions and reduce contrast in bright regions. Thus, the reflectance equation becomes:

$$\log(\mathbf{B}) = \log(\mathbf{R}) + \log(\mathbf{I}) \tag{4.4}$$

The assumption with homomorphic filtering is that lighting changes slowly and smoothly across an image. Based on this assumption, a reasonable approximation for $\log(\mathbf{I})$ would be a lowpass-filtered version of the log–input image $\log(\mathbf{B})$, which we will denote $\mathrm{LPF}_{\omega_c}(\log(\mathbf{B}))$. This gives us:

$$\log(R(x,y)) = \log(B(x,y)) - [\mathrm{LPF}_{\omega_c}(\log(\mathbf{B}))](x,y) \tag{4.5}$$

An exponent is then applied to $\log(R(x,y))$ as computed in Equation 4.5 to invert the log function applied to $R(x,y)$, resulting in the final expression:

$$R(x,y) = \exp\left(\log(B(x,y)) - [\mathrm{LPF}_{\omega_c}(\log(\mathbf{B}))](x,y)\right) \tag{4.6}$$

Figures 4.7 and 4.8 show examples of this algorithm applied to PIE images without and with ambient light.

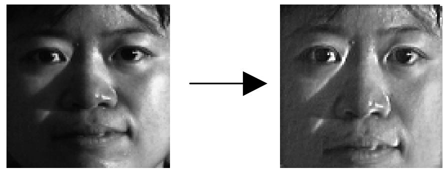

Figure 4.7: Homomorphic Filtering applied to a PIE image illuminated by only a single point light source

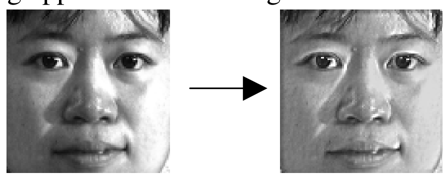

Figure 4.8: Homomorphic Filtering applied to a PIE image illuminated by point light source and ambient light

4.5. Single–Scale Retinex

Single–Scale Retinex by Z. Rahman [35] is similar to homomorphic filtering. It differs in the placement of the log function, and it does not use an exponent. The lowpass filter is also defined as an FIR filter. It is formulated as

$$R(x,y) = \log\left(B(x,y)\right) - \log\left([(\mathbf{B}\star\mathbf{H})](x,y)\right) \quad (4.7)$$

$$= \log\left(\frac{B(x,y)}{[\mathbf{B}\star\mathbf{H}](x,y)}\right) \quad (4.8)$$

Where $[\mathbf{B}\star\mathbf{H}]$ denotes the convolution of the input image $\mathbf{B}$ with the Gaussian kernel $\mathbf{H}$. Convolution with $\mathbf{H}$ effectively performs a lowpass filtering operation. After this initial step of computing $R(x,y)$, gain/offset correction is applied to make the final image viewable. For our final implementation, we used a first order IIR Butterworth filter to perform the lowpass filter, since the performance was always better than using FIR filters. Figures 4.9 and 4.10 show examples of this algorithm applied to PIE images without and with ambient light.

4.6. Multi–Scale Retinex

One very visible problem with the Single–Scale Retinex algorithm is the “halo–ing” effect at sharp boundaries. An example of halo–ing can be seen below the nose of the subject in Fig. 4.11. Z. Rahman et. al. [34]

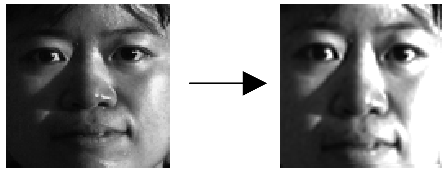

Figure 4.9: Single–Scale Retinex applied to a PIE image illuminated by only a single point light source

Figure 4.10: Single–Scale Retinex applied to a PIE image illuminated by point light source and ambient light

proposed a method to suppress this effect and maintain better color constancy, which they dubbed "Multiscale Retinex." Recall that the original Retinex construction used Gaussian impulse–response FIR filters to perform the lowpass filter. Multiscale retinex generates several lowpass images by using Gaussians of different widths (variances), and combines them with a weighted sum. For general usage, satisfactory performance is gained with equal weighting.

4.7. Perception Model with Companding and Smoothness Constraint

R. Gross and V. Brajovic [32] proposed an algorithm based on the Retinex Equation and Weber's law. Weber's law states that the perceived intensity of a region with intensity $B(x, y)$ is proportional to $\frac{\Delta B(x,y)}{B(x,y)}$; i. e. the perceived intensity varies logarithmically with the input intensity. The goal once again is to find the reflectance $R(x, y)$; this is accomplished by minimizing the objective function:

$$J(R) = \int\int \rho(x, y)(R(x, y) - B(x, y))^2 dxdy + \lambda \int\int (R_x^2(x, y) + R_y^2(x, y))dxdy \qquad (4.9)$$

Where $\rho(x, y) \sim \frac{\Delta B(x,y)}{B(x,y)}$ encodes the perception model, the entire first term causes the solution to be follow the perception model, and the second term is a smoothness constraint. This is a variational calculus problem, and the resulting Euler-Lagrange equation is:

$$R(x, y) + \frac{\lambda}{\rho(x, y)}(R_{xx}(x, y) + R_{yy}(x, y)) = B(x, y) \qquad (4.10)$$

This can be descretized onto a rectangular lattice and numerically solved with Multigrid algorithms (see [33]). [32] claims that the use of Weber's law causes the algorithm to handle boundaries in shadowed regions as well as in bright areas, since $\rho(x, y)$ measures the relative intensity of neighboring pixels. However, many of the artifacts found in Retinex are also found in the results of this algorithm.

