國立交通大學

電機與控制工程研究所

碩士論文

適用於眼睛偵測之眼鏡影像增強與反光分離

Glasses Image Enhancement and Reflection Separation for Eye Detection

研究生：何憲信

指導教授：張志永

中華民國九十五年七月

適用於眼睛偵測之眼鏡影像增強與反光分離

Glasses Image Enhancement and Reflection Separation for Eye Detection

研究生：何憲信　　Student：Hsien-Hsin Ho

指導教授：張志永　　Advisor：Jyh-Yeong Chang

國立交通大學

電機與控制工程學系

碩士論文

A Thesis

Submitted to Department of Electrical and Control Engineering

College of Electrical Engineering and Computer Science

National Chiao Tung University

in Partial Fulfillment of the Requirements

for the Degree of Master in

Electrical and Control Engineering

July 2006

Hsinchu, Taiwan, Republic of China

中華民國 九十五 年 七 月

適用於眼睛偵測之眼鏡影像增強與反光分離

學生: 何憲信　　　　指導教授: 張志永博士

國立交通大學電機與控制工程研究所

摘要

當人們在工作中或是在駕駛的環境中，打瞌睡常常是造成意外事故最常見的因素。而以眼睛的開閉狀態為基礎的瞌睡偵測系統，最重要的便是精確的眼睛偵測。在本篇論文中，我們提出從一張人臉影像中偵測出眼睛位置的演算法。在昏睡偵測或人臉辨識的系統中，當被偵測者有佩帶眼鏡或太陽眼鏡，除了太陽眼鏡本身色度會影響眼睛的偵測，也常常會因為有反光在眼鏡鏡片上產生，而使得偵測系統偵測失敗。所以如何消除太陽眼鏡帶來的干擾以及從這些鏡片上將反光正確地去除或分離是相當重要的問題。在此，我們使用影像增強的技術以及將鏡片上的反光去除或分離的方法，來改善這些情況的發生。如何將一張輸入的影像正確分離成反光與非反光兩個部分是非常困難的問題，因為缺乏有關所見影像的額外資訊的限制條件，分離的結果可能會有無數種組合發生。我們提供一種簡單的演算法來執行這種分離。給定一張有反光的影像當作輸入，演算法會將此輸入分解成兩張影像，而使得所分解出來的兩張影像，它們具有最少的角和邊緣的數量總和；這個方法在從有反光的單張影像上做出正確的分離，是相當有效的。圖片上有反光的眼鏡的區域也是類似上述的情況，所以我們將上述的原理應用在眼鏡的反光去除。

Glasses Image Enhancement and Reflection Separation for Eye Detection

STUDENT: HSIEN-HSIN HO ADVISOR: Dr. JYH-YEONG CHANG

Institute of Electrical and Control Engineering
National Chiao-Tung University

ABSTRACT

Drowsiness is often one of the most important factors causing accidents on various occasions such as work fields and vehicle driving. For drowsiness detection system based on the states of eyes, accurate eye detection is the most important. For a given face image, we present an algorithm to detect the eye location automatically. In drowsiness detection or face recognition systems, in addition to the effect caused by sunglasses, the detection also often fails from the reflections on the wearing glasses or sunglasses. Therefore, eliminating the interference caused by sunglasses, and removing or separating the reflections from the glasses are very important for drowsiness and face detections. In thesis, we utilize an image enhancement technique and an approach which can separate the reflections on the glasses to improve the problems above. How to decompose a single input image into reflection and non-reflection images correctly is very difficult because of the absence of additional knowledge or constraints about the scene being viewed. There will be an infinite number of valid decompositions. We describe an algorithm that uses a simple implementation to perform the decomposition. Given a single image with reflection as input, the algorithm searches for a decomposition into two images that minimize the total amount of edges and corners of the two images. The approach is effective to obtain quite correct separations on reflection scenes using only a single image. In a similar manner, we apply our method to the reflection removal on glasses.

.

ACKNOWLEDGEMENTS

I would like to express my sincere appreciation to my advisor, Dr. Jyh-Yeong Chang. Without his patient guidance and inspiration during the two years, it is impossible for me to complete the thesis. In addition, I am thankful to all my Lab members for their discussion and suggestion.

Finally, I would like to express my deepest gratitude to my family. Without their strong support, I could not go through the two years.

Contents

List of Figures

Chapter 1

Introduction

Eye detection plays an important role in various applications such as human or face recognition, eye gaze detection, drowsiness detection, and so on. In these applications, we particularly aim at drowsiness detection system in dealing with eyeglasses interference problem. Many traffic accidents happen due to a diminished driver's vigilance level every year. Driver fatigue is a significant factor in these accidents. Drivers with a diminished alertness level suffer from a heavy slump in their abilities of dangers detection and vehicle control, and therefore serious dangers often occur. Statistics show that a primary cause for fatal traffic accidents is due to drivers with a diminished vigilance level. In the trucking industry, fatal truck accidents are often due to driver fatigue. This is a serious problem for driver safety. As a result, systems which can effectively monitor a driver's level of vigilance and warn the drivers of the danger arise from drowsiness are essential for preventing traffic accidents. Systems for drowsiness detection based on the Long Duration Blink Frequency (LDBF) and the PERcentage of eyelid CLOSure (PERCLOS) are commonly used [1]–[4]. In these systems, accurate eye detection is quite important. We not only pay attention to eye detection but also further consider the condition that drivers wear glasses or sunglasses and utilize some techniques to eliminate the interference in eye detection. Moreover, reflections on glasses also interfere with eye detection significantly. We attempt to adopt some techniques to deal with this condition as well.

1.1 Motivation

Eye detection has been a topic of general research in computer vision for many applications such as face recognition and drowsiness detection in recent years. Drowsiness detection for a driver is one of the important subjects among these applications. For drowsiness detection based on eye features, accurate eye detection is significantly essential to the system. However, when a driver wears eyeglasses or sunglasses, the system often cannot detect the eye position accurately because the frame of glasses overlapped with eyes. Besides, the reflectance property of glasses differs significantly from that of human skin. Sometimes the reflection on the glasses is the brightest area on face region and even covers up the eyes. An unsuccessful drowsiness detection often follows an imperfect eye detection and then may lead to a traffic accident if the system was installed in a car.

For more perfect eye detection, there are many conditions we have to consider. Sunglasses and reflections are critical interferences so that some methods must be developed to overcome these barriers. Sunglasses region must be enhanced so that the detection of eyes could be more accurate. When there are reflections on sunglasses, they must be removed to avoid interfering the detection as well. Consequently, researches for these purposes are quite essential subjects.

1.2 Face Detection and Eye Detection Module

Many methods or algorithms have been proposed for face detection in the recent years. Hjelmas [5] did a comprehensive survey on this subject, listing more than two hundred references. These approaches utilize techniques such as principal component analysis, neural networks, machines learning, information theory, geometrical modeling, (deformable) template matching, Hough transform, motion extraction, and color analysis. One of the most popular face candidate extraction methods is to extract a skin-tone region in color images [6]–[9].

It is a important feature that human faces have a special color distribution that differs significantly from those of the background objects. Therefore, we use colors of input images as a feature [6] to identify the face region of a human. First we locate the face region, and then try to find the eye region such that eye features could be used for some applications, e.g. drowsiness detection. In [10]–[12], there is a brief explanation of the eyeball detection procedure.

After locating the face region, preprocessing is first performed to binarize the facial image and then remove noises, which makes it possible for the image to be accepted easily by the image processor.

The maximum width of the face is then detected so that the right and left boundaries of the face can be found. After the right and left boundaries of the face was determined, the vertical position of each eye is detected respectively within an area defined by the center line of the face width and lines running through the outermost points of the face. After that, the region within which each eye locates could be determined.

1.3 Eye Detection with sunglasses

When drivers wear sunglasses, the visibility of eyes often becomes weaker, and the detection for eyes often fails. Image enhancement techniques are needed to solve this problem. Many methods for image enhancement have been proposed. Among some famous methods, a method named as retinex is popular and used for many image applications. We will introduce this image enhancement method and apply it to solve the interference caused by sunglasses. Eyes in the region processed by this way are enhanced and are more distinct. As a result, the detection for eyes will be more accurate.

1.4 Reflection Separation

Many methods have been developed to separate reflections. Wolff and Boult [13] and Nayar *et al*. [14] used a polarizing filter to identify pixels that had a specular reflection component. In [15] and [16], two photographs of the same scene were taken with a different polarizing filter. The filter attenuates the reflection in different amounts and it is possible to decompose the images by using ICA on the two input images. Generally, methods using polarizing filters are sufficiently accurate to separate reflection components; however, using such additional devices is impractical in some circumstances. Another method to decompose images by analyzing a movie sequence in which the reflection and the non-reflected images have different motions is mentioned [17]–[19]. Other methods using multiple images can be found in the literature [20], [21].

When the area of reflections on the sunglasses is smaller, we can view these reflections as noises, and consider some noise removal methods to eliminate them. There are many smoothing methods used to eliminate noises from images. An image smoothing method using bilateral filters have been proposed in [22]. This smoothing method is helpful to eliminate the small reflections on sunglasses.

If the areas of reflections on glasses or sunglasses are bigger, the noise removal methods do not avail against eliminating them. Therefore, there are some other approaches should be considered to address these conditions. To overcome these conditions, many methods have been proposed as above. We are more interested in those ones based on image processing techniques. A user assisted separation of reflections from a single image method was proposed by Levin and Weiss [23]. Nevertheless, decomposing an input image from a single image automatically may be more attractive and effective to a drowsiness detection system. Therefore, a method proposed in [24] may be useful to separate reflections for a real time drowsiness detection system. We will introduce this method and adopt it to deal with images with reflections. In this method, a concept of viewing scenes through transparent glass is mentioned, and an algorithm that uses a simple form of prior knowledge is used to perform the separation of reflections from images.

1.5 Thesis Outline

The functions of eyeball detection comprise the first half segment of this thesis. We adopt some techniques to deal with the interference caused by sunglasses as well. In the second half, some reflection removal and separation techniques are adopted to eliminate the reflections on sunglasses.

The contents of this thesis are organized as follows. In Chapter 2, the face location detection will firstly be described, and then the eye position detection follows. Then the detection will be extended to drivers wearing sunglasses. An image enhancement method will be introduced and then applied to the sunglasses region to enhance the visibility of eyes. In Chapter 3, reflection separation methods are referred and adopted to deal with the reflections with bigger areas. In Chapter 4, we do some simulations and show our experimental results of the methods utilized in this paper. Some discussions about the experimental results are given as well. In the end, we make some conclusions and discuss future works for advances in Chapter 5.

Chapter 2

Face and Eye Detection

2.1 Introduction

For drowsiness detection, it is necessary to preprocess the captured image sequences for a subject. The preprocessing operation includes face detection and eye detection. Eye area detection will be useful to estimate the degree of eye openness. Consequently, an accurate detection for face and eyes is very essential for drowsiness detection system. Here we adopt a method proposed by Chai *et al.* [6]. This method involves a fast, reliable, and effective algorithm that utilizes the spatial distribution characteristics of human skin color. Based on the special spatial distribution of the detected skin-color pixels, the algorithm employs a set of regularization processes to reinforce regions of skin-color pixels that are more likely to belong to the facial regions and eliminate those that are not.

2.2 Face Segmentation Algorithm

The algorithm in [6] is an unsupervised segmentation algorithm, and hence no manual adjustment of any design parameter is needed in order to suit any particular input image. This algorithm is very robust and effective to detect sundry human faces. The only principal prerequisite is that the person’s face must be present in the given image, since we are locating the face rather than detecting whether there is a face. The algorithm we use for face segmentation consists of four stages, as outlined in figure 2.1. The face segmentation algorithm is within the frame composed of dotted lines.

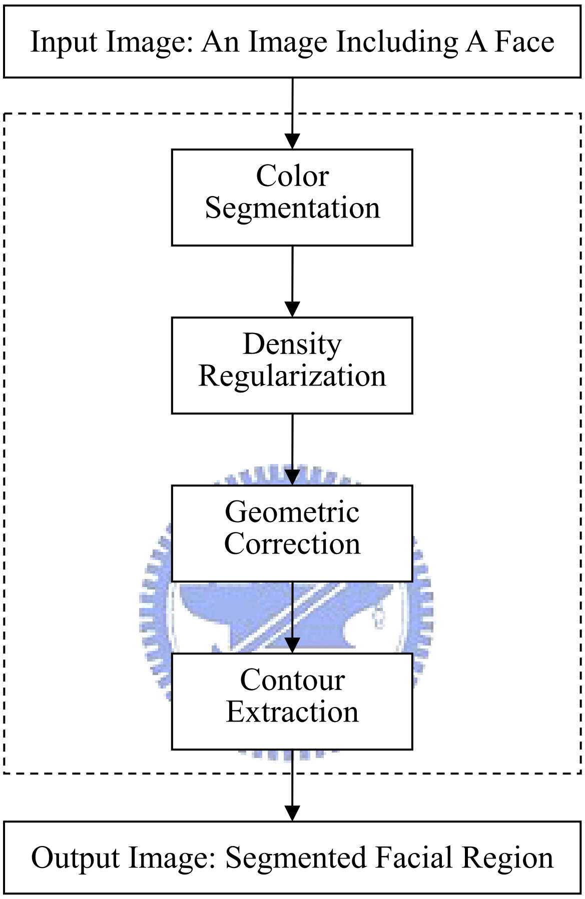


Fig. 2.1. Outline of face-segmentation algorithm.

A. Color Segmentation

In this stage, it is important to choose an appropriate color space for modeling human skin color. Application and effectiveness for face segmentation are two factors we have to consider. Here we use the YCbCr color space. The reason is as follows. First, an effective use of the chrominance information for modeling human skin color can be achieved in this color space. Second, this format is typically used in video coding, and therefore the use of the same color space for segmentation will avoid the extra computation required in conversion.

The first stage of the algorithm is to classify the pixels of the input image to skin region and non-skin region. We first convert the RGB color space of input image into the YCbCr color space according to the transformation given in Eq. (2.1), and then refer to a skin-color reference map in YCbCr color space for this task. We denote R_{C_r} and R_{C_b} as the ranges of C_r and C_b values respectively that correspond to skin color. The ranges have tested are $R_{C_r} = [133,\ 173]$ and $R_{C_b} = [77,\ 127]$ for this paper. With the skin-color reference map, we can get the color segmentation result O_A as

$$O_A(x,y) = \begin{cases} 1, & \text{if } \left[C_r(x,y) \in R_{C_r}\right] \cap \left[C_b(x,y) \in R_{C_b}\right] \\ 0, & \text{otherwise} \end{cases} \tag{2.1}$$

where $x = 0, \ldots, M$ and $y = 0, \ldots, N$. A pair (x, y) represents the coordinate of a pixel in the input image. M, N are the height and width of the input image respectively. Fig. 2.3 shows an example to illustrate the classification of the original image given in Fig. 2.2.

The result of color segmentation is the detection of pixels in a facial region and may also include other areas where the chrominance values coincide with those of the skin color coincidentally. As seen in Fig. 2.2, the chrominance values of some part of hair in the upper right of the image are consistent with skin color, and therefore those pixels are classified to the skin region falsely. Hence the successive operating stages of the algorithm are used to remove these non-facial areas as described below.

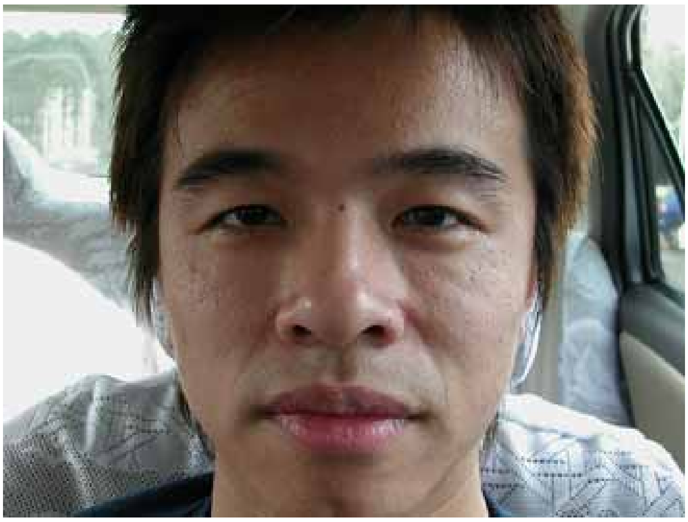


Fig. 2.2. Original image.

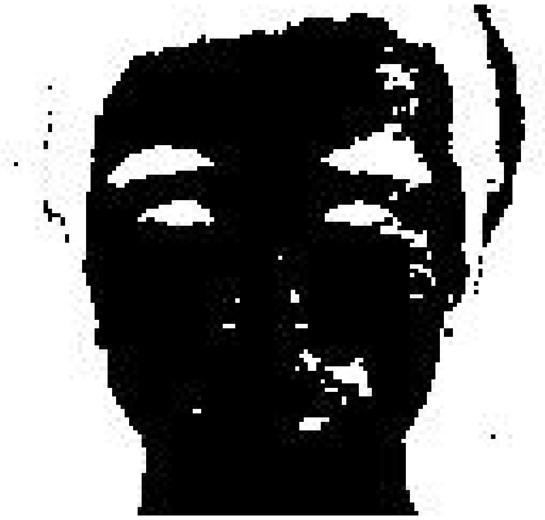


Fig. 2.3. Image after color segmentation by skin-color map in stage A.