4.8. Ratio Images

We propose a simple method of approximating the reflectance image:

$$R(x, y) = \frac{B(x, y)}{[\mathrm{LPF}_{\omega_c}(\mathbf{B})](x, y)} \tag{4.11}$$

This is similar to automatic gain–control in the spatial domain. Notice that with this method, for low–noise images R will mostly contain values close to 1, so some post–processing is required to make the image viewable. We found that applying gain/offset correction to R did not produce viewable results; this was due to the intensity of a few edges dominating the image in most scenes. However, scaling the image intensity by a fixed factor and then saturating all the pixel values greater than 255 produced very good results. The fixed factor ν is defined as $\nu = \frac{255}{\beta}$, where β is a parameter that specifies how much local contrast will be retained. For example, if $\beta = 2$, then $R(x, y)$ can take on values less than 2 without saturating. We call these resulting images "Ratio Images," not to be confused with "Quotient Images" as introduced by Riklin-Raviv et al [14].

The resulting images tend to preserve edges while turning large areas of solid colors or slow gradients into solid–colored regions (with value 1). A cutoff frequency that is very high will make the fraction approach 1, whereas a very low cutoff will make the fraction approach $\mathbf{B}(x, y)$. We found that a cutoff frequency (ω_c) between 0.1π and 0.2π worked well on face images in the PIE database. If other cutoff frequencies are chosen, we found that the performance degrades, although not by a large amount. Although FIR filters with very wide Gaussian impulse–responses were used for Single–Scale Retinex, we used first order IIR butterworth filters in our experiments for their effectiveness on small images.

Since IIR filters do not have a linear phase response, and if they are causally implemented the output is delayed, by shifting the filtered image in space before performing the divide, better results can be obtained.

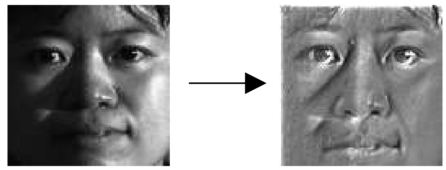

Figure 4.11: “Ratio Image” applied to a PIE image illuminated by only a single point light source

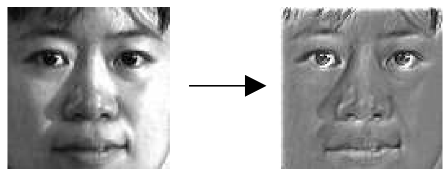

Figure 4.12: “Ratio Image” applied to a PIE image illuminated by point light source and ambient light

This can be expressed as

$$R_{\tau_x,\tau_y}(x,y) = \frac{B(x,y)}{[\mathrm{LPF}_{\omega_c}(\mathbf{B})](x-\tau_x, y-\tau_y)} \tag{4.12}$$

where (τ_x, τ_y) determine the amount to shift the lowpass image. Many reflectance approximations R_{τ_x,τ_y} can be computed by varying (τ_x, τ_y); these reflectances can be averaged together before applying the scaling and saturation.

Examples of this “Ratio Image” algorithm applied to PIE images with and without ambient lighting can be seen in Figs. 4.11 and 4.12.

4.9 Combining Algorithms

One very easy way to combine SFS and the “Ratio Image” is to perform the SFS processing at a low resolution, then use texture–mapping to restore the test image to full resolution. The texture map is generated by dividing the input image by the upsampled low–resolution image. For example, we could take a test image $\mathbf{b}_{test\ (100\times100)}$, and downsample it to $\mathbf{b}_{td\ (20\times20)}$. We apply a SFS–based normalization algorithm to $\mathbf{b}_{td}$, resulting in a frontally lit (downsampled) face we call $\mathbf{b}_{nd\ (20\times20)}$. Separately, we form the texture map $\mathbf{T}_{(100\times100)}$ by performing the operation: $\mathbf{T} = \mathbf{b}_{test}/\mathrm{LPF}_{\omega=0.2\pi}(\mathbf{b}_{test})$. Now we upsample $\mathbf{b}_{nd}$ to get $\mathbf{b}_{n\ (100\times100)}$, and perform the pixelwise multiplication $\mathbf{b}_{norm}(x,y) = \mathbf{b}_n(x,y)\cdot\mathbf{T}(x,y)$ to get our final normalized image, $\mathbf{b}_{norm}$. This is illustrated in Fig. 4.13.

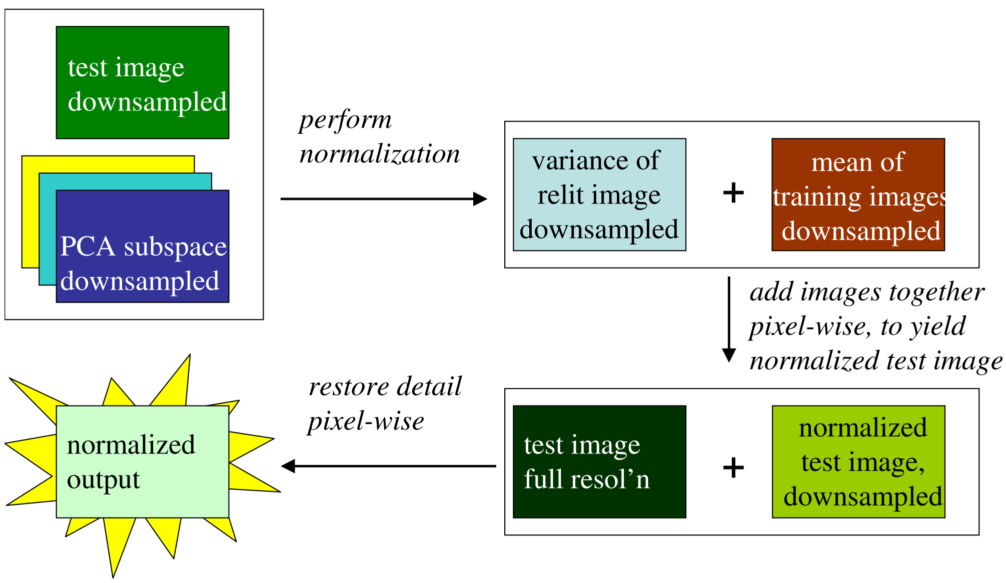

Figure 4.13: Block diagram demonstrating a combination of SFS– and Retinex–based normalization.

Chapter 5

Results

In this section, we demonstrate the performance of the algorithms described in Sections 2.2 and 3.3. We used cropped, scaled images of 64 subjects taken under 21 illumination conditions from the CMU PIE database [25] for our experiments. There are two datasets in the PIE database: images that were lit with a point light source, and images that were lit with a point light source and ambient light. We used the Torch library [27] for its Matrix routines in our implementation.

5.1. Photometric Stereo

For our first set of experiments, we show that our photometric stereo algorithm yields better reconstruction error than the LS solution in Eqn. (2.5). We measure the reconstruction error by relighting the recovered albedo–integrated normal–map with the illumination directions of each of the input images, and taking the sum of squared pixel differences (the same criteria that is being minimized by the LS solution). Comparisons were made on images from five subjects (0, 1, 6, 7, and 10) at resolutions of 33x33 and 100x100, with varying input image lighting conditions. The three input image sets were: *(a)* all 21 PIE illumination conditions [2 . . . 22] (see Fig. 5.1 for the input images for PIE subject 0), *(b)* 7 illumination conditions including from a variety of directions [2, 4, 6, 8, 10, 12, 1] (see Fig. 5.2), and *(c)* 7 illumination conditions with light sources mostly at extreme angles [2, 3, 4, 8, 15, 16, 17] (see Fig. 5.3). Figure 5.4 contains all the images of subject 6 of the PIE database, arrayed from 2 . . . 8 on the top row, 9 . . . 15 in the middle row, and 16 . . . 22 on the bottom row (Fig. 5.1 is similarly arrayed). Table (5.1) summarizes the results. Notice that we receive a substantial improvement in reconstruction error when mostly shadowy images are included

Figure 5.1: All 21 Images of Subject 0 of the PIE database

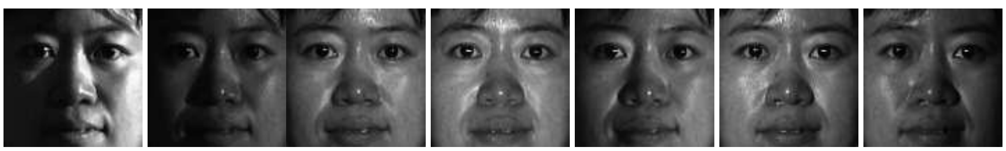

Figure 5.2: Images of PIE Subject 0 used for test case (b)

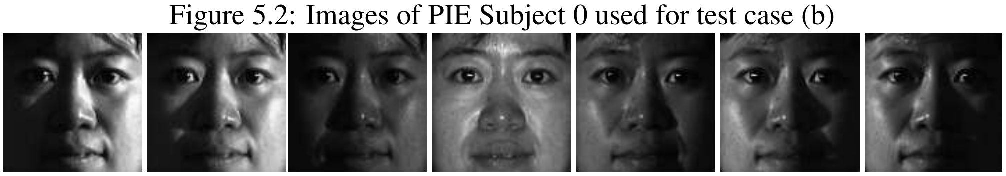

Figure 5.3: Images of PIE Subject 0 used for test case (c)

in the input set. Fig. 5.5 shows the surface (not including albedo) computed by the LS solution on the left and the proposed solution on the right, for subject 6 in the 100x100 *(a)* case from a frontal view. Fig. 5.6 shows the same surface from a profile view. The vertical lines in Fig. 5.6 project down to a plane below the face surface, to aid the viewer in judging the relative heights of parts of the surface. Notice the protruding lower face that is computed by the LS solution (Fig. 5.6, left side). The identity of the subject is preserved better by the proposed solution (Fig. 5.5), and the height of the lower half of the face is more consistently computed (Fig. 5.6). We can attribute the improved height to the LS estimation phenomenon described earlier, of shadowed pixels being "pushed away" from the light direction they were captured under. Since pixels on the side of the face are often under shadow, this tends to make their estimated surface–normals steeper than they should be; the result is the poorly determined height of the lower face area. We also show similar images of subject 0 of the PIE database in Figures 5.1, 5.7, and 5.8.

Test Case	Avg. % Improvement in Reconstruction Error.
33x33 *(a)*	7.4
33x33 *(b)*	2.92
33x33 *(c)*	22.46
100x100 *(a)*	7.16
100x100 *(b)*	1.15
100x100 *(c)*	12.87

Table 5.1: Percent Improvement of Squared Error from LS solution to Proposed solution

Figure 5.4: All 21 Images of Subject 6 of the PIE database

5.2. Face Recognition – SFS

We conducted our experiments with frontal images of 64 subjects from the PIE database, taken with the location of the flash bulb at 21 different locations, with and without ambient light (overhead fluorescent lights). The images have been scaled and cropped, and are at a resolution of 100x100. We used a Support–Vector Machine classifier [28] in the recognition tests. The Torch library [27] was used for its matrix manipulation routines. When we say that we are testing a given algorithm's performance, we mean that we are using that algorithms as a pre–processing step for both the gallery and probe images provided to the classifier.