B. Density Regularization

This stage considers the bitmap produced by the preceding stage to contain the facial region that is corrupted by noise. The noise may be small holes on the facial region due to undetected facial features such as eyes, mouth, and even glasses. It may also appear as objects with skin-color appearance in the background scene. This stage performs simple morphological operations such as dilation to fill in any small hole in the facial region and erosion to remove any small object in the background scene. To distinguish facial region form non-facial region, we first need to identify regions of the bitmap that have higher probability of being the facial region. For this task, a density map is calculated as follows.

$$D\left(x,y\right)=\sum_{i=0}^{3}\sum_{j=0}^{3}O_A\left(4x+i,\ 4\mathrm{y}+j\right) \qquad (2.2)$$

It first partitions the output bitmap of stage A $O_A(x,\ y)$ into non-overlapping groups of 4×4 pixels, then counts the number of skin-color pixels within each group and assigns this value to the corresponding point of the density map.

According to the density value, we classify each pixel into one of three clusters, namely, zero ($D = 0$), intermediate ($0 < D < 16$), and full ($D = 16$). Fig. 2.4 shows the density map of the output bitmap of stage A shown in Fig. 2.3 with three density classifications. The point of zero density is shown in white, intermediate density in yellow, and full density in black. A group of points with white color will likely represent a non-facial region, while a group of black points will signify a cluster of skin-color pixels and a high probability of belonging to a facial region. Points with intermediate density values and therefore shown in yellow will probably indicate the presence of noise.

After the density map is derived, we can then begin the process that we termed as density regularization. This includes the three steps as below.

1) Discard all points at the edge of the density map, i.e., set $D(0, y) = D(M/4–1, y) = D(x, 0) = D(x, N/4–1)$ for all $x = 0, 1, …, M/4–1$ and $y = 0, 1, …, N/4–1$.

2) Erode any full-density point (i.e., set to zero) if it is surrounded by less than five other full-density points within its local 3╳3 neighborhood.

3) Dilate any point with either zero or intermediate density (i.e., set to 16) if there are more than two full-density points within its local 3╳3 neighborhood.

Processed by density regularization, the density map is converted to the output bitmap of stage B as

$$O_B(x,y)=\begin{cases}1, & \text{if } D(x,y)=16\\ 0, & \text{otherwise}\end{cases} \tag{2.3}$$

for all $x = 0, 1, …, M/4–1$ and $y = 0, 1, …, N/4–1$. The result of the bitmap in Fig. 2.3 processed after stage B is shown in Fig. 2.5.

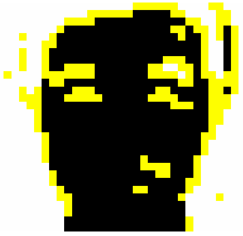


Fig. 2.4. Density map of Fig. 2.3 after classified to three classes.

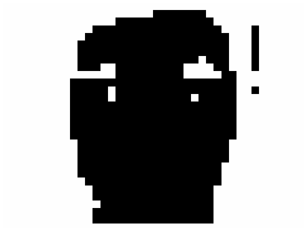


Fig. 2.5. Result of Fig. 2.3 produced by stage B.

C. Geometric Correction

After stage B, there may be still some fragmented areas in the output bitmap. In order to eliminate or mend these areas, we adopt some methods of geometric correction. In this stage, we performed a horizontal and vertical scanning process to identify the presence of any odd structure in the preceding bitmap obtained from stage B, $O_B(x, y)$, and subsequently removed it. This is to ensure that we can obtain a correct geometric shape of the facial region. First, we use a technique similar to that introduced in stage B to further remove any more noise. A pixel in $O_B(x,y)$ with a value of one will remain as a detected facial pixel if there are more than three other pixels with the same value in its local 3×3 neighborhood. Simultaneously, a pixel in $O_B(x,y)$ with a value of zero will be reconverted to a value of one (i.e., as a potential pixel of the facial region) if it is surrounded by more than five pixels with a value of one in its local 3×3 neighborhood.

A bitmap of well-detected facial region should look continuous, and therefore any short run of pixels with the value different from the detected facial region should unlikely belong to this region. As a result, next to the process above, we then begin the horizontal scanning process on the filtered bitmap. We search for any short continuous run of pixels which are assigned with the value of one. Any group of less than four horizontally connected pixels with the value of one will be eliminated and assigned to zero. A similar process is then performed in the vertical direction. After all processes in this stage, the output bitmap should contain the facial region with minimal or even no noise, as shown in Fig. 2.6.

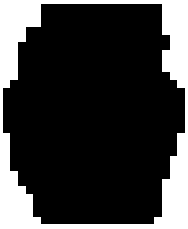


Fig. 2.6. Output of the bitmap produced by stage C.

D. Contour Extraction

In this stage, we convert the output bitmap of stage C back to the original dimension of the extracted face region from stage A. To achieve the increase in spatial resolution, we utilize the edge information that is already made available by the color segmentation in stage A. Therefore, all the boundary points in the previous bitmap will be mapped into the corresponding group of 4×4 pixels with the value of each pixel as defined in the output bitmap of stage A. The output of this final stage of the algorithm is shown in Fig. 2.7.

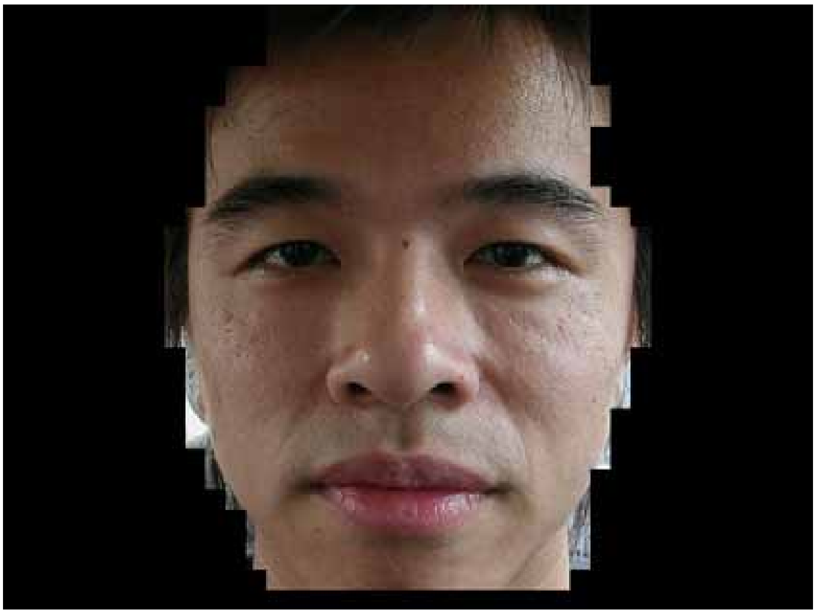


Fig. 2.7. Image produced by stage D.

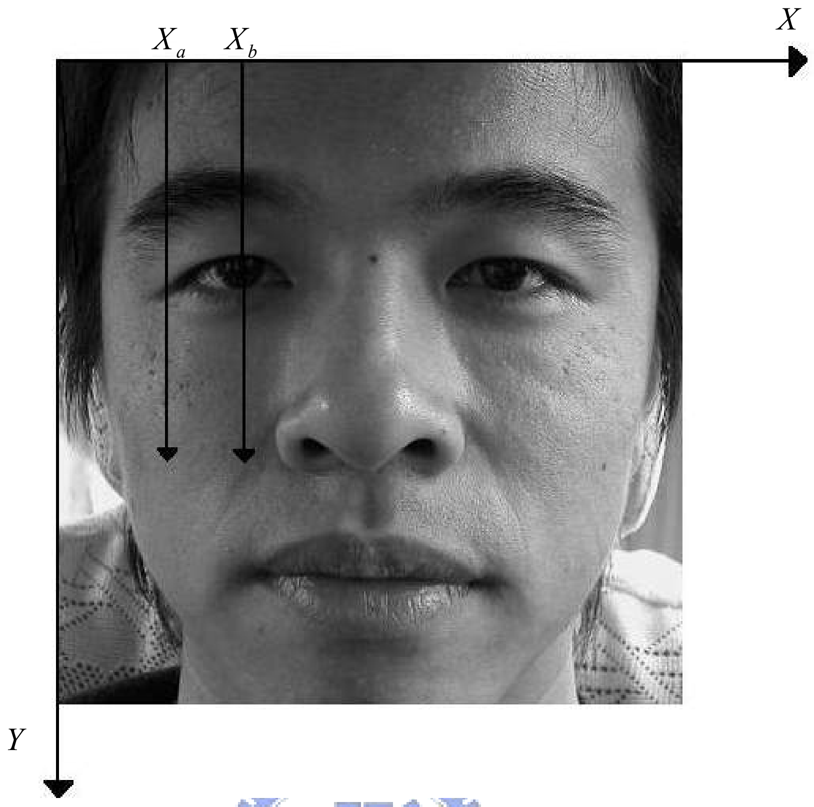
2.3 Eye Position Detection

After the face region of a input image is located, we then try to extract the eye position such that we could measure eye features for applications such as drowsiness detection and so on. Many methods have been proposed to find the eye position such as using a circle-frequency filter to find a candidate "between-eyes" in [25] proposed by Kawato *et al*. For the general purpose to every user without database, we adopt a simple method to implement the detection for eye position.

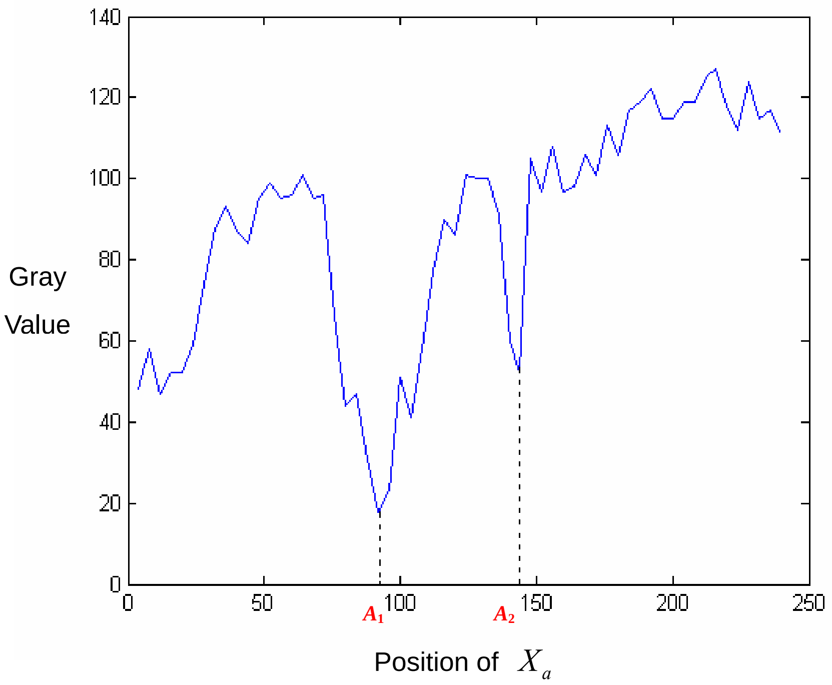
In order to determine the eye position, the maximum width of the face on the input image must be estimated first. Then based on the fact that the eyeball is about one-fourth of the facial width, we can easily obtain the lateral eye position on the face. The details of the process are summarized in Fig. 2.8. We consider the gray scale of an input image and then draw two vertical lines, or more if necessary, around one-fourth of the facial width. As shown in Fig. 2.8(a), one line X_a does not cross the bulb of the eye and another line X_b does. Fig. 2.8(b) depicts the changes in the gray scale values along line X_a of Fig. 2.8(a). Likewise, the changes in the gray scale values along line X_b are shown in Fig. 2.8(c).

Observing Fig. 2.8(a), along these two lines we can find that there are two darker candidates which are expected to have lower gray-level values. As the gray-level value falls to the local minimum, it may correspond to the position where the eyebrow or eyeball possibly locates. Since the eyebrow corresponds to a lower gray value, from Figs. 2.8(b) and 2.8(c), we can infer that the valleys A_1 and B_1 are the positions of eyebrow crossed by the two lines X_a and X_b respectively. It is also known that the eyeballs are most likely the darkest in the gray scale. It is clear that the line X_a does not include the pupil of the eye, but the line X_b does. Consequently, we can expect that the gray scale values corresponding to B_2 (the second valley) in Fig. 2.8(c) is smaller than those corresponding to A_2 in Fig. 2.8(b).

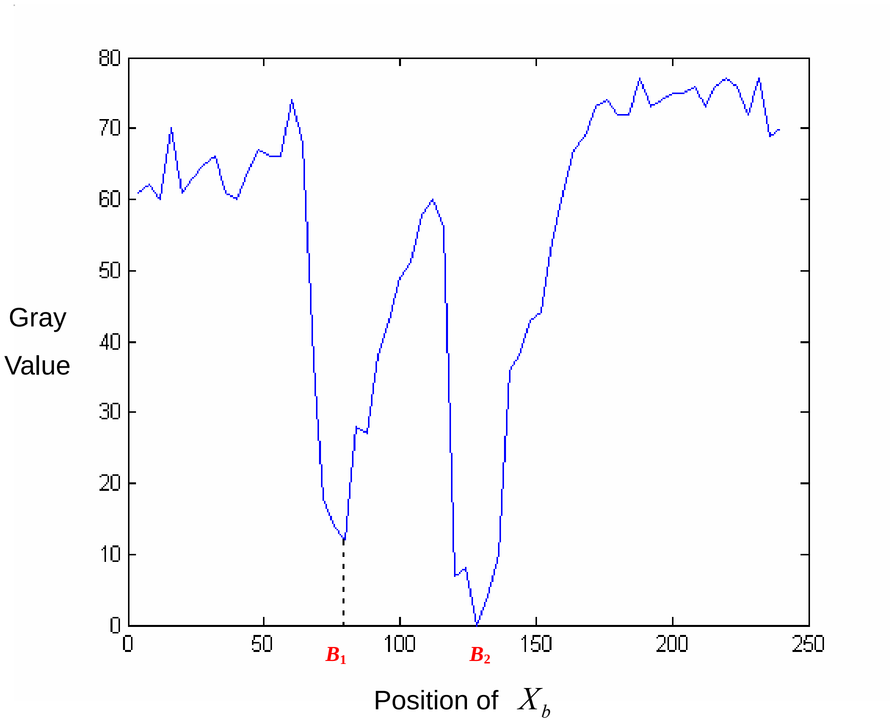
As a result, we can extract the point B_2 along line X_b by selecting the minimum gray scale values corresponding to Y axis and the minimum values must be smaller than a threshold which is about 30. Moreover, since the position of the eyebrow is about one-fourth of the facial width, we can easily search the line X_b. Then we can find the eye position corresponding to the detected valley point B_2 of a suitable scanning line X_b.


X_a
X_b
X
Y

(a)


140
120
100
80
60
40
20
0
0
50
100
150
200
250
A_1
A_2
Gray
Value
Position of X_a

(b)



(c)

Fig. 2.8. Gray-level value variations along lines in (a). (b) Along X_a. (c) Along X_b.

2.4 Sunglasses Image Enhancement

The presence of sunglasses will weaken the visibility of eyes, and therefore the performance of extracting eye position will drop off. The sunglasses may overlap with the eyes such that we could not separate the eyes from the detected eye regions easily. Sunglasses with deeper color will cause more severe interference to eye detection. For accurate eye detection, it is important and essential to solve this problem by using some methods. Using an infrared camera to catch the input images may be a possible way. But methods based on image processing techniques are more attractive to us. Consequently, in this thesis, we apply image processing techniques to eliminate the bad effect caused by sunglasses in eye detection.

2.4.1 Retinex Image Enhancement Technique

In this section, the purpose is to enhance the sunglasses region such that the eye region could be better extracted. Many image enhancement methods have been proposed. Typical methods to this end are gamma correction and gain/offset application, histogram equalization and manual histogram adjustment methods, homomorphic filtering method, and retinex image enhancement method. These methods have their different characteristics. Of these methods the retinex one is the most popular and most widely used because of its simplicity in use and its powerful ability to enhance images.

The retinex theory was first devised by Land [26]. It is sometimes also known as the Land Effect. Land's retinex theory of lightness and color constancy was one of the first computational models of an important form of perceptual constancy. Color constancy is the well known tendency for an object to always appear to have the same color, no matter what the viewing conditions are. In other words, a bright green post box appears green by daylight, by moonlight, and even under dingy street lights. According to Land, we decide the color of something by comparing its ability to reflect short, medium and long wavelengths with that of adjoining objects. Land considered that the eye and the brain (the retina and cortex) form a single optical system, which he called the retinex.

The retinex image enhancement algorithm is an automatic image enhancement method that enhances a digital image in terms of dynamic range compression, color independence from the spectral distribution of the scene illuminant, and color/lightness rendition. The digital images enhanced by the retinex image enhancement algorithm are much closer to the scene perceived by the human visual system, under all kinds and levels of lighting variations, than those enhanced by most other methods. A comparison of retinex with other image enhancement techniques can be found in [27].

Jobson *et al*. [28] defined a Single-Scale Retinex (SSR), which is an implementation of center/surround retinex. But depending on the special scale, it can either provide dynamic range compression (small scale) or tonal rendition (large scale), but not both simultaneously. Superposition of weighted different scale SSR is obvious a choice to balance these two effects. This is named Multi-Scale Retinex (MSR) [29]. The Multi-Scale Retinex (MSR) is a generalization of the Single-Scale Retinex. For color images, if the content is out of "gray world," which means the spatial averages of three color bands are far from equal, the output will be forced to be gray by MSR. This problem could be solved by introducing weight factor for different channels in Multi-Scale Retinex with Color Restoration (MSRCR) [30]. MSRCR combines the retinex dynamic range compression and color constancy with a color restoration that provides excellent color rendition.

After MSRCR, generally the outputs will be out of the range of display. Auto gain/offset can be used to shift and compressed the histogram of MSRCR outputs to the display domain.

In this thesis, we will implement SSR, MSR, and MSRCR with gain/offset. We adjust the gain/offset parameters to adjust most of the pixels values to display domain and clap small part of the values to improve the contrast.