We start by presenting the results of the Shape–From–Shading based work. For the face recognition task we used an SVM classifier [28], and compare the performance of images with no pre–processing, after normalizing using the algorithm described in Section 3.2 (referred to as SFS+PCA), and the normalizing

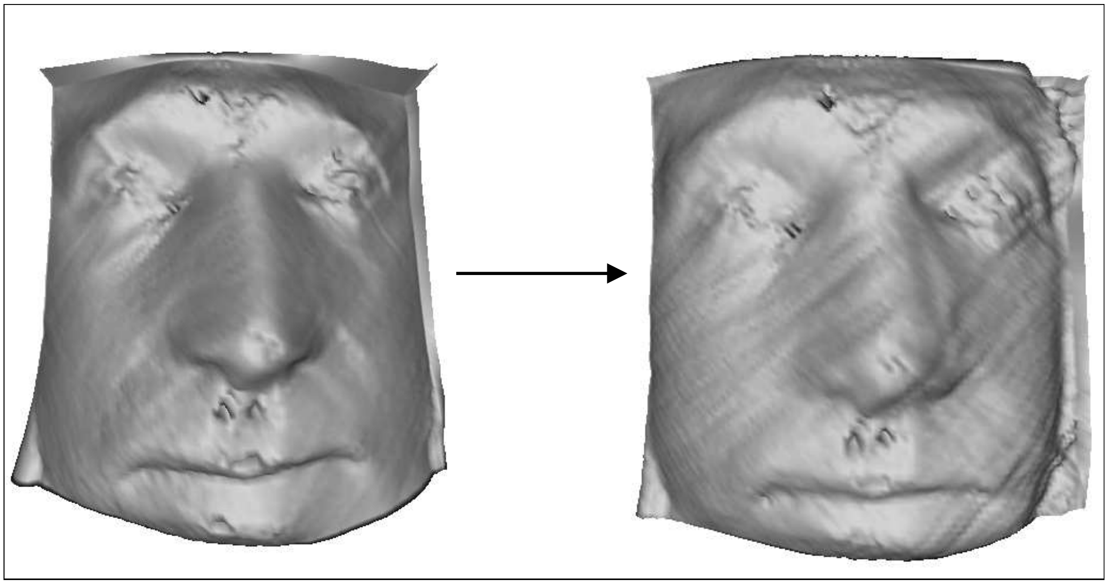

Figure 5.5: Surface of PIE Subject 6, Frontal View; LS solution (left), proposed solution (right)

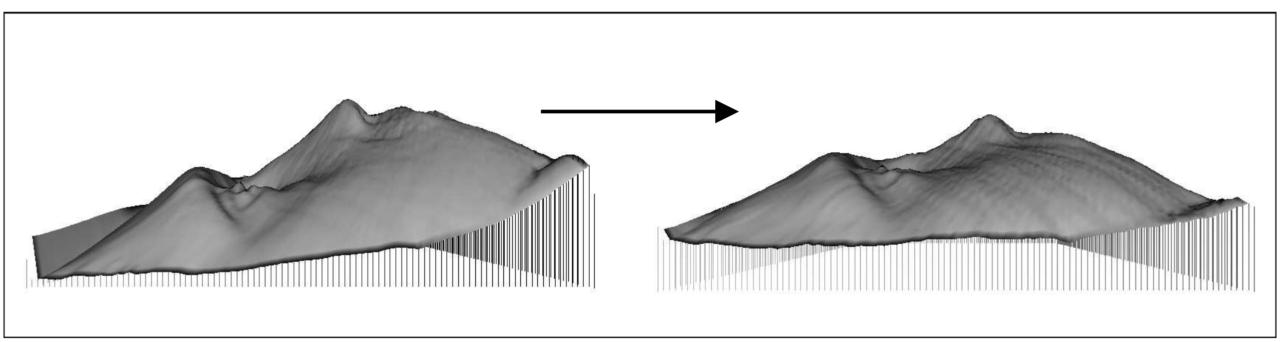

Figure 5.6: Surface of PIE Subject 6, Profile View; LS solution (left), proposed solution (right)

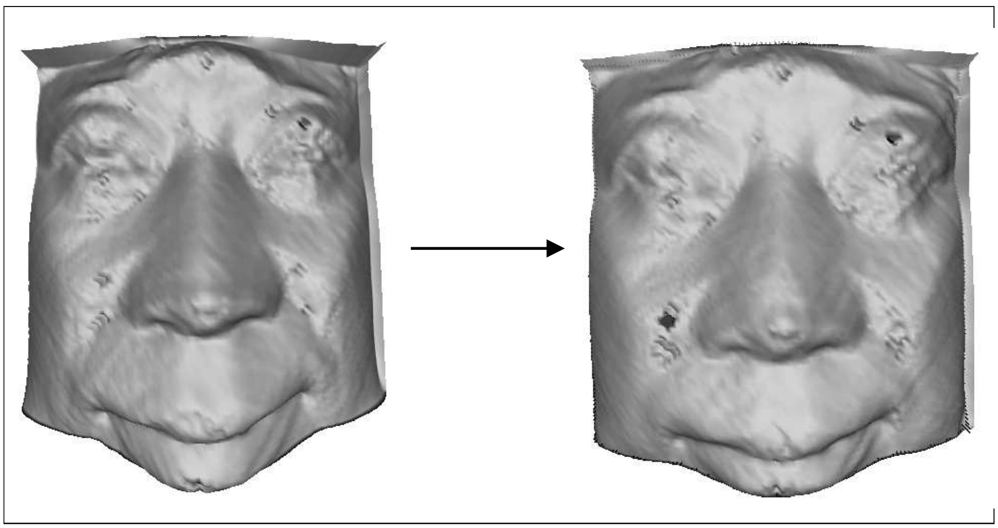

Figure 5.7: Surface of PIE Subject 0, Frontal View; LS solution (left), proposed solution (right)

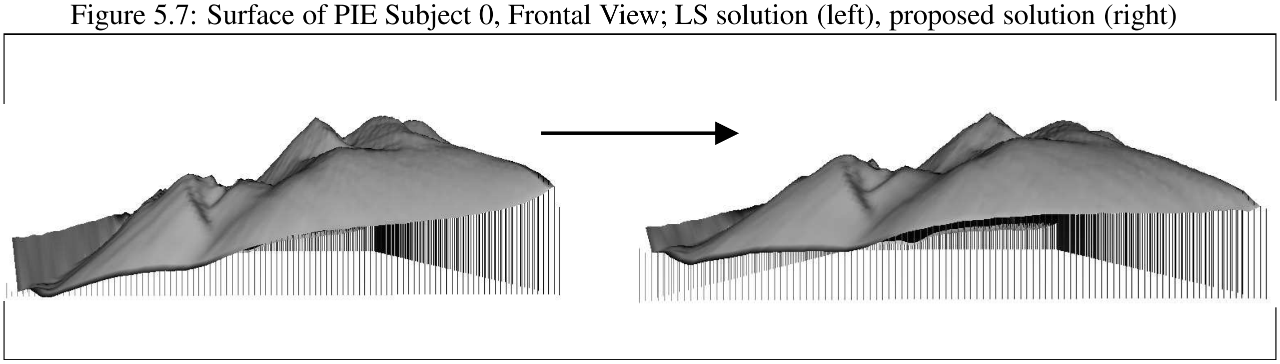

Figure 5.8: Surface of PIE Subject 0, Profile View; LS solution (left), proposed solution (right)

Test Case	Recognition Rate
Baseline - (6, 7, 8, 9, 10)	28
SFS+PCA - (6, 7, 8, 9, 10)	58
SFS+PCA+S. Det - (6, 7, 8, 9, 10)	62
Baseline - (3, 5, 9, 13, 16)	27
SFS+PCA - (3, 5, 9, 13, 16)	53
SFS+PCA+S. Det - (3, 5, 9, 13, 16)	55

Table 5.2: Recognition Rates of Baseline, SFS+PCA, and SFS+PCA+S. Det Methods, when gallery and probe images have no ambient light

Test Case	Recognition Rate
Baseline - (6, 7, 8, 9, 10)	37
SFS+PCA - (6, 7, 8, 9, 10)	45
SFS+PCA+S. Det - (6, 7, 8, 9, 10)	46
Baseline - (3, 5, 9, 13, 16)	23
SFS+PCA - (3, 5, 9, 13, 16)	59
SFS+PCA+S. Det - (3, 5, 9, 13, 16)	61

Table 5.3: Recognition Rates of Baseline, SFS+PCA, and SFS+PCA+S. Det Methods, when gallery images have no ambient light, but probe images do have ambient light

algorithm involving PCA and shadow–detection in all the SFS steps (referred to as SFS+PCA+S. Det). Twenty-four (24) of the PIE subjects were used to compute the PCA–space, and the remaining 40 subjects were used for recognition. The PCA space contained 20 eigenvectors, and the images were processed at a resolution of 25x25 pixels. Naturally, each image was processed independently (this is the single–face–image normalization test). Table (5.2) shows the results of the three image sets being compared, when different subsets of the images are used as the gallery set for the classifier, and the gallery and probe images do not contain ambient light. The paranthesized lists in the first column show which PIE illuminations are being used for the gallery for each subject: (6,7,8,9,10) are all relatively frontal lighting, whereas (3,5,9,13,16) spans extreme illumination variation in the X–direction. Table (5.3) shows the recognition results when the gallery contains images with no ambient light, but the probe images do contain ambient light. We can see a great deal of improvement (roughly double the recognition rate, from ~ 25% to ~ 50–60%) when we perform either of our algorithms versus performing no pre–processing, and the shadow–detection enhancement to the normalization algorithm (SFS+PCA+S. Det) provides an additional improvement of 2–4% as well.

From this, we can say that our illumination normalization algorithm is able to improve recognition dramatically under a variety of conditions. The normalized images of subject 0 of the PIE database are shown in two groups: Figures 5.9, 5.10, and 5.11 are images with no ambient light, and Figures 5.12, 5.13, and 5.14 are images with ambient light. Notice that all of the normalized images appear to be frontally lit.

Test Case	Recognition Rate
Baseline (unprocessed PIE)	31
SFS+PCA+S. Det	40
SFS+PCA+S. Det w/ Ratio Image	42
Gain/Offset Correction	48
Histogram Equalization	47
Canny Edge Detector	65
Classic Retinex	36
Homomorphic Filtering	41
Single–Scale Retinex	43
Ratio Image	88
Ratio Image w/ Shifts and Avg.	**93**
Ratio Image, Sobel Filter	73
Ratio Image, Canny Edge Det.	54

Table 5.4: Recognition Rates of many algorithms for the (gallery/probe) dataset case (noamb/noamb)

5.3. Face Recognition – The Big Picture

We now focus our attention on the image processing algorithms presented in Section 4. Although substantial gains in recognition performance can be achieved with the SFS methods, we were able to achieve even higher levels of performance with simple image processing algorithms. Below, we show the average performance of these algorithms when one gallery image is provided for each subject; the performance given below is the average of the recognition rates that are achieved when PIE images 3, 5, 9, 13, 16, and 20 are used as the gallery sets. For each algorithm that we test, 4 tests are conducted; these tests show the performance when different PIE datasets (with no ambient light, with ambient light) are used for the gallery and probe. We will abbreviate “with no ambient light” as “noamb,” and “with ambient light” as “amb.” The four possible combinations, which we write in the form (gallery/probe), are (noamb, noamb) — Table 5.4, (noamb, amb) — Table 5.5, (amb, noamb) — Table 5.6, and (amb, amb) — Table 5.7. We believe this is indicative of real–world performance, since there are many applications where only a single image of a subject is available.