The retinex is a member of the class of center surround functions where each output value of the function is determined by the corresponding input value (center) and its neighborhood (surround). For the retinex the center is defined as each pixel value and the surround is a Gaussian function. The mathematical form of the Single-Scale Retinex (SSR) is given by

$$R_i(x,y) = \log I_i(x,y) - \log\left[F(x,y) * I_i(x,y)\right] \qquad (2.4)$$

Where $I_i(x,y)$ is image distribution in the *i*-th color band, R is the output of the SSR process, and $*$ represents the convolution operator. $F(x,y)$ is the normalized surround function defined as

$$F(x,y) = k\ \exp\left[-\left(x^2+y^2\right)/\sigma^2\right] \qquad (2.5)$$

where k is a normalization factor given as

$$k = 1 \Big/ \left(\sum_x \sum_y F(x,y)\right). \qquad (2.6)$$

Fig. 2.9 shows an original medical image and processed image by SSR with different scales of surround. The narrow and medium surround cases are self-explanatory. The wide surround case deserves some discussion because it looks a better output image. However, the lack of dynamic range obscures the features that were visible to the observer, hence it may fail the test.

As shown in Fig.2.9, the selection of scale is related with visual angle in the direct observation. Because of the tradeoff between dynamic range compression and color rendition, we have to choose a good scale σ in the formula of *F*(*x*,*y*) in SSR. However, for different images, adaptable scales are often different. If we do not want to sacrifice either dynamic range compression or color rendition, Multi-Scale Retinex, which is a combination of weighted different scales of SSR, is a good solution. It is defined by

$$R_i(x,y)=\sum_{k=1}^{K}W_k\left(\log I_i(x,y)-\log\left[F_k(x,y)*I_i(x,y)\right]\right) \quad i=1,\ 2,\ ...,N \tag{2.7}$$

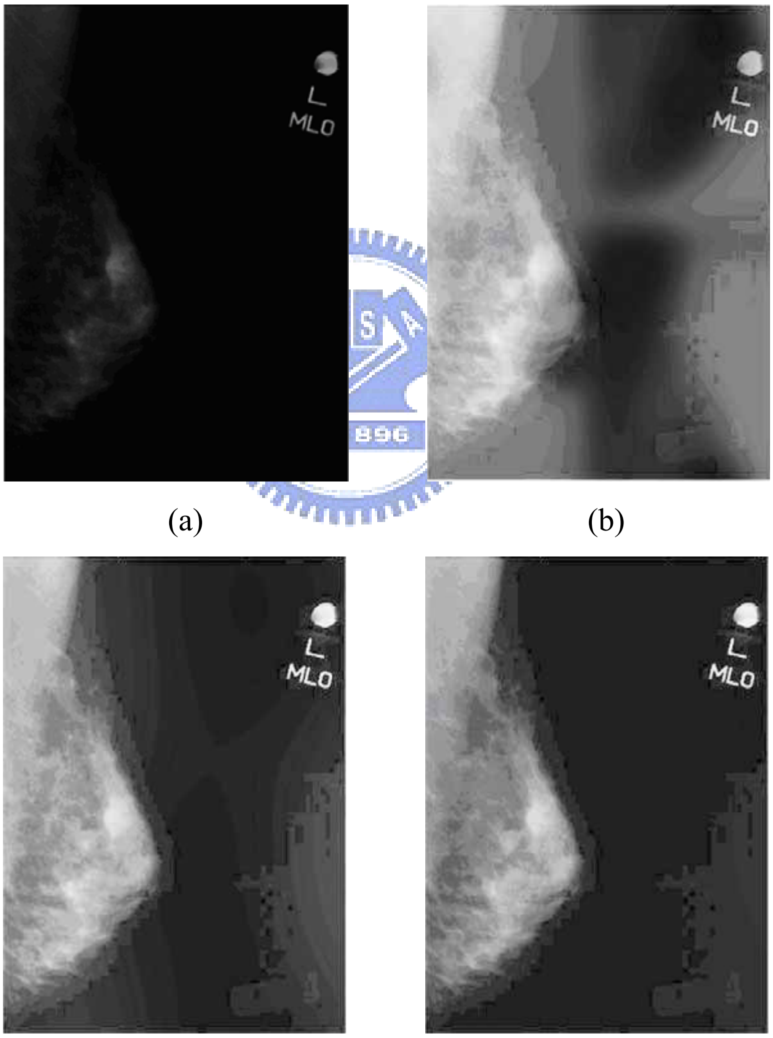
where i represents the i-th spectral band, N is the number of spectral bands—$N = 1$ for grayscale images and $N = 3$ for typical color images. In the latter case, $\mathrm{i} \in \mathrm{R,G,B}$. $R(x,y)$ is the output of the MSR process, W_k are the weights associated with F_k, K is the number of surround functions, or scales. F_k represents the k-th surround function and is defined as:

$$F_k(x,y)=k\,\exp\left[-\left(x^2+y^2\right)/\sigma_k^2\right] \tag{2.8}$$

where σ_k are the scales that control the extent of the surround function and the amount of spatial detail that is retained. Fig. 2.10 is the MSR result of previous medical examples. The MSR processed image uses features from all the three scales to provide simultaneous dynamic range and tonal rendition. It has significant dynamic range compression in the boundary between the lighted parts and dark parts, and reasonable color rendition in the whole image scale.

Actually, the suitable number of scales needed by the MSR is application dependent. However, experiments showed that three scales respectively representing narrow, medium, and wide surrounds are often enough for most of the images. The weights can be selected equal or adjusted to weight more on dynamic range compression or color rendition.

Fig. 2.11 shows another color example processed by SSR with different surround scales and then by MSR. Similar to Fig. 2.9, none of the individual scale attains the goal that we are trying to achieve. The narrow surround acts as a high-pass filter, capturing more fine details in the image but at a severe loss of tonal information. The wide surround captures more fine tonal information but at the loss of dynamic range. The medium surround captures gets a balance between dynamic range and tonal information. The MSR is the average of the three renditions and has the characters close to the medium surround scale result.



(a) (b)

(c) (d)

Fig. 2.9. SSR with different scales. (a) Original image. (b) SSR with narrow surround scale 15. (c) SSR with medium surround scale 80. (d) SSR with wide surround scale 250.

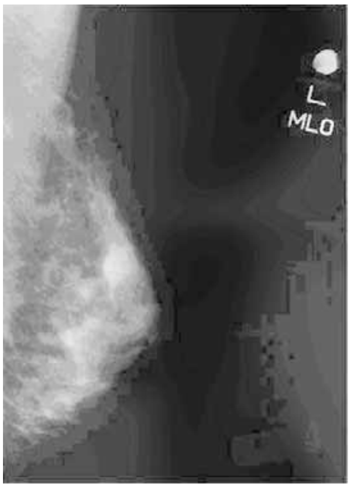


Fig. 2.10. Result of MSR with scales = 15, 80, and 250.

Observing Fig. 2.11, we can find that the color rendition of the results of SSR and MSR have a certain degree of deviation from the original image. They look close to gray images. Actually, bad performance for color images is the weakness of SSR and MSR. MSR is good for gray images. But it could be a problem for the color images because it does not consider the relative intensity of color bands. This can be seen from formula of MSR, whose output is the relative reflectances in the special domain. Considering the images "out of gray world", whose average intensity for three color band are far from equal, the output of MSR for the three channels will be more close, which make it looks more gray. The solution to this problem is to utilize weights for three color channels respectively depending on the relative intensity of the three channels in the original images. A color restoration factor is computed as the following form:

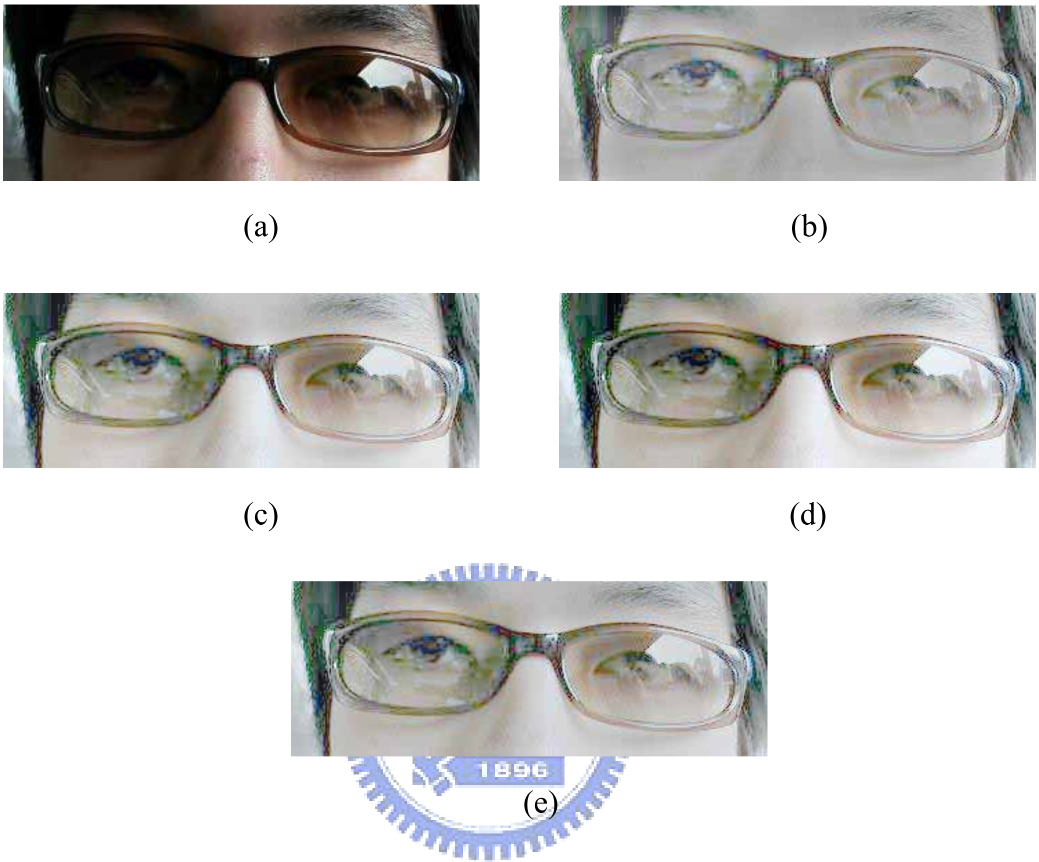
$$C_i(x,y) = \beta \log\left[\alpha \ I_i(x,y) \Big/ \sum_{n=1}^{N} I_n(x,y)\right] \quad (2.9)$$

where $C_i(x,y)$ is the color restoration coefficient in the i-th spectral band, N is the number of spectral bands, I_i is the i-th spectral band in the input image, β is gain constant, and α controls the strength of non-linearity. Analog to the spatial operation of the retinex which utilizes a log operator, the internal form of the color restoration process and the retinex process is essentially the same. Combining Eq. (2.9) with Eq. (2.7), the MSRCR is given by

$$R_i(x,y) = C_i(x,y)\sum_{k=1}^{K} W_k\left(\log I_i(x,y) - \log\left[F_k(x,y) * I_i(x,y)\right]\right) \quad i = 1,\ 2,\ \ldots, N \qquad (2.10)$$

An integral scheme of MSRCR is given in Fig. 2.12. In order to observe the effect of MSRCR for color images, we apply MSRCR on an input image the same as shown in Fig. 2.11(a), the result is shown in Fig. 2.13. Comparing Fig. 2.13 with Fig. 2.11, we can easily find the result of MSRCR has better color rendition close to the original image than SSR or MSR.

When there are reflections on glasses or sunglasses, some influences on the detection for eyes will be caused. More serious reflections will cause more terrible interference whether the sunglasses region is processed by image enhancement techniques or not. As a serious reflection overlaps the eyes fortuitously, the accuracy of eye detection will drop off critically. Solving this problem is quite essential for an eye detection system. In Chapter 3, we will further discuss these conditions and attempt to utilize some methods to overcome them.



(a)

(b)

(c)

(d)

(e)

Fig. 2.11. (a) Original image. (b) Narrow surround (scale = 15). (c) Medium surround (scale = 80). (d) Wide surround (scale = 250). (e) MSR output with scales = 15, 80, and 250.

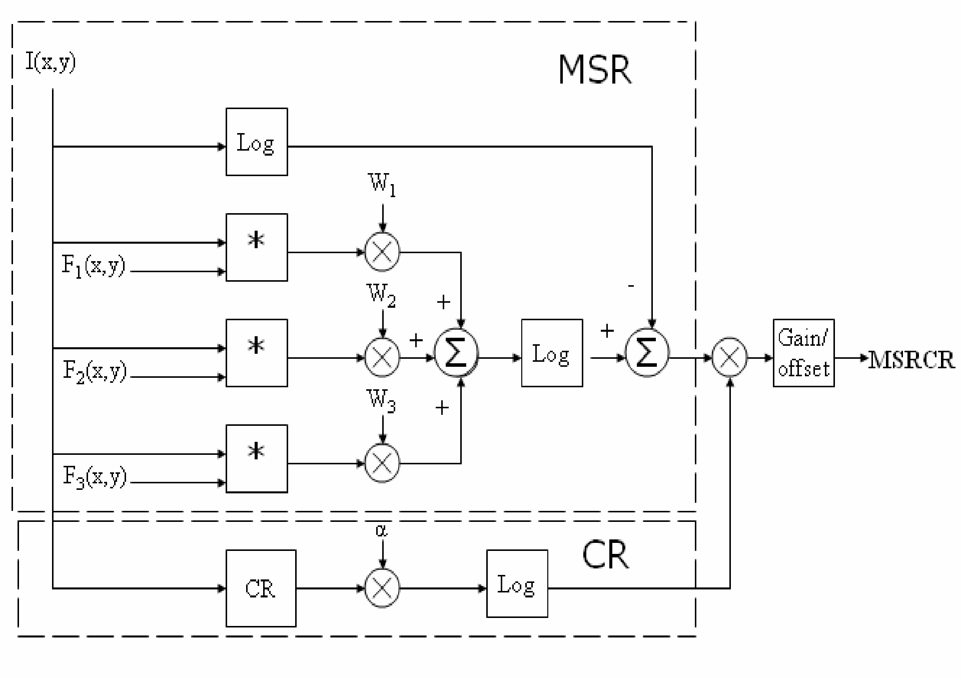
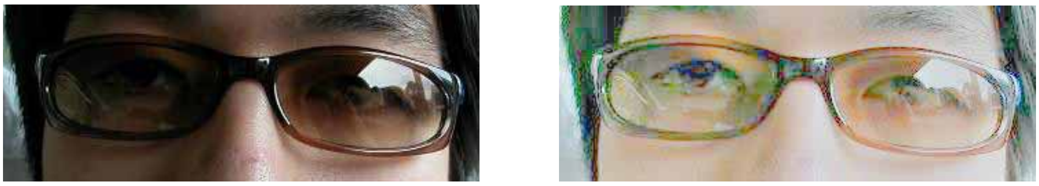


Fig. 2.12. Integral scheme of MSRCR.



(a) (b)

Fig. 2.13. (a) Original image. (b) MSRCR output with scales = 15, 80, and 250.

2.4.2 Histogram Equalization Enhancement Technique

In order to analyze the performance of the retinex image enhancement techniques, we compare the retinex result with another image enhancement technique: histogram equalization technique. Histogram equalization technique is based on the idea of remapping the histogram of the image to a histogram that has a near-uniform probability density function. This will result in reassigning dark regions to brighter values and bright regions to darker values. Consequently, the histogram equalization technique deeply depends on the distribution of gray scale of input images. The probability of occurrence of gray level r_k in an image is defined by

$$p_r(r_k) = \frac{n_k}{n} \qquad k = 0,1,\ldots,L-1 \tag{2.11}$$

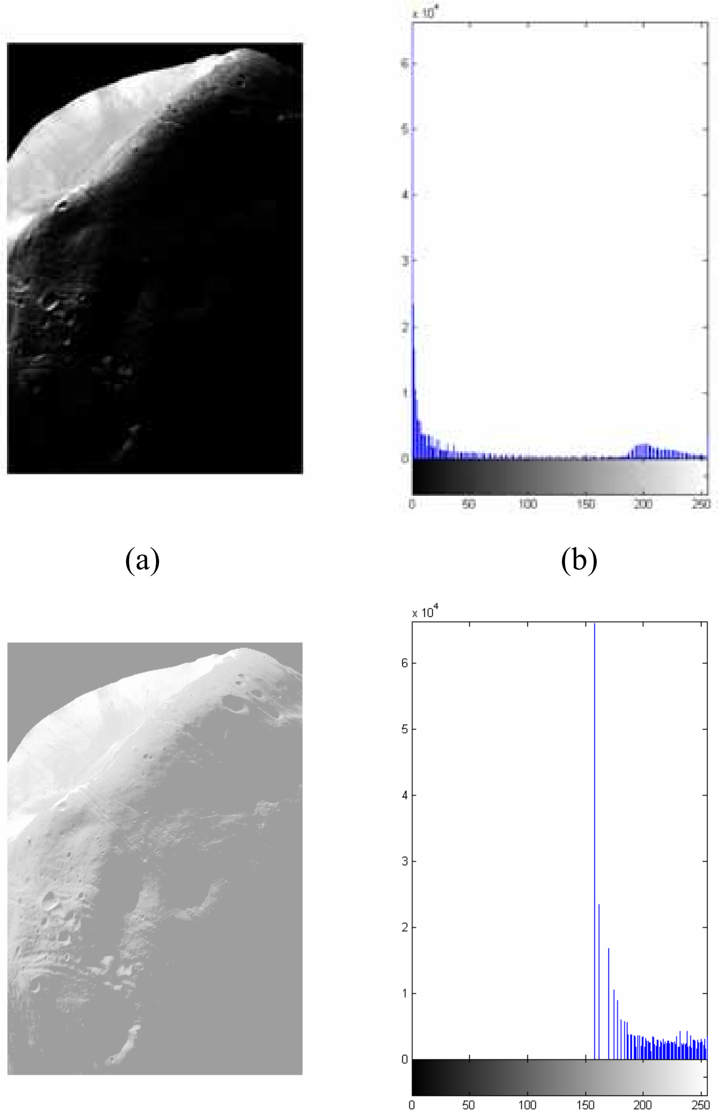
where n is the total number of pixels in the image, n_k is the number of pixels that have gray level r_k, and L is the total number of possible gray levels in the image. The transformation function of Histogram equalization is given as

$$s_k = T(r_k) = \sum_{j=0}^{k} p_r(r_j) = \sum_{j=0}^{k} \frac{n_j}{n} \qquad k = 0,1,\ldots,L-1. \tag{2.12}$$

A processed image is obtained by mapping each pixel with gray level r_k in the input image into a corresponding pixel with gray level s_k in the output image.

Histogram equalization works well for scenes that have unimodal or weakly bi-modal histograms, i.e., very dark, or very bright ones, but it is not effective to those images with strongly bi-modal histograms, i.e., images containing very dark and very bright regions simultaneously. Fig. 2.14(a) shows a bi-mode image, and Fig. 2.14(b) shows its histogram. The image includes large, dark areas, and hence its histogram is characterized by a large concentration of pixels in the dark end of the gray scale. One might think that histogram equalization would be a good approach to enhance this image, so that details in the dark regions would become more visible. However, the result in Fig. 2.14(c) shows that histogram equalization in fact did not produce a particularly good result in this case. The reason for this can be seen by studying the histogram of the equalized image shown in Fig. 2.14(d). We can see that the intensity levels have been shifted to the upper one-half of the gray scale, thus giving the image a washed-out appearance. The cause of the shift is the large concentration of dark components at or near 0 in the original histogram. The cumulative transformation function obtained from this histogram is steeply increasing, and thus mapping the large concentration of pixels in the low end of the gray scale to the high end of the scale. It should be better to re-do histogram equalization once for a bi-modal image Fig. 2.14(c), the resulting image is shown in Fig. 2.15(c), compared with the MSRCR result shown in Fig. 2.15(d). Figs. 2.16–2.19 show four examples of eyes images processed by histogram equalization and MSRCR, respectively. As we can see, the MSRCR provided the better overall visual quality. By our experience, eye region enhancement by histogram equalization does not perform consistently, i.e., sometimes good and sometimes bad. However, the MSRCR technique performs constantly well.

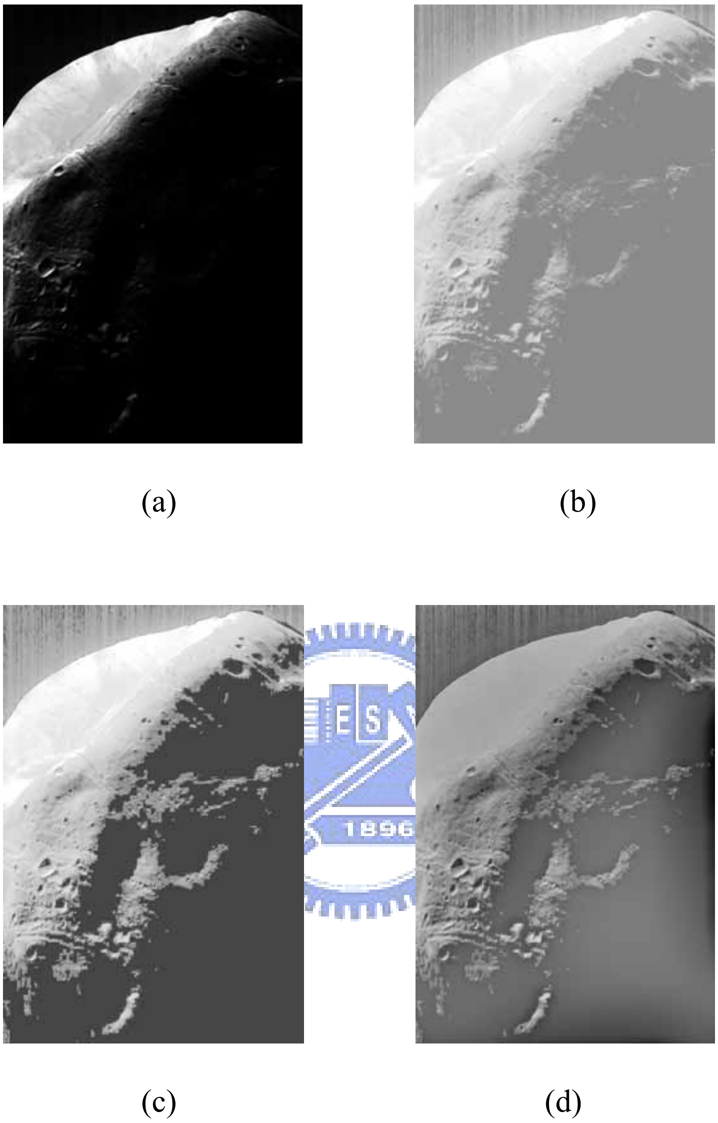
.



(a) (b)

(c) (d)

Fig. 2.14. Illustration of histogram equalization. (a) Original image. (b) Histogram of (a). (c) Image processed by histogram equalization. (d) Histogram of (c).



(a) (b)

(c) (d)

Fig. 2.15. A comparison of histogram equalization and the MSRCR. (a) Original image. (b) Histogram equalization result of (a). (c) Histogram equalization result of (b). (d) MSRCR result.

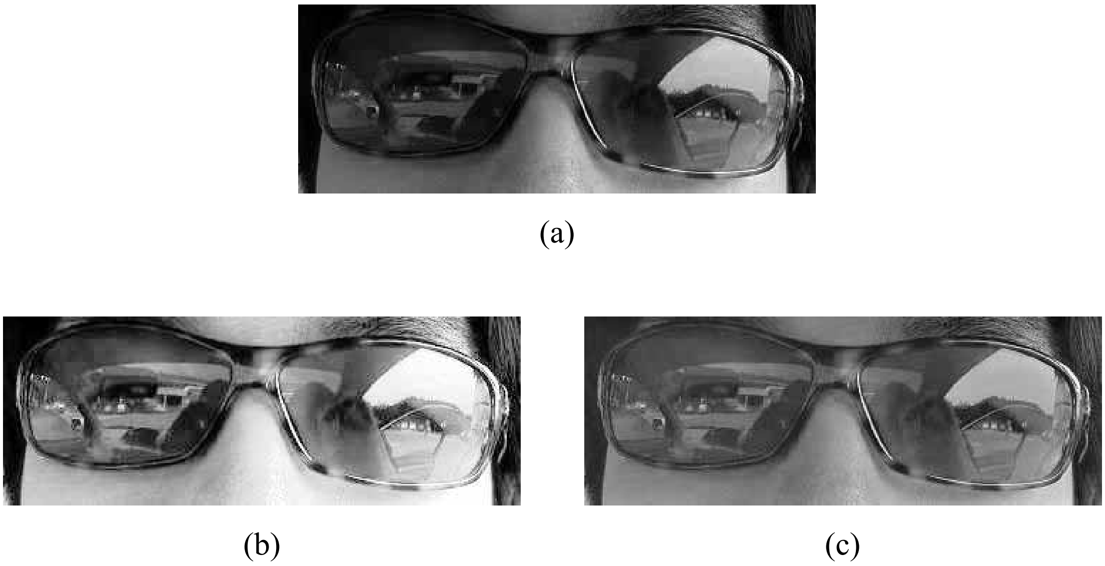


Fig. 2.16. A comparison of histogram equalization and the MSRCR. (a) Original image. (b) Histogram equalization result. (c) MSRCR result.

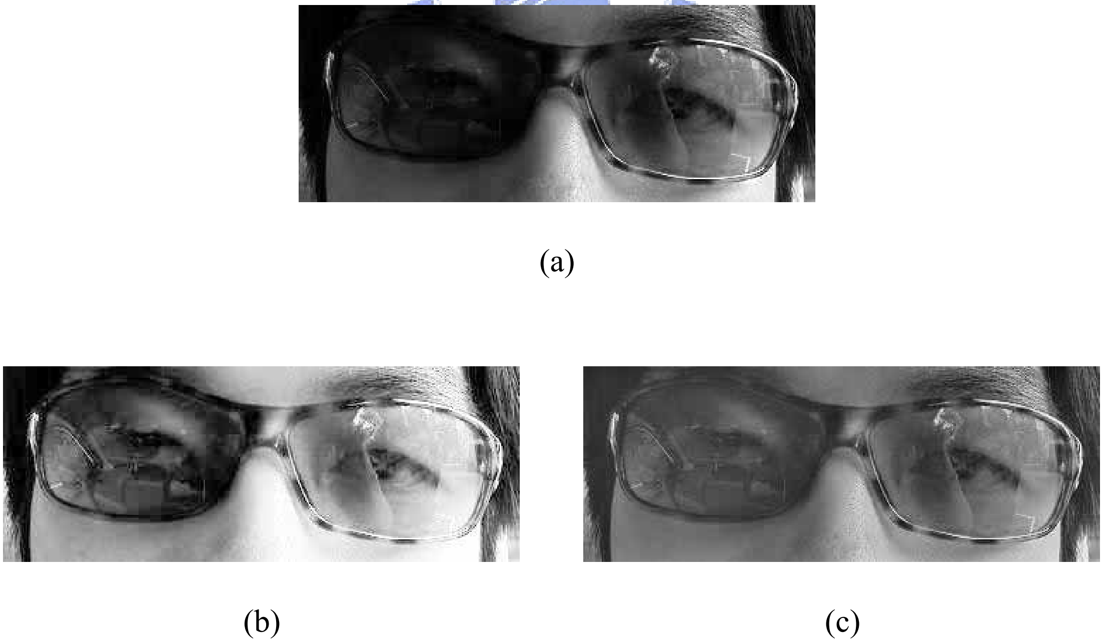
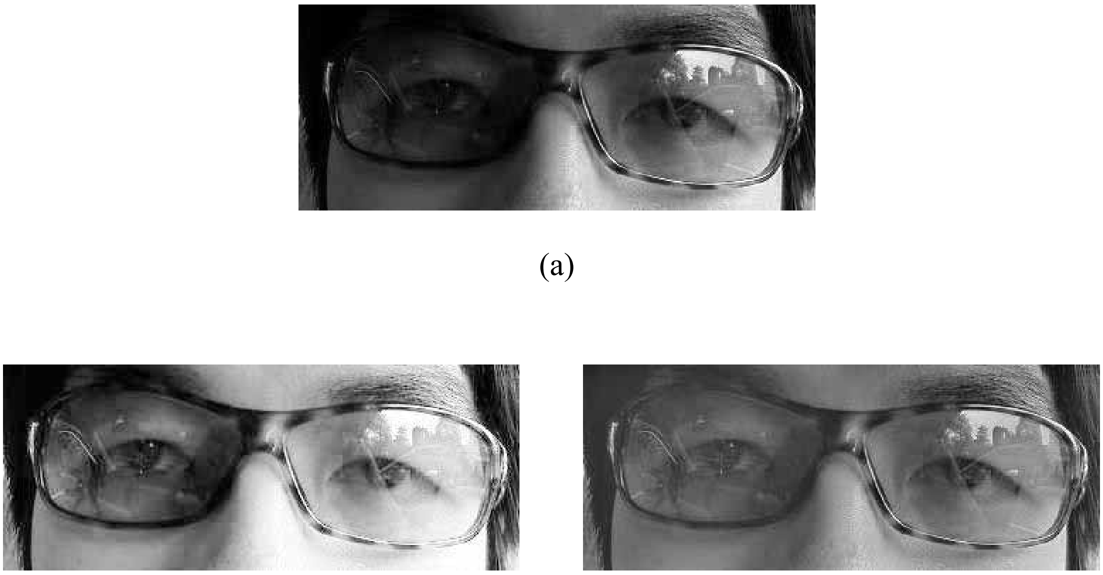


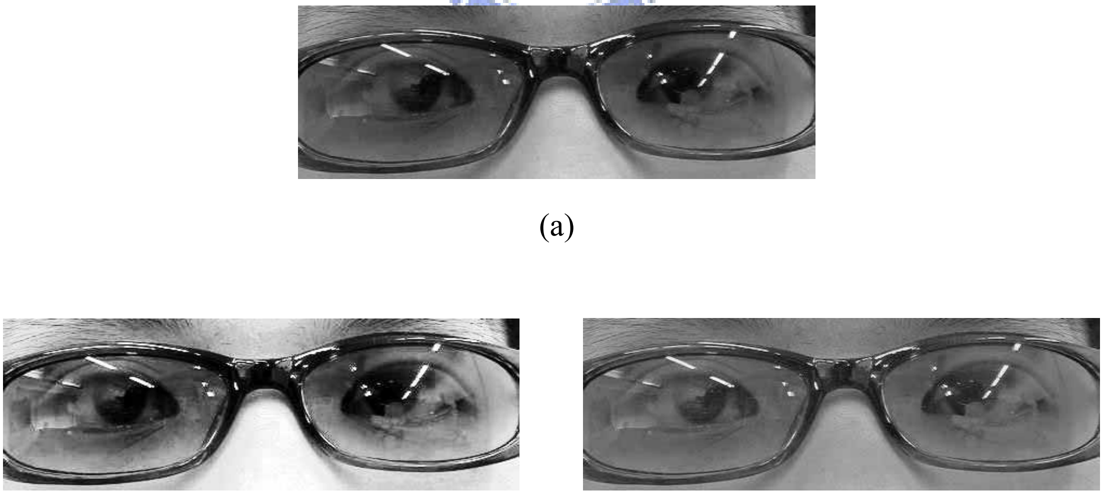
Fig. 2.17. A comparison of histogram equalization and the MSRCR. (a) Original image. (b) Histogram equalization result. (c) MSRCR result.



(a)

(b) (c)

Fig. 2.18. A comparison of histogram equalization and the MSRCR. (a) Original image. (b) Histogram equalization result. (c) MSRCR result.



(a)

(b) (c)

Fig. 2.19. A comparison of histogram equalization and the MSRCR. (a) Original image. (b) Histogram equalization result. (c) MSRCR result.

Chapter 3

Reflection Separation

3.1 Image with Reflection

When we view a scene through a transparent glass, the image is often similar to those shown in Fig. 3.1. Perceptually, we can view each one of these images as a superposition of two images: the foreground and the reflection. Hence each image could be decomposed into two transparent layers. We need a computer vision algorithm to find this decomposition. Mathematically, the problem can be modeled as follows. Given an image $I(x,y)$, we wish to find two layers, I_1 and I_2 such that:

$$I(x,y) = I_1(x,y) + I_2(x,y) \tag{3.1}$$

This problem is obviously difficult because there are two variables but only one equation. If we have no additional prior knowledge, there will be an infinite number of possible decompositions. In this chapter, we adopt an algorithm that can separate reflections from images using a single input image. The algorithm is based on a cost function: it favors decompositions which have a small number of edges and corners.

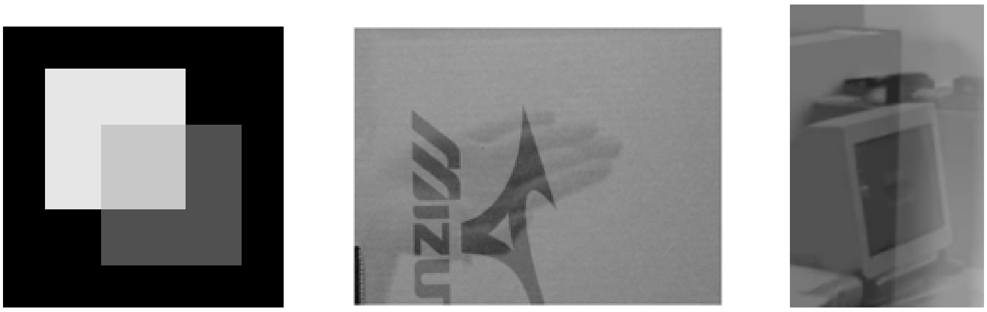


Fig. 3.1. Some examples for images with reflections.

3.2 Cost Function, Edge and Corner

Consider a simple image which is the superposition of two squares in Fig. 3.2(a). We want to decompose the image into two layers. There will be an infinite number of possible decompositions. Figs. 3.2(b)–(e) show some possible decompositions including the perceptually "correct" decomposition (Fig. 3.2(e)). What rule is the "correct" decomposition based on? One reason is that it has the smallest total number of edges and corners among the decompositions shown in Fig. 3.2. The original image has ten corners: four corners from each square and two corners caused by the superposition of the two squares. When we decompose the image into two squares, the two corners caused by the superposition disappeared and there are just eight corners left. The decomposition shown in Figs. 3.2(b) and 3.2(d) increase the number of corners and edges. Clearly, we can also see that the decomposition shown in Fig. 3.2(e) has smaller total number of edges and corners than the other decompositions in Fig. 3.2, and it looks an appropriate decomposition perceptually.

How do we translate the preference for a small number of edges and corners into a cost function? We need operators used for edge and corner detectors; besides, we also need a mathematical form to give the cost of an image. Next we will describe how to define a cost function based on natural statistics of natural scenes.

A remarkably robust property of natural images that has received much attention lately is the fact that when derivative filters are applied to natural images, the filter outputs tend to be sparse [31], [32]. Fig. 3.3 can illustrate this fact. We take two arbitrary examples of natural images and apply a horizontal derivative filter to them respectively. We can see that the histograms of their derivative filter outputs are peaked at zero and fall off much faster than a Gaussian. As a result, the derivative filter outputs are concentrated at zero and therefore are sparse. Similar histograms are observed for vertical derivative filters and for the gradient magnitude: $|\nabla I|$.