Note that we additionally tested the Canny edge detector [20]; we did this because the methods that performed best (such as the Ratio Image) seemed to preserve the edges of the input images and discard the remaining features. However, we can see that the Ratio Image is not simply performing edge–extraction,

Figure 5.9: Subject 0 of the PIE database (no ambient light)

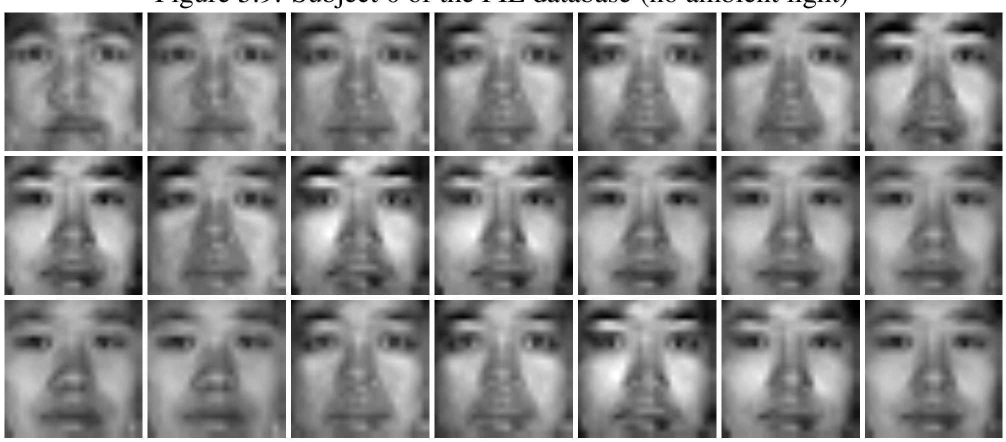

Figure 5.10: Subject 0 of the PIE database (no ambient light), normalized with SFS+PCA

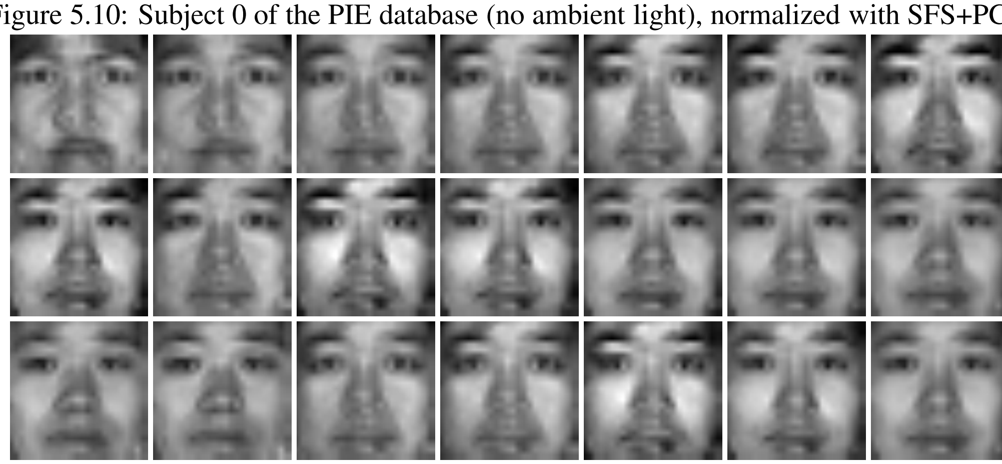

Figure 5.11: Subject 0 of the PIE database (no ambient light), normalized with SFS+PCA+S. Det

Figure 5.12: Subject 0 of the PIE database (with ambient light)

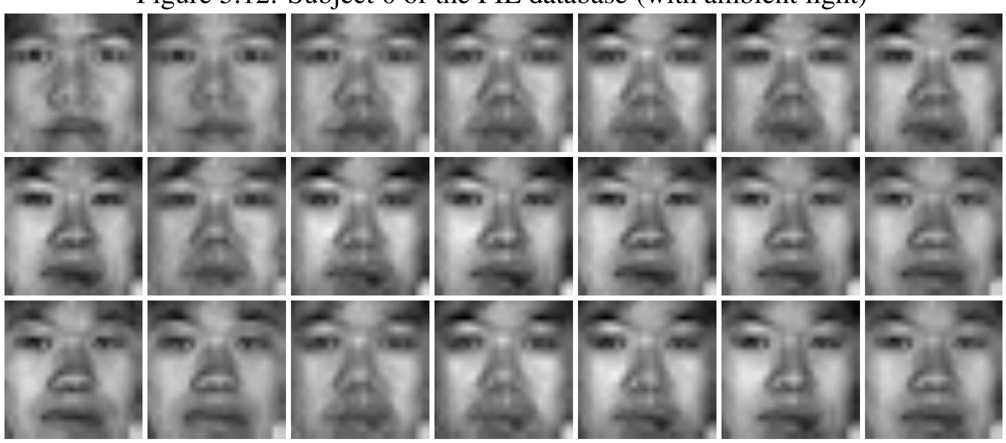

Figure 5.13: Subject 0 of the PIE database (with ambient light), normalized with SFS+PCA

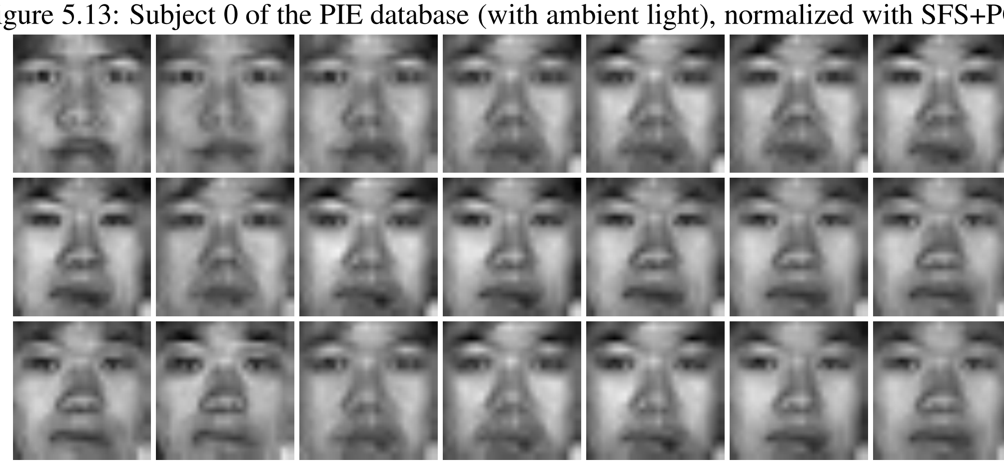

Figure 5.14: Subject 0 of the PIE database (with ambient light), normalized with SFS+PCA+S. Det

Test Case	**Recognition Rate**
Baseline (unprocessed PIE)	3
SFS+PCA+S. Det	37
SFS+PCA+S. Det w/ Ratio Image	39
Gain/Offset Correction	48
Histogram Equalization	47
Canny Edge Detector	26
Classic Retinex	32
Homomorphic Filtering	31
Single–Scale Retinex	35
Ratio Image	62
Ratio Image w/ Shifts and Avg.	**64**
Ratio Image, Sobel Filter	33
Ratio Image, Canny Edge Det.	14

Table 5.5: Recognition Rates of many algorithms for the (gallery/probe) dataset case (noamb/amb)

Test Case	**Recognition Rate**
Baseline (unprocessed PIE)	3
SFS+PCA+S. Det	17
SFS+PCA+S. Det w/ Ratio Image	18
Gain/Offset Correction	40
Histogram Equalization	39
Canny Edge Detector	21
Classic Retinex	25
Homomorphic Filtering	18
Single–Scale Retinex	39
Ratio Image	**60**
Ratio Image w/ Shifts and Avg.	54
Ratio Image, Sobel Filter	21
Ratio Image, Canny Edge Det.	17

Table 5.6: Recognition Rates of many algorithms for the (gallery/probe) dataset case (amb/noamb)

Test Case	Recognition Rate
Baseline (unprocessed PIE)	58
SFS+PCA+S. Det	52
SFS+PCA+S. Det w/ Ratio Image	62
Gain/Offset Correction	75
Histogram Equalization	76
Canny Edge Detector	96
Classic Retinex	90
Homomorphic Filtering	79
Single–Scale Retinex	74
Ratio Image	99
Ratio Image w/ Shifts and Avg.	**100**
Ratio Image, Sobel Filter	96
Ratio Image, Canny Edge Det.	93

Table 5.7: Recognition Rates of many algorithms for the (gallery/probe) dataset case (amb/amb)

since it yields much higher recognition rates than using the Canny edge detector by itself. We also tried applying edge extraction to the images generated by the Ratio Image (denoted as “Ratio Image, Sobel Filter” and “Ratio Image, Canny Edge Det” in the tables). However, the performance is significantly worse when these edge detectors are used, so clearly both edge information and texture is being used by the classifier. We can conclude from the data that the “Ratio Image w/ Shifts and Avg.” provides the best face recognition rates, out of all the image processing algorithms tested.

Chapter 6

Conclusion

In our final remarks, we can say that the Lambertian model typically used for SFS can be extended to account for the presence of shadows, yielding favorable performance. The overall approach when multiple input images of the same subject are provided, is to start with an estimate of the surface, and iteratively find the shadowed pixels and disclude them from the subsequent surface estimate. Following the proposed approach has yielded up to a 22% improvement in squared error. We can also conclude that our illumination normalization approach for face recognition is capable of improving recognition rates dramatically, under a variety of situations.

We have also found that certain simple image processing algorithms can provide excellent results for illumination normalization. The "Ratio Image w/ Shifts and Avg." provided the best face recognition rates, and the image quality is also (subjectively) good. One interesting result of this work is finding that images which are easier for humans to recognize features in, are not necessarily the best images to provide to classifiers. This is clear when we look at the recognition rates; image processing algorithms (such as Single–Scale Retinex) whose goal is to make images look better for humans do not perform as well as the "Ratio Image" which produces more un–natural images.

Future work includes applying the "Ratio Image" to video, which could involve operations such as applying the lowpass filter over time and using motion–compensation. Efficient implementations for use on embedded systems should be possible with this algorithm as well.

Bibliography

[1] B. K. P. Horn and M. J. Brooks. "Shape From Shading." The MIT Press, Cambridge, Massachusetts, 1989.

[2] R. J. Woodham. "Photometric Method for Determining Surface Orientation from Multiple Images." *Optical Engineering*, 19(1):139-144, 1980.

[3] Y. Hsu and T. Chen. "Illumination Normalization for Face Recognition," Master's Thesis, Dept. of Electrical and Computer Engineering, Carnegie–Mellon University, 2002.

[4] S. K. Nayar, K. Ikeuchi, T. Kanade. "Recovering Shape in the Presence of Interreflections." *IEEE Int. Conference on Robotics and Automation*, 1991.