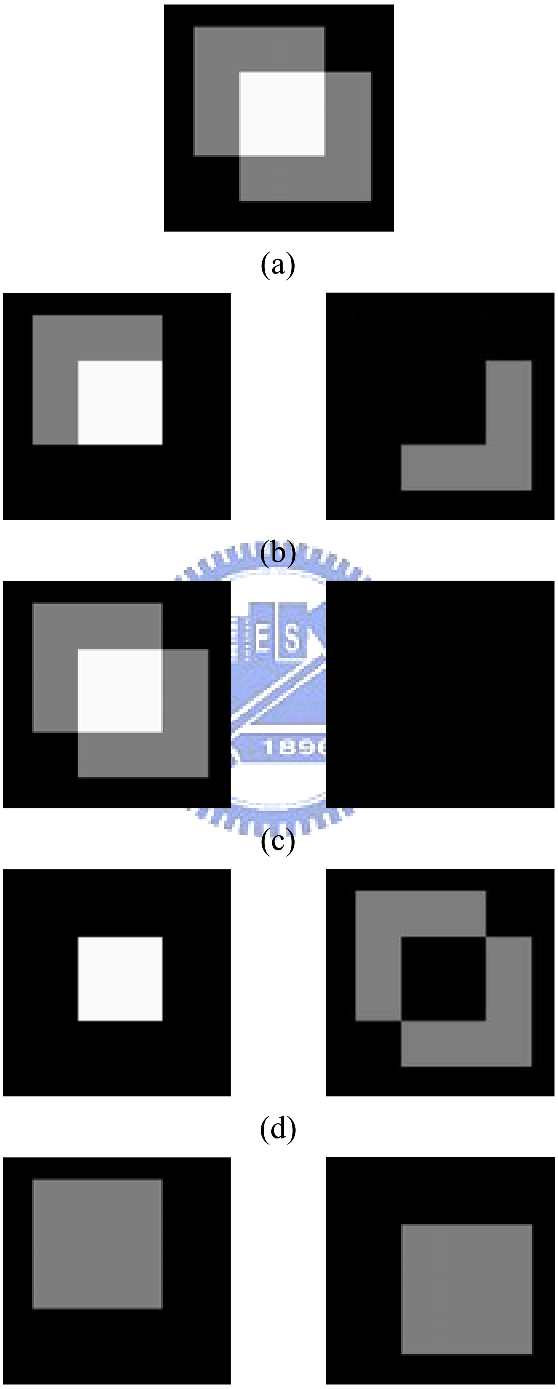
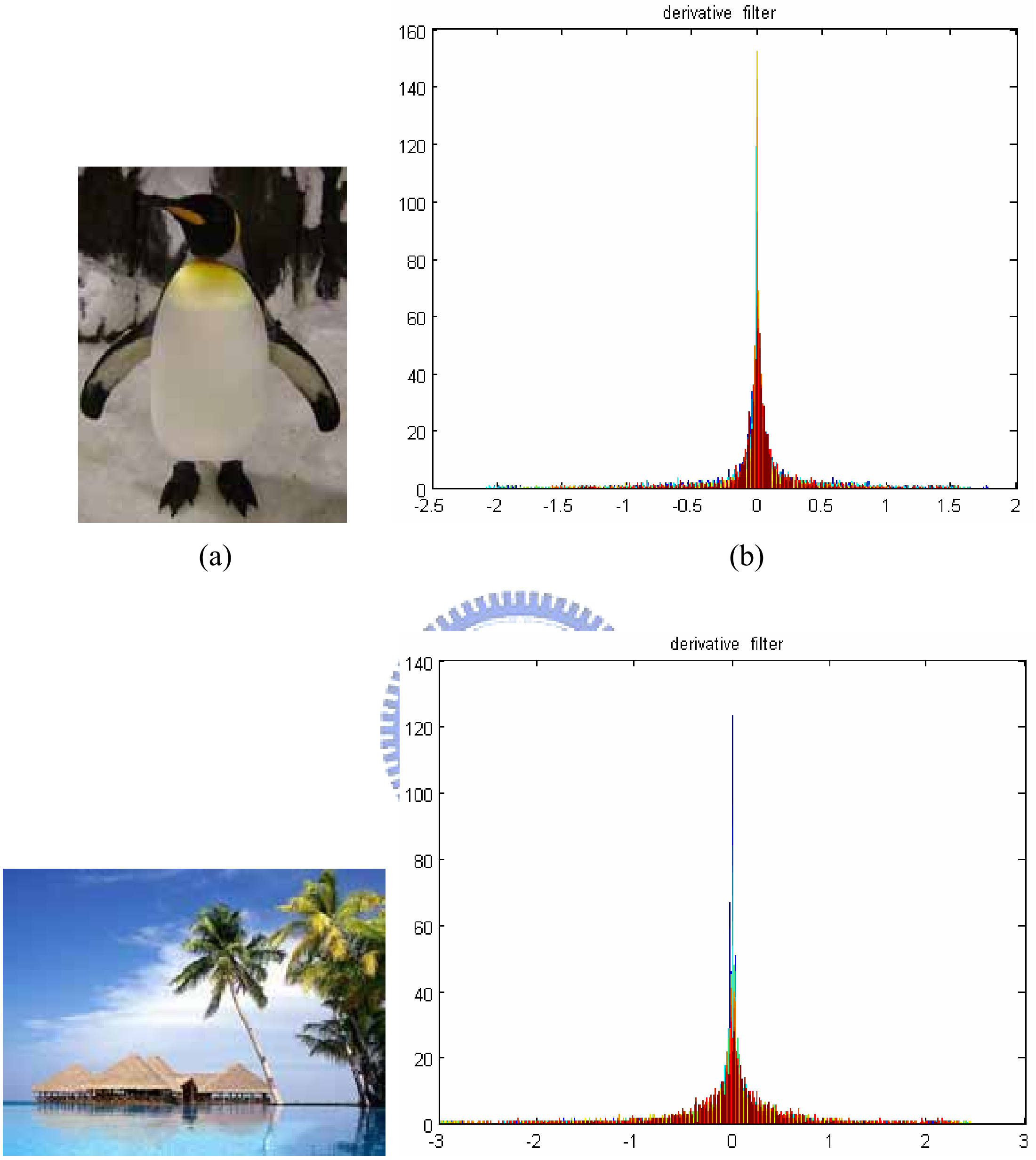

(a)
(b)
(c)
(d)
(e)

Fig. 3.2. An input image and some decompositions. (a) The original input image. (b)–(d) Some possible decompositions. (e) The perceptually correct decomposition.



(a) (b)

(c) (d)

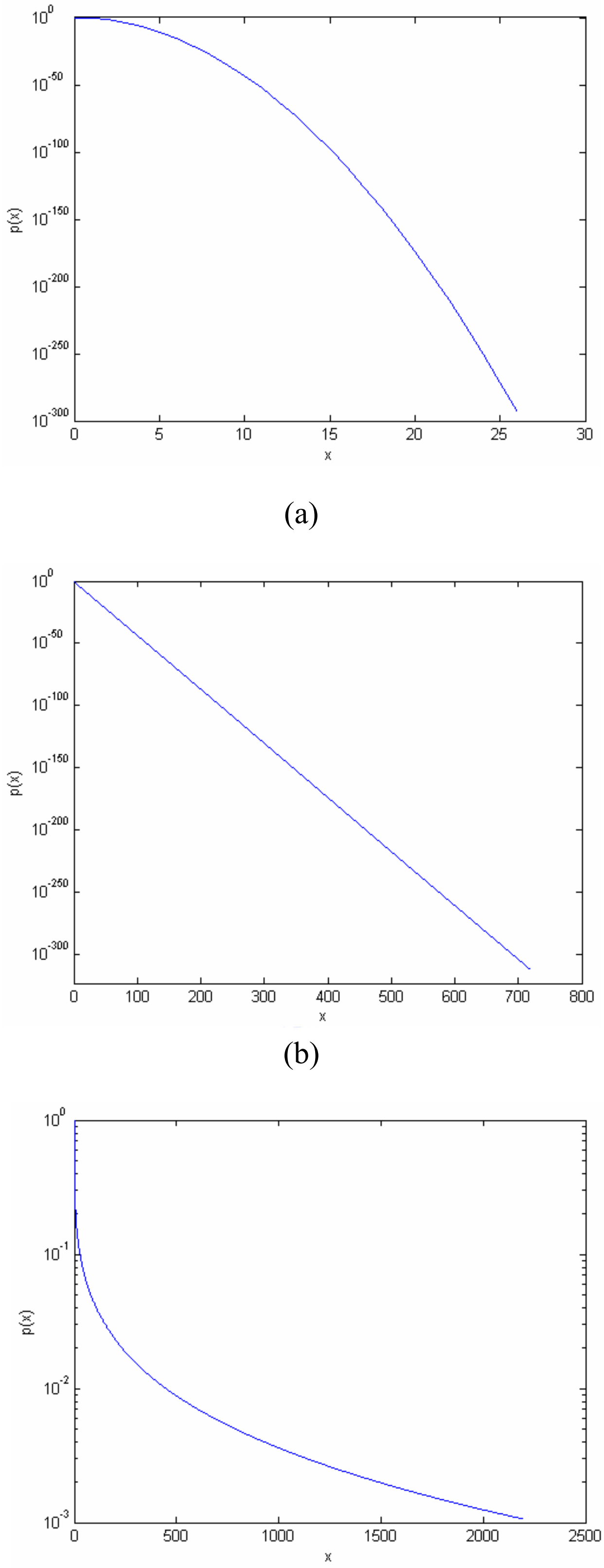
Fig. 3.3. Two natural images and their filter derivative output diagrams. (a), (c) Natural image. (b), (d) Histogram of derivative filter outputs.

Subsequently we convert each of the filter outputs of Fig. 3.3 into a log histogram type and show the results in Fig. 3.5. Observing Fig. 3.5, we can find that the distributions are similar to an exponential density with exponent less than 1. For comparison, we show the log probability for densities of the form $p(x) = e^{-x^{\alpha}}$ which is presented in [33]. We plot the corresponding log probabilities for $\alpha > 1$, $\alpha = 1$ and $\alpha < 1$, respectively for $x \geq 0$. The results are shown in Fig. 3.4. Comparing Figs. 3.5(b) and 3.5(d) with Fig. 3.4, it can be found that the natural statistics for derivative filters has the qualitative nature of a distribution $e^{-x^{\alpha}}$ with $\alpha < 1$. Similar to derivative filters, the gradient magnitude also has the character. When we define a cost function, we will use this character of derivative filters and gradient magnitude operators. More descriptions about edge detector and gradient magnitude are in Sec. 3.2.1.

Now we consider the other operator, "corner detector." In this paper we use a Harris-like operator *c(x, y)* as a corner detector. There are more detailed descriptions about Harris corner detector [34] in Sec. 3.2.1. Here we first treat the qualitative statistic of the operator. The output of the detector at a given location x_0, y_0 is defined as:

$$c(x_0, y_0) = \det\left(\sum w(x,y)\begin{pmatrix} I_x^2(x,y) & I_x(x,y)I_y(x,y) \\ I_x(x,y)I_y(x,y) & I_y^2(x,y) \end{pmatrix}\right) \tag{3.2}$$

where $w(x,y)$ is a small Gaussian window around x_0, y_0, and I_x, I_y are the derivatives of an image *I*.



(a)

(b)

(c)

Fig. 3.4. The log probability for densities of the form $p(x) = e^{-x^{\alpha}}$. (a) $\alpha > 1$ ($\alpha = 2$). (b) $\alpha = 1$. (c) $\alpha < 1$ ($\alpha = 0.25$).

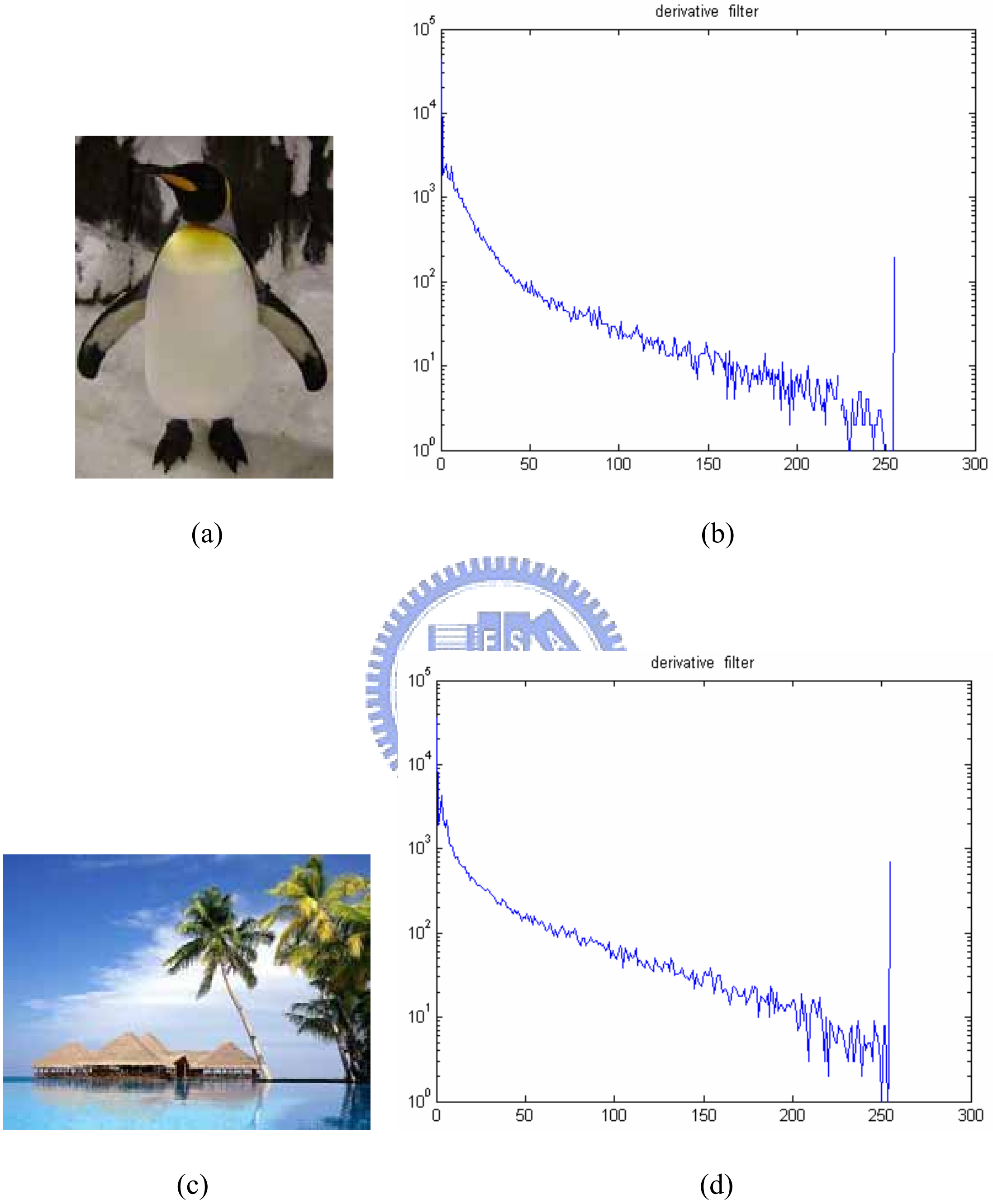


Fig. 3.5. Two natural images and their filter derivative output diagrams. (a), (c) Natural images. (b), (d) Log histograms of derivative filter outputs.

Similarly, we show the histograms of this corner operator for the two natural images given in Fig. 3.3. The results are shown in Fig. 3.6. They are also sparse. Again, we convert each of the outputs of Fig. 3.6 into a log histogram type in Fig. 3.7. Likewise, we compare Fig. 3.7 with Fig. 3.4 and then find the same character as derivative filters and gradient magnitude operators. The corner detector also has the qualitative statistic of a distribution $e^{-x^{\alpha}}$ with $\alpha < 1$.

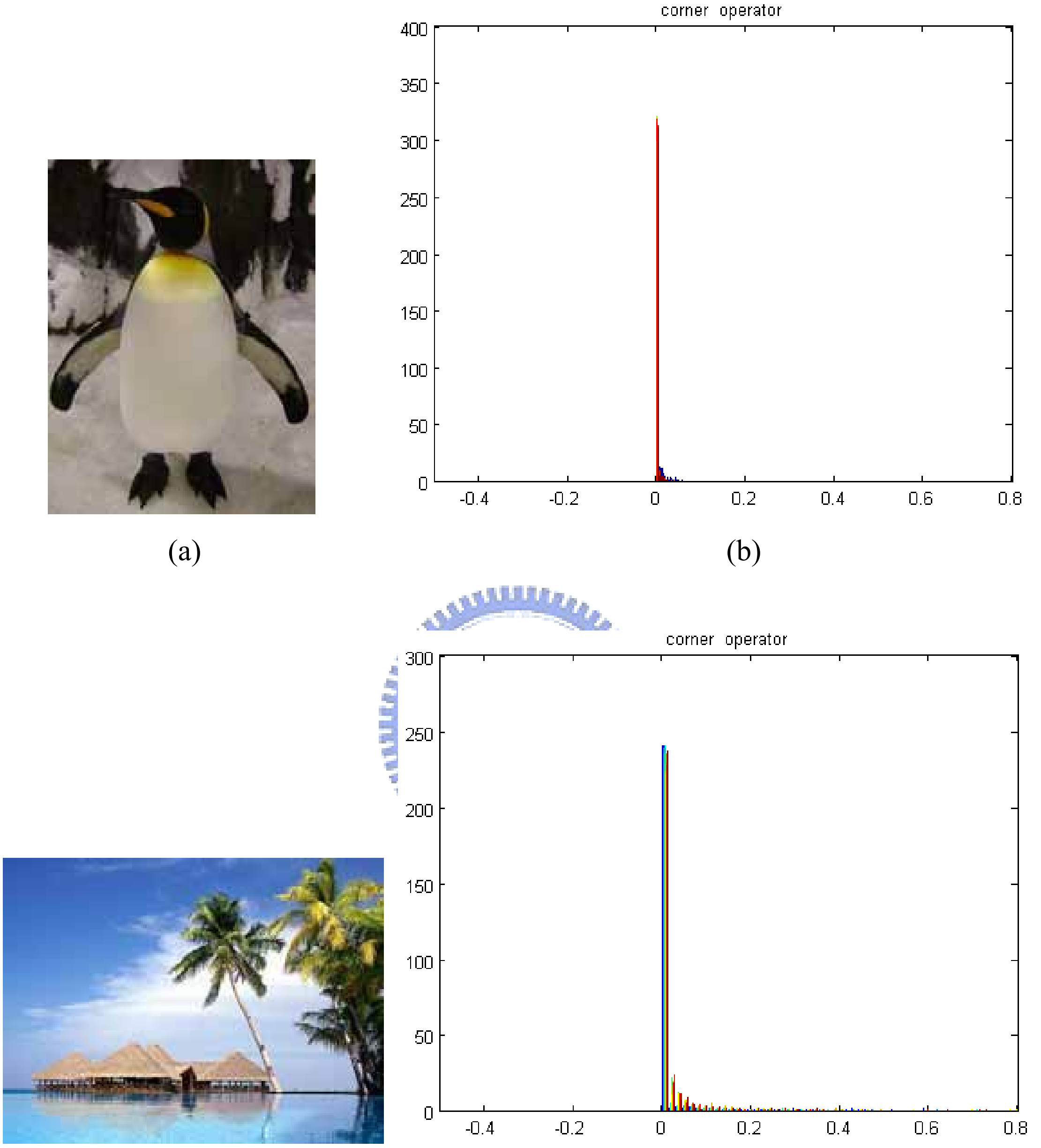
By applying the gradient magnitude and corner detectors described above on a number of images, it can be found that the histograms shown in Fig. 3.3, Fig. 3.5, Fig. 3.6, and Fig. 3.7 are typical. For both gradients and corner detectors the exponent was less than 1 and the exponent for the corner detector was smaller than that of the gradients. Typical exponents are 0.7 for the derivative filter and 0.25 for the corner detector [33].

The qualitative statistics observed in natural images motivated the type of the cost function. The histograms of these operators can be fit with a generalized Gaussian distribution $p(x) \propto e^{-x^{\alpha}}$. By using the negative log probability of these operators on natural images, the cost function for a single layer is defined as:

$$\cos t(I) = \sum_{x,y} \left|\nabla I(x,y)\right|^{\alpha} + \eta c(x,y;I)^{\beta} \tag{3.3}$$

with $\alpha = 0.7$, $\beta = 0.25$ which are obtained from the qualitative statistic of the histograms of the operators (gradient and corner operators) in natural images, and $\eta = 15$ which is determined by the ratio of the scaling parameters in the corner and gradient distributions [33]. The cost for a two layer decomposition is the sum of the costs for the two decomposed layers:

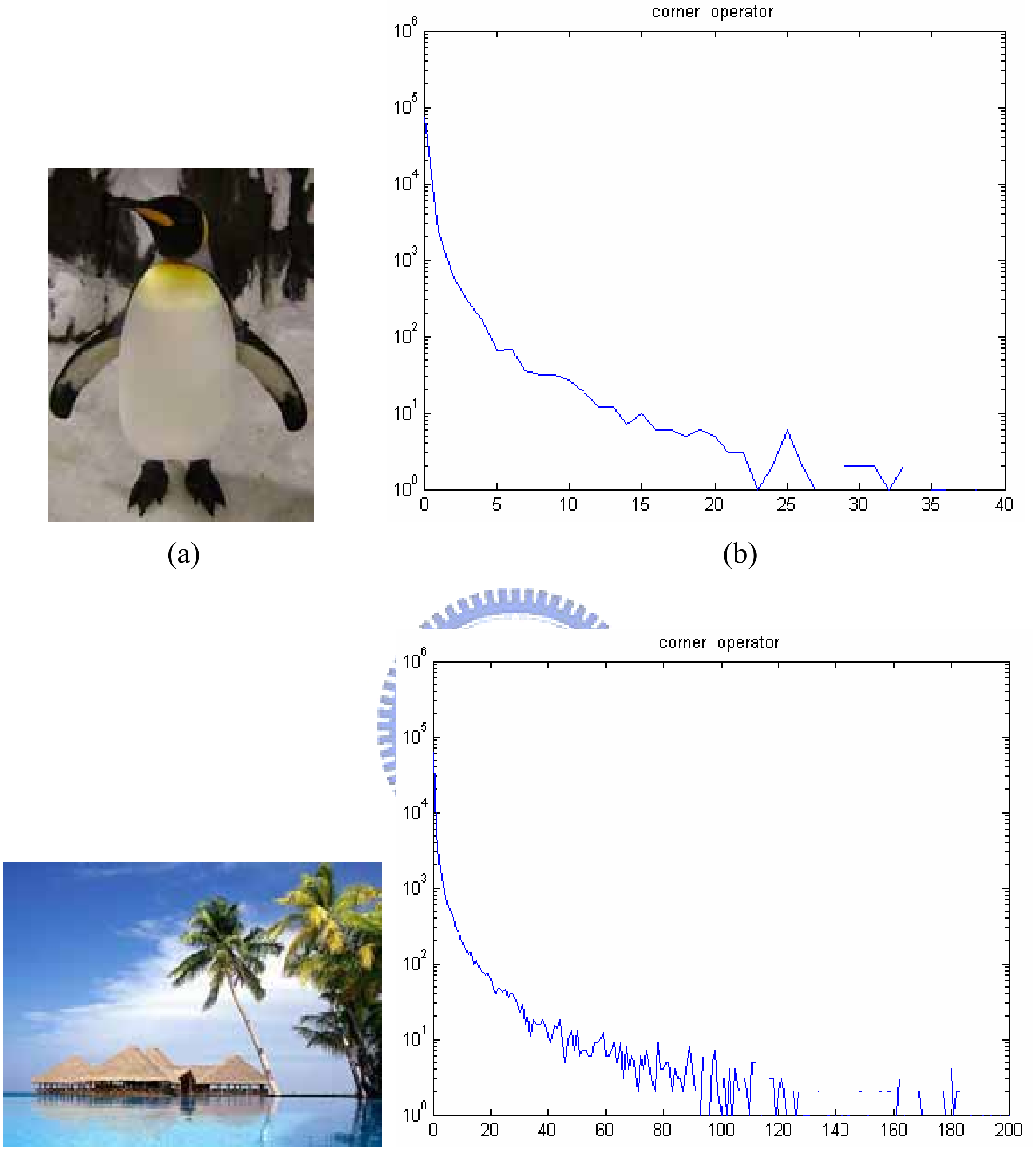
$$\cos t(I_1,\, I_2) = \cos t(I_1) + \cos t(I_2) \tag{3.4}$$



(a) (b)

(c) (d)

Fig. 3.6. Two natural images and their corner detector output diagrams. (a), (c) Natural images. (b), (d) Histograms of corner detector outputs.



(a) (b)

(c) (d)

Fig. 3.7. Two natural images and their corner detector output diagrams. (a), (c) Natural images. (b), (d) Log histograms of corner detector outputs.

Now we give an example to test the accuracy of the cost function. We apply the cost function to the image and some possible decompositions of it shown in Fig. 3.2. The cost values of them are evaluated and shown in Fig. 3.8. Clearly, we can find that the perceptually correct decomposition in Fig. 3.8(e) indeed has smaller cost than the other decompositions in Fig. 3.8. Of course, these decompositions are just a small amount out of an infinite number of possible decompositions. In Chapter 4, we will show some examples of one dimensional decompositions.

3.2.1 Edge Detector and Corner Detector

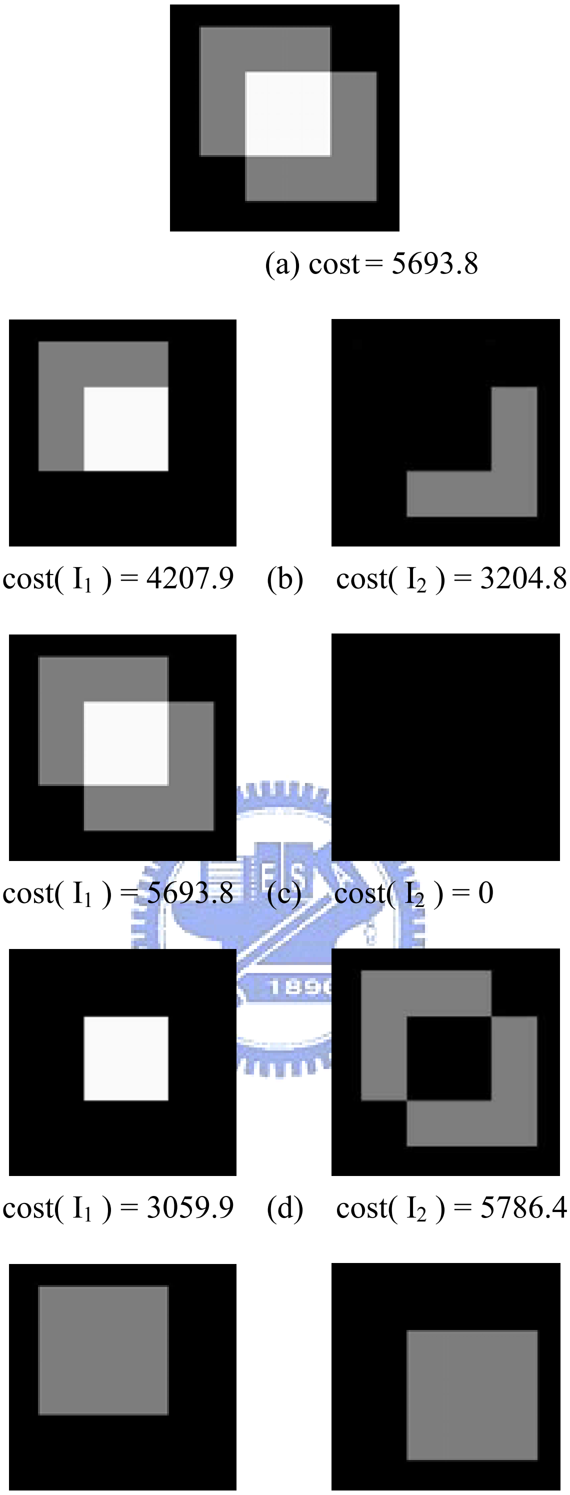
For the cost function defined above, the derivative and corner operators are essentials. Image edges have already been defined as local variations of image intensity. Therefore, local image differentiation techniques can produce edge detector operators. The gradient of an image at location (x, y) is defined as

$$\nabla f(x,y)=\begin{bmatrix}\dfrac{\partial f}{\partial x} & \dfrac{\partial f}{\partial y}\end{bmatrix}^{\mathrm{T}} \tag{3.5}$$

where the partial derivatives $\partial f/\partial x$ and $\partial f/\partial y$ can be used to detect the perpendicular and horizontal edges respectively in an image. Its magnitude is given as

$$\left|\nabla f(x,y)\right|=\sqrt{\left(\frac{\partial f}{\partial x}\right)^2+\left(\frac{\partial f}{\partial y}\right)^2} \tag{3.6}$$

which can be used as an edge detector. Fig. 3.9 shows two common gradient operator masks, the left masks are $\partial f/\partial y$, the right ones are $\partial f/\partial x$. Fig. 3.10(b) shows a result of a simple image by convolving it with the Prewitt masks and then using Eq. (3.6) to determine the gradient magnitude.



(a) cost = 5693.8

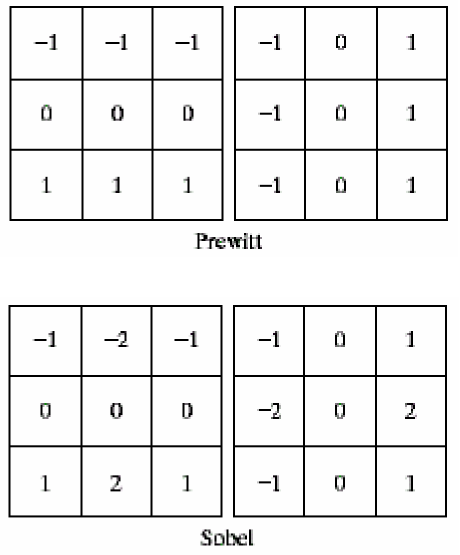
cost(I_1) = 4207.9 (b) cost(I_2) = 3204.8

cost(I_1) = 5693.8 (c) cost(I_2) = 0

cost(I_1) = 3059.9 (d) cost(I_2) = 5786.4

cost(I_1) = 2380.1 (e) cost(I_2) = 2762.7

Fig. 3.8. Cost values for an input image and some decompositions. (a) Cost = 5693.8. (b) Cost = 4207.9 + 3204.8 = 7412.7. (c) Cost = 5693.8 + 0 = 5693.8. (d) Cost = 3059.9 + 5786.4 = 8846.3. (e) Cost = 2380.1 + 2762.7 = 5142.8.



−1	−1	−1
0	0	0
1	1	1

−1	0	1
−1	0	1
−1	0	1

Prewitt

−1	−2	−1
0	0	0
1	2	1

−1	0	1
−2	0	2
−1	0	1

Sobel

Fig. 3.9. Sobel and Prewitt edge detector masks.

Now we discuss the other operator — corner detector. One of the first approaches to find corners was to segment the image into regions, extracting the boundaries as a chain code, and then identify corners as points where boundary direction changes rapidly. This approach has been largely abandoned as it relies on the previous segmentation step, which is a complex task itself, and is also computationally expensive.

Harris and Stephens [34] presented a corner detector which is widely used today. The Harris corner detector first computes a corner response function. Corners are detected by finding the local maxima in the corner response function. The program returns the corners in the decreasing order of their corner response function value. A Harris matrix is defined as

$$\mathrm{A}=\begin{bmatrix} \langle I_x^2 \rangle & \langle I_x I_y \rangle \\ \langle I_x I_y \rangle & \langle I_y^2 \rangle \end{bmatrix} \tag{3.7}$$

where $\langle\ \rangle$ is the smoothing by a 2D Gaussian function. Harris corner detector procedures are:

1) For each pixel find corresponding A.
2) For each A find two eigenvalues $\lambda_{\max}$ and $\lambda_{\min}$.
3) Sort all $\lambda_{\min}$, discard pixels with small $\lambda_{\min}$.
4) Discard pixels with large $\lambda_{\max} - \lambda_{\min}$.
5) The remaining pixels are corner points.

However, a more efficient approach is presented by using a corner responses function which is defined as:

$$R = \det(A) - k\ (\text{trace(A)})^2 \tag{3.8}$$

where $k \approx 0.04$. This function avoids the explicit eigenvalue decomposition of *A*, and therefore is computationally faster. The output *R* is a constant for each pixel, and pixels with large *R* are picked as corners. Fig. 3.10(c) shows the result of an image processed by the corner responses function. The corner points are detected and marked with red "+" symbols.

3.2.2 Preprocess and Anisotropic Diffusion

The decomposition of an image is based on the cost value of its two decomposed layers. However, in real images, the textures are more complicated and sometimes there may be noise existing. Consequently, when we detect edges by edge magnitude and corners with a Harris detector, it may give responses in many seemingly flat regions of the image. With these wrong responses, the output of the cost function likely has a deviation, hence the decomposition may fail. To solve this problem, we make preprocess before apply cost function to calculate the cost value of the two decomposed layers. We first apply a nonlinear smoothing respectively to each layer and then apply Eq. (3.3) to the smoothed layers. The reason for this preprocess is that we want our edge and corner operators to return zero in regions that are nearly uniform, and hence the deviation of cost value will be decreased. Here we use a method named as anisotropic diffusion [35] to do the smoothing.

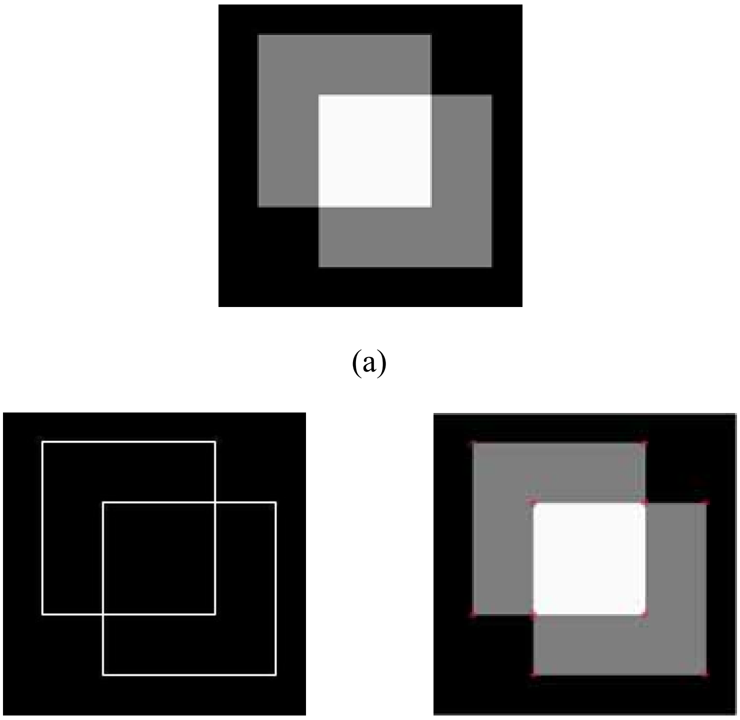
One of the most prevalent uses of image processing is for smoothing or denoising images. Denoising of images is often done with a low-pass filter, which reduces noise, but also blurs sharp features and details, such as edges. Anisotropic diffusion is a method for smoothing complex, noisy surfaces, while preserving sharp, geometric features. An anisotropic diffusion equation is defined as

$$I_t = div\left(c\left(x,y,t\right)\nabla I\right) = c\left(x,y,t\right)\Delta I + \nabla c \cdot \nabla I \tag{3.9}$$

where div represents the divergence operator, ∇ and Δ respectively represents the gradient and Laplacian operator with respect to the space variables, and $c\left(x,y,t\right)$ is conduction coefficient at different locations and times. It reduces to the isotropic heat diffusion equation $I_t = c\Delta I$ if $c\left(x,y,t\right)$ is a constant. A diffusion in which the conduction coefficient is chosen locally as a function of the magnitude of the gradient of the brightness function, i.e.,

$$c\left(x,y,t\right) = g\left(\left\|\nabla I\left(x,y,t\right)\right\|\right) \tag{3.10}$$

will not only preserve, but also sharpen the brightness edges if the function $g\left(\bullet\right)$ is chosen properly.



(a)

(b)

(c)

Fig. 3.10. Image processed by edge detector and corner detector. (a) Input image. (b) Edge detected result. (c) Harris corner detected result.

For implementation, Eq. (3.9) can be discretized on a square lattice, with brightness values associated to the vertices, and conduction coefficients to the arcs such as shown in Fig. 3.11. A 4-nearest-neighbors discretization of the Laplacian operator can be used:

$$I_{i,j}^{t+1} = I_{i,j}^{t} + \lambda\left[c_N \cdot \nabla_N I + c_S \cdot \nabla_S I + c_E \cdot \nabla_E I + c_W \cdot \nabla_W I\right]_{i,j}^{t} \tag{3.11}$$

where $0 \le \lambda \le 0.25$ for the numerical scheme to be stable, N, S, E, W are the subscripts respectively for North, South, East, and West. The superscript and subscripts on the square bracket are applied to all the terms it encloses, and the symbol ∇ here indicates nearest-neighbor differences defined as:

$$\nabla_N I_{i,j} \equiv I_{i-1,j} - I_{i,j}$$

$$\nabla_S I_{i,j} \equiv I_{i+1,j} - I_{i,j}$$

$$\nabla_E I_{i,j} \equiv I_{i,j+1} - I_{i,j}$$

$$\nabla_W I_{i,j} \equiv I_{i,j-1} - I_{i,j} \tag{3.12}$$

The conduction coefficients are updated at every iteration as a function of the brightness gradient (3.10). The value of the gradient can be computed on different neighborhood structures achieving different compromises between accuracy and locality. The simplest choice is approximating the norm of the gradient at each arc location with the absolute value of its projection along the direction of the arc:

$$c_{N_{i,j}}^t = g\left(\left|\nabla_N I_{i,j}^t\right|\right)$$

$$c_{S_{i,j}}^t = g\left(\left|\nabla_S I_{i,j}^t\right|\right)$$

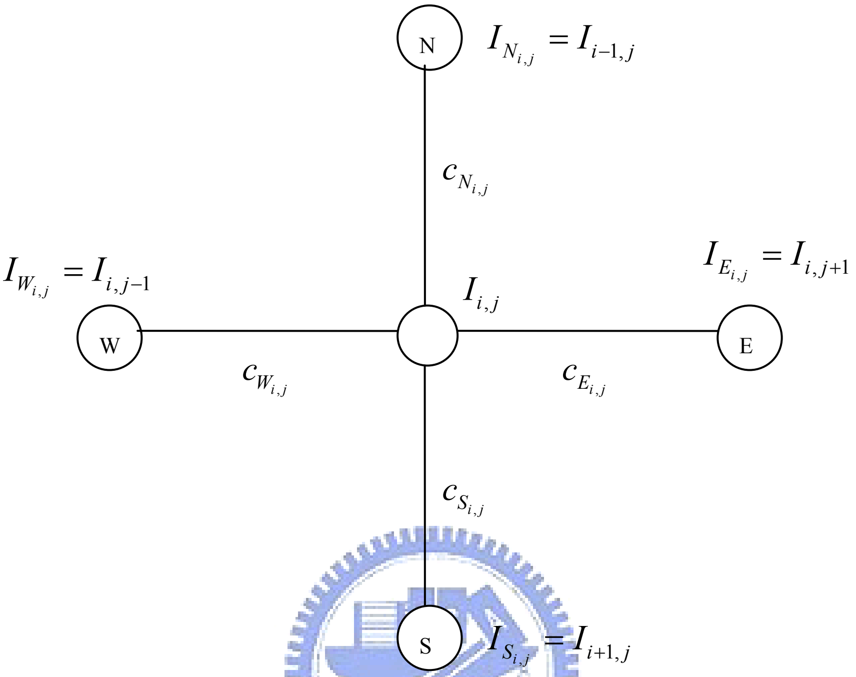
$$c_{E_{i,j}}^t = g\left(\left|\nabla_E I_{i,j}^t\right|\right)$$

$$c_{W_{i,j}}^t = g\left(\left|\nabla_W I_{i,j}^t\right|\right) \tag{3.13}$$

where t is the number of iteration. Functions used for $g(\bullet)$ are not specific but chosen by the users. Nevertheless, different functions which are chosen properly sometimes give perceptually similar results. Here we use the function below:

$$g\left(\nabla I\right) = e^{-(\|\nabla I\|/K)^2} \tag{3.14}$$

Fig. 3.12 shows the result of an image processed by anisotropic diffusion compared with the linear smoothing result. Since anisotropic diffusion transforms the nearly uniform regions into uniform ones and reduces noise while preserving edges, it greatly reduces the number of spurious "edges" and "corners" found by the gradient and Harris operators.



N
$I_{N_{i,j}} = I_{i-1,j}$
$c_{N_{i,j}}$
$I_{W_{i,j}} = I_{i,j-1}$
$I_{i,j}$
$I_{E_{i,j}} = I_{i,j+1}$
W
E
$c_{W_{i,j}}$
$c_{E_{i,j}}$
$c_{S_{i,j}}$
S
$I_{S_{i,j}} = I_{i+1,j}$

Fig. 3.11. The structure of the discrete computational scheme for simulating the diffusion equation. The brightness values $I_{i,j}$ are associated with the nodes of a lattice, the conduction coefficients c to the arcs.

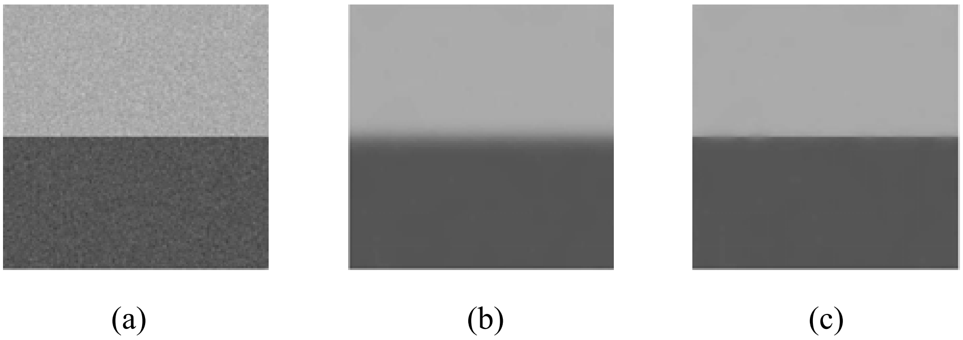


Fig. 3.12. Comparison between linear smoothing and anisotropic diffusion. (a) Original image. (b) Linear smoothing result. (c) Anisotropic diffusion result.

3.3 Discretization Using A Natural Images Database

Since there will be huge number of decompositions for an image, we have to find the minimum cost among all the possible decompositions. This problem is extremely difficult because of the huge space for the decompositions. We first consider a one dimensional subspace, a one dimensional family of decompositions for an image and their cost values will be calculated. The results are presented in Chapter 4. Subsequently, we attempt to find out more decompositions than one dimension. We use an approach whereby the problem is first discretized using a database of natural images. Instead of optimizing over the infinite space of possible decompositions, we discretize the problem by dividing the image into small 7×7 patches, and then search the suitable decomposition for each patch. The notion of using a patch representation was motivated by the success of this approach in a number of recent vision applications [36], [37] and some feature matching methods.

For each patch, its decomposition is obtained by searching a database of natural images for pairs of patches that approximately sum to the input patch, as some examples in Fig. 3.13. The database of patches are simply all patches contained in some images chosen for decomposing an image. The more features similar to the input image the database has, the more accurate the decomposition will be. For a patch *p*, we need to find a pair of patches (p_1, p_2) such that $p \approx p_1 + p_2$. For a database built with a 640×480 image, a naive way of performing this search for each patch is to search all about $10^5 \times 10^5$ possible pairs of patches, but this will be incredibly slow. To speed up the search, we make use of a filter bank which is a collection of directional filters at different locations, orientations and phases [38]. The oriented filter bank used in this work is based on rotated copies of a Gaussian derivative and its Hilbert transform. Let $f_1(x,y)=G''_{\sigma_1}(y)G_{\sigma_2}(x)$ and $f_2(x,y)$ is the Hilbert transform of $f_1(x,y)$ along the y axis, i.e.,

$$f_1(x,y)=\frac{d^2}{dy^2}\left(\frac{1}{C}\exp\left(\frac{y^2}{\sigma^2}\right)\exp\left(\frac{x^2}{\lambda^2\sigma^2}\right)\right) \tag{3.15}$$

$$f_2(x,y)=Hilbert\left(f_1(x,y)\right) \tag{3.16}$$

where σ is the scale, λ is the aspect ratio of the filter, and C is a normalization constant. Why is this useful from a computational point of view? The vector of filter outputs $I*f_i(x_0,\ y_0)$ characterizes the image patch centered at $(x_0,\ y_0)$ by a set of values at a point. This is similar to characterizing an analytic function by its derivatives at a point. The collection of response images $I*f_i$ is referred as the hypercolumn transform of the image. The hypercolumn transform provides a convenient method for contour and texture analysis and a good local descriptor of image patches.

To find patches $p\approx p_1+p_2$, we first search the database for the patches p_1 which minimize $\sum_i\left|f_i*p-f_i*p_1\right|$ where f_i is a collection of 12 directional filters which consist of two phases (even and odd), one scales, and six orientations in the upper left of the filter bank depicted in Fig. 3.14. After p_1 is found for a patch, p_2 could subsequently be chosen by performing a second search for $p-p_1$. The search now requires only $O\left(2\times10^5\right)$ operations rather than $O\left(10^5\times10^5\right)$ and hence saves a lot of time.

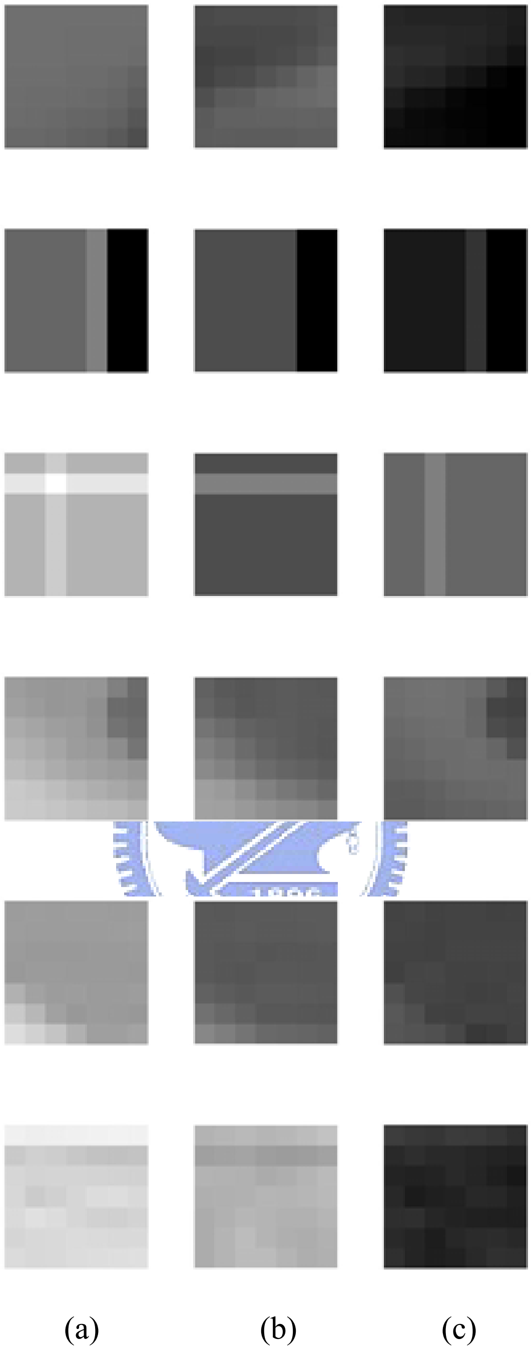


Fig. 3.13. Some examples for local patches decomposition. (a) Input patches. (b), (c) A possible pair of patches of decomposition for (a).

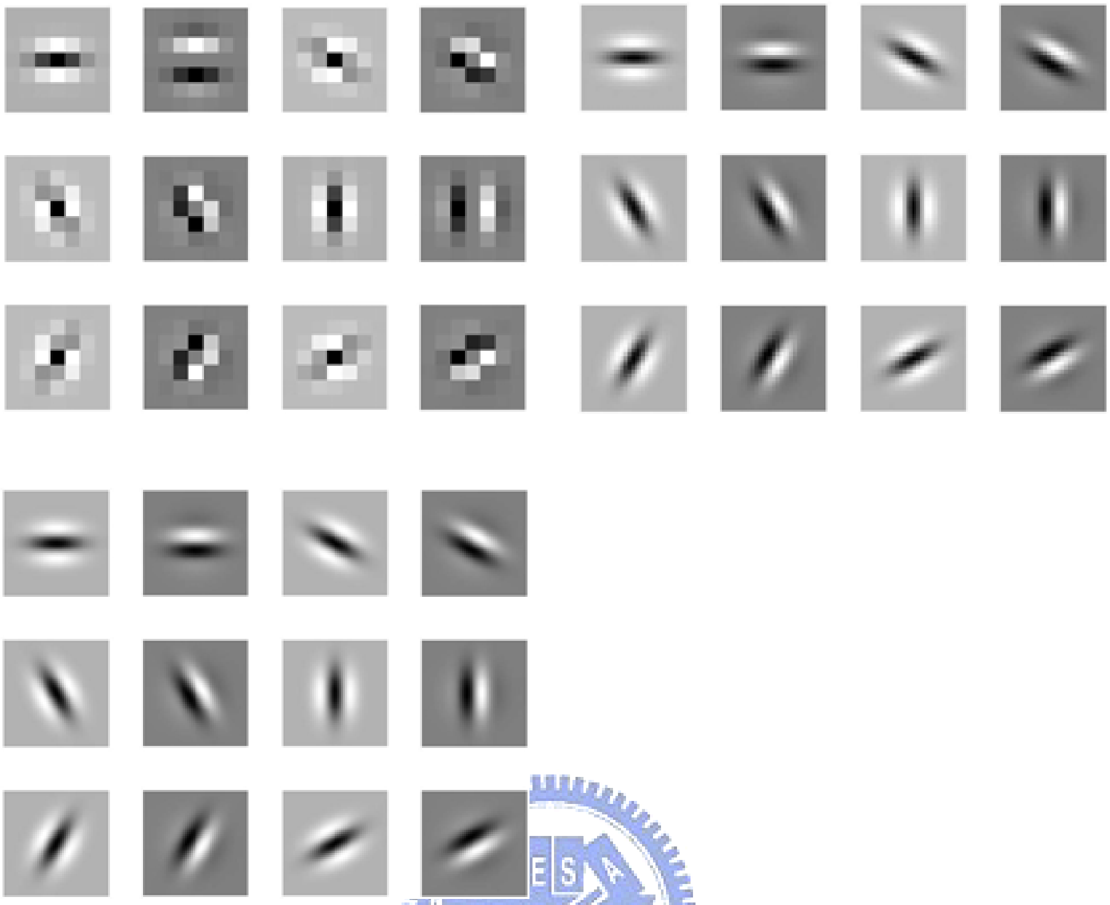


Fig. 3.14. A filter bank consisting of two phases (even and odd), three scales, and six orientations (equally spaced from 0 to π).

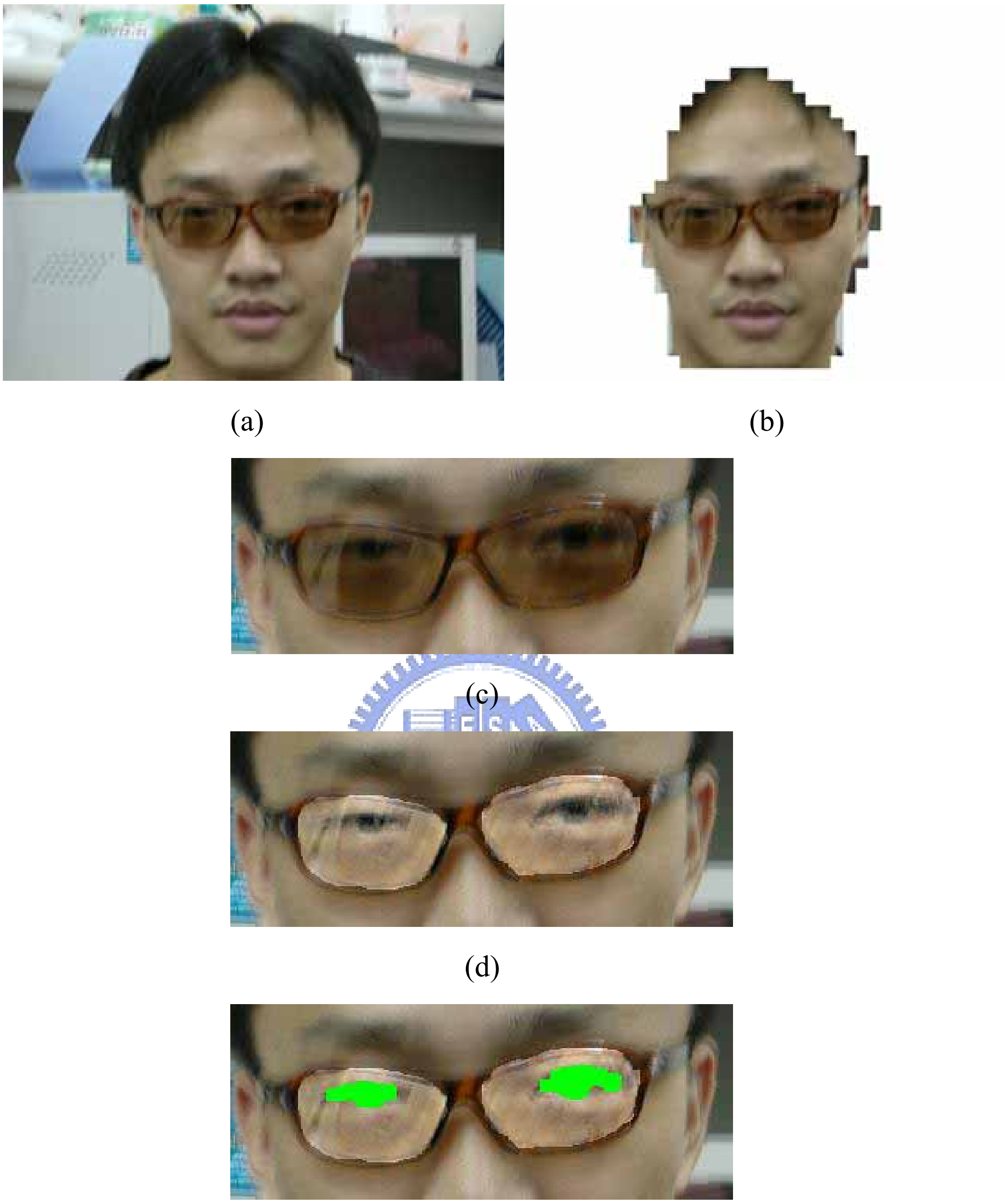
Chapter 4

Simulation and Results

We first show the results of face detection, MSRCR image enhancement technique, and eye detection algorithm. First, we capture frontal face images of people by a CCD camera, and then we will apply these algorithms to these images. In Sec. 4.1, the algorithm will be tested of people wearing sunglasses. We will give the results step-by-step of these algorithms. In Sec. 4.2, we will first show some results of one dimensional decompositions to verify the accuracy of the cost function for separation. Subsequently, results of decomposing some images with reflection into two layers by discretization, which makes use of patches and a collection of directional filters, will be shown.

4.1 Experiment Results of Eye Detection with Sunglasses

We take face images of two students in our laboratory to test the face detection and eye extraction with sunglasses images. The size of input images is 640×480 and the simulation is implemented on a Pentium IV 3.0GHz personal computer. These two examples are shown in Figs. 4.1–4.2. In each example, (a) is the original face image, and (b) is the face extraction result. Sub-image (c) is the region including the eyes and sunglasses areas, and (d) is enhanced image using MSRCR for the sunglasses region. Final output for eye extraction result is shown in (e). Note that generally the output after MSRCR processing will be out of the display domain. It needs to be shifted and compressed to the display domain.



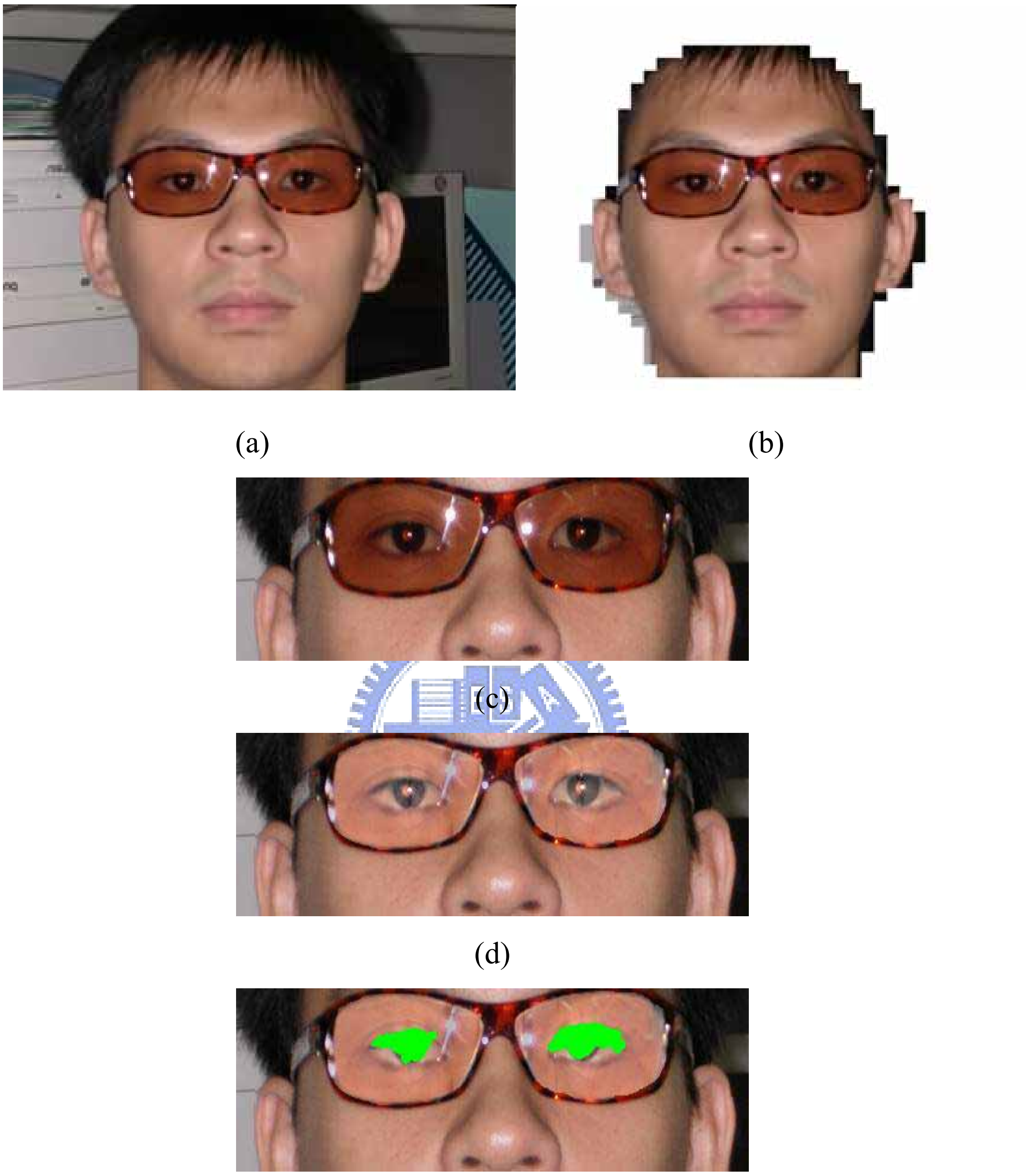
(a) (b)

(c)

(d)

(e)

Fig. 4.1. Images of Example 1 for face detection and eye location. (a) Input image. (b) Face extraction result. (c) Region including eyes and sunglasses. (d) The sunglasses image enhanced using MSRCR. (e) The eye extracted from (d).



(a) (b)

(c)

(d)

(e)

Fig. 4.2. Images of Example 2 for face detection and eye location. (a) Input image. (b) Face extraction result. (c) Region including eyes and sunglasses. (d) The sunglasses image enhanced using MSRCR. (e) The eye extracted from (d).

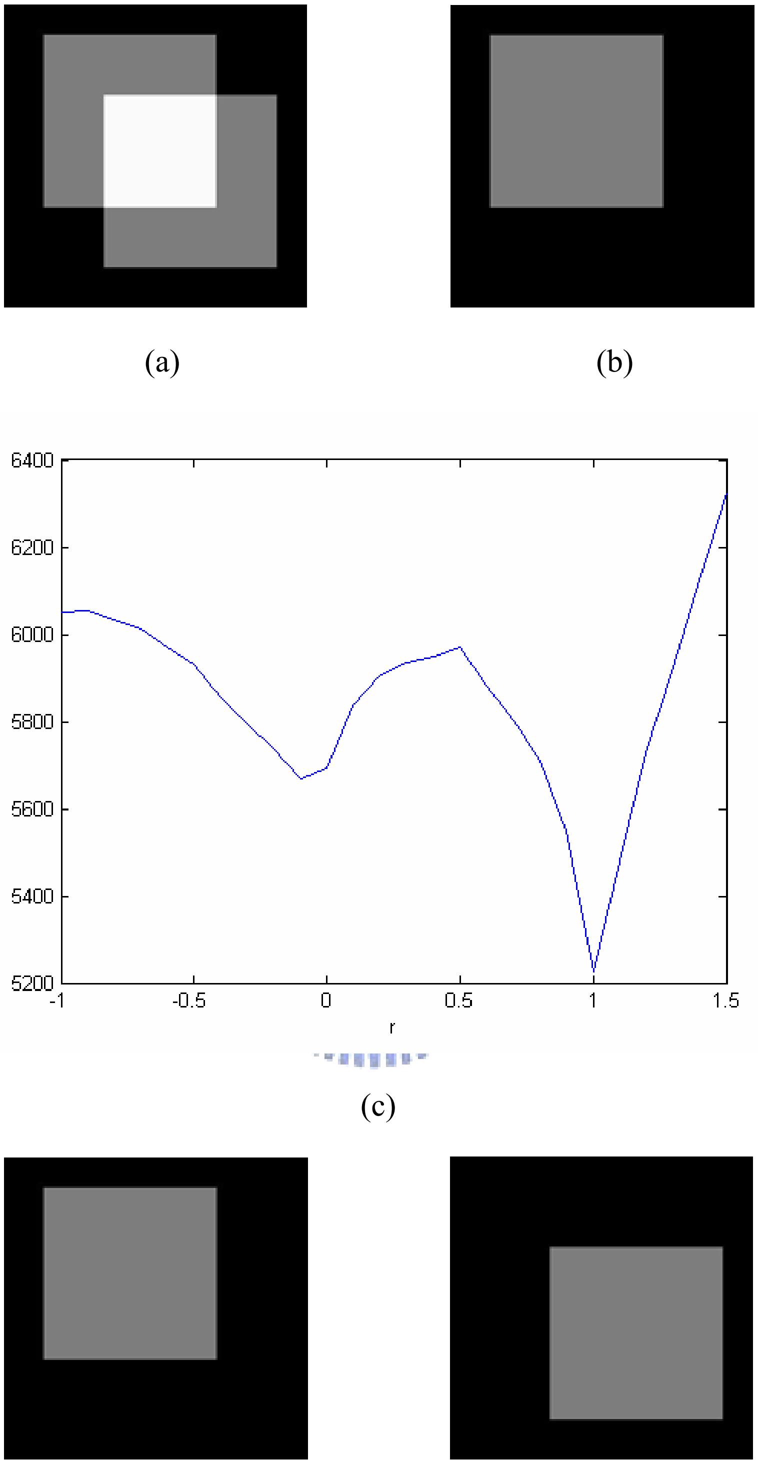
4.2 Experiment Results of Reflection Separation

4.2.1 One Dimensional Reflection Separation

In this section, we first consider a one dimensional family of solutions for an input image with reflection. Figs. 4.3–4.5 show the results of one dimensional function of three examples. For each example, (a) is the input image for test, and we defined the image of a single layer as $s(x,y)$ in (b). We considered decompositions of the form $I_1 = r \cdot s(x,y)$, $I_2 = I - I_1$ with respect to different values of *r* and then evaluated the cost for each decomposition as shown in (c). The x-axis is the values of *r*, and the y-axis is the corresponding cost. The decomposition result for the *r* with a minimum cost is shown in (d). Observing these results, we can find that indeed the minimum in this one dimensional subspace of solution is obtained at the perceptually correct decomposition.

4.2.2 Reflection Separation by Discretization

Subsequently, we applied five examples to test the decompositions by using discretization. We controlled the contrast of the reflection and made it vary with different tints for the test. Figs. 4.6–4.10 showed these five experiment results which were processed by the discretization method described in Sec. 3.3. To illustrate by using Example 1 of Fig. 4.6, for example, (a) is the input image consisting of the foreground image multiplied by 0.8 and the reflection image multiplied by 0.2; (d) is the input image consisting of the foreground image multiplied by 0.85 and the reflection image multiplied by 0.15, (b) and (c) are the separated layer images which include the foreground layer and the reflection layer of (a), respectively; (e) and (f) are the separated layer images which include the foreground layer and the reflection layer of (d), respectively. The reflection layer images were multiplied by a corresponding normalization scale to be seen clearly. Figs. 4.7–4.9 follow similar notion of Fig. 4.6. Fig. 4.10(a) is the input image consisting of the foreground image multiplied by 0.8 and the reflection image multiplied by 0.2, 4.10(b) and 4.10(c) are the separated layer images which include the foreground layer and the reflection layer, respectively. In these examples, it can approximately be seen that the decomposition results are more accurate for images with paler reflections than those with deeper reflections.



(a) (b)

(c)

(d)

Fig. 4.3. Example 1 for testing a one dimensional subspace of decompositions. (a) Original image *I*. (b) A single layer $s(x,y)$. (c) Diagram of *r* and corresponding cost. (d) The decomposition at the value of *r* with a minimum cost.

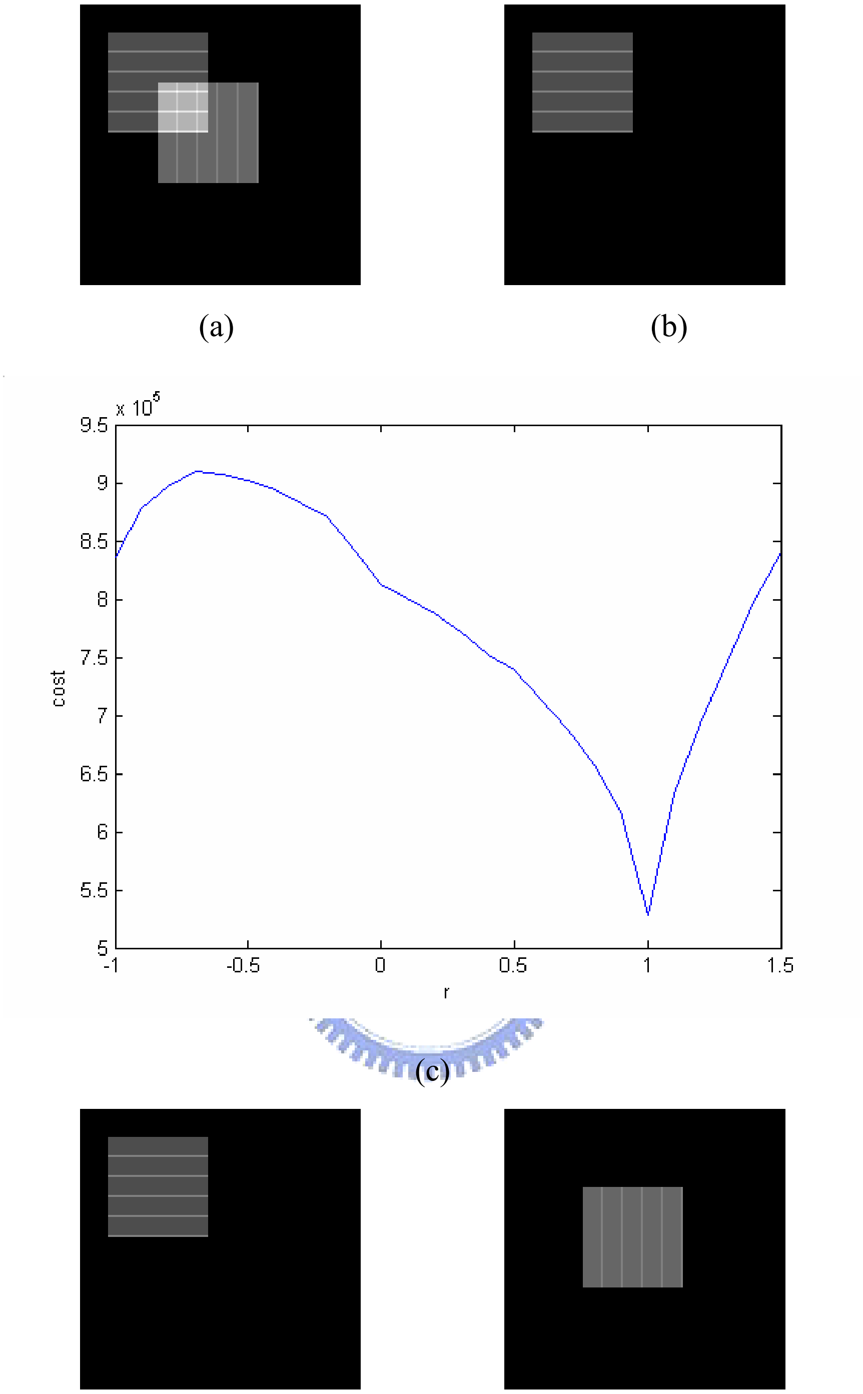