[5] S. Barsky and M. Petrou. "The 4-source Photometric Stereo Technique for Three-Dimensional Surfaces in the presence of highlights and shadows." *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 25, Issue 10, pp 1239-1252, 2003.

[6] Tae-Eun Kim, Sun-Ho Lee, Seok-Hyun Ryu, Jong-Soo Cho. "Shape Reconstruction Using Extended Photometric Stereo." *Proc. of the International Conference on Information, Communications, and Signal Processing*, 1997.

[7] Q. Zheng and R. Chellappa. "Estimation of Illuminant Direction, Albedo, and Shape from Shading." *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 540–545, 1991.

[8] A. Shashua. "Geometry and Photometry in 3D Visual Recognition." PhD Thesis, MIT, 1994.

[9] R. Zhang, P. Tsai, J. E. Cryer, and M. Shah. "Shape From Shading: A Survey." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7), July 1997.

[10] T. Sim and T. Kanade. "Combining Models and Exemplars for Face Recognition: An Illuminating Example," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1997.

[11] J. Atick, P. Grifn, and N. Redlich. "Statistical Approach to Shape from Shading: Reconstruction of Three-Dimensional Face Surfaces from Single Two-Dimensional Images." *Neural Computation*, Vol 8, pp. 1321-1340, 1996.

[12] V. Blanz and T. Vetter. "A Morphable Model for the Synthesis of 3D Faces." *Proceedings of ACM SIGGRAPH 99*, pages 187-194, 1999.

[13] W. Zhao, and R. Chellappa. "Illumination-insensitive Face Recognition Using Symmetric Shape-from-Shading." *Proc. Conference on Computer Vision and Pattern Recognition*, SC, pp. 286-293, 2000.

[14] T. Riklin-Raviv and A. Shashua. "The Quotient Image: Class Based Recognition and Synthesis Under Varying Illumination Conditions." *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, Vol. 2, pp. 2566-2573, June, 1999.

[15] D. W. Jacobs, P. N. Belhumeur and R. Basri. "Comparing Images Under Variable Illumination." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1998.

[16] D. J. Kriegman and P. Belhumeur. "What Shadows Reveal About Object Structure." *Proc. European Conf. on Computer Vision*, pages 399–414, 1998.

[17] R. Basri and D. Jacobs. "Lambertian Reflectance and Linear Subspaces." *International Conference on Computer Vision*, pages 383–390, 2001.

[18] A. Georghiades, P. N. Belhumeur, and D. J. Kriegman. "From Few to Many: Illumination Cone Models for Face Recognition under Variable Lighting and Pose." *IEEE Trans. on Pattern Analysis and Machine Intelligence*, June 2001, pp. 643-660.

[19] A. Georghiades, D. J. Kriegman, and P. Belhumeur. "Illumination Cones for Recognition Under Variable Lighting: Faces." *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 52-59, 1998.

[20] J. Canny. "A Computational Approach to Edge Detection." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, pp. 679–698, 1986.

[21] A. Prati, I. Mikic, M. M. Trivedi, R. Cucchiara. "Detecting Moving Shadows: Algorithms and Evaluations." *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 25, No. 7, July 2003.

[22] J. E. Bresenham. "A Linear Algorithm for Incremental Digital Display of Circular Arcs." *Communications of the ACM*, 20(2), Feb. 1977.

[23] F. K. Musgrave. "Grid Tracing: Fast Ray Tracing For Height Fields." *Technical Report YALEU/DCS/RR-639*, Yale University Dept. of Computer Science Research, 1988.

[24] The Persistence of Vision Raytracer (POV-Ray). *http://www.povray.org*

[25] T. Sim, S. Baker, and M. Bsat. "The CMU Pose, Illumination, and Expression (PIE) Database." *Proc. of the 5th IEEE International Conference on Automatic Face and Gesture Recognition*, 2002.

[26] F. De la Torre and M. J. Black. "Robust principal component analysis for computer vision." *Proc. of the International Conference on Computer Vision*, 2001.

[27] R. Collobert, S. Bengio, and J. Mariéthoz. "Torch: A Modular Machine Learning Software Library." *Technical Report IDIAP-RR 02-46, IDIAP*, 2002.

[28] C. C. Chang and C. Lin. "LIBSVM: a Library for Support Vector Machines (2001)." *http://www.csie.ntu.edu.tw/ cjlin/libsvm/index.html*

[29] R. T. Frankot and R. Chellappa. "A Method for Enforcing Integrability in Shape from Shading Algorithms." *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 10:439-451, July 1988.

[30] E. Land and J. McCann. "Lightness and Retinex Theory." *Journal of the Optical Society of America*, No. 61, 1971.

[31] B. Funt, F. Ciurea, and J. McCann. "Retinex in Matlab," *Proceedings of the IS&T/SID Eighth Color Imaging Conference: Color Science, Systems and Applications*, 2000, pp 112-121.

[32] R. Gross and V. Brajovic. "An Image Preprocessing Algorithm for Illumination Invariant Face Recognition." *Proc. of the Fourth International Conference on Audio– and Video–Based Biometric Person Authentication*, June, 2003.

[33] W. Press, S. Teukolsky, W. Vetterling, B. Flannery. Numerical Recipes in C. Cambridge University Press, 1992.

[34] Z. Rahman, G. A. Woodell, and D. J. Jobson. "A Comparison of the Multiscale Retinex with Other Image Enhancement Techniques." *Proceedings of the IS&T 50th Anniversary Conference*, May, 1997.

[35] Z. Rahman. "Properties of a Center/Surround Retinex: Part 1. Signal Processing Design." NASA CR-198194, August 1995, pp. 15.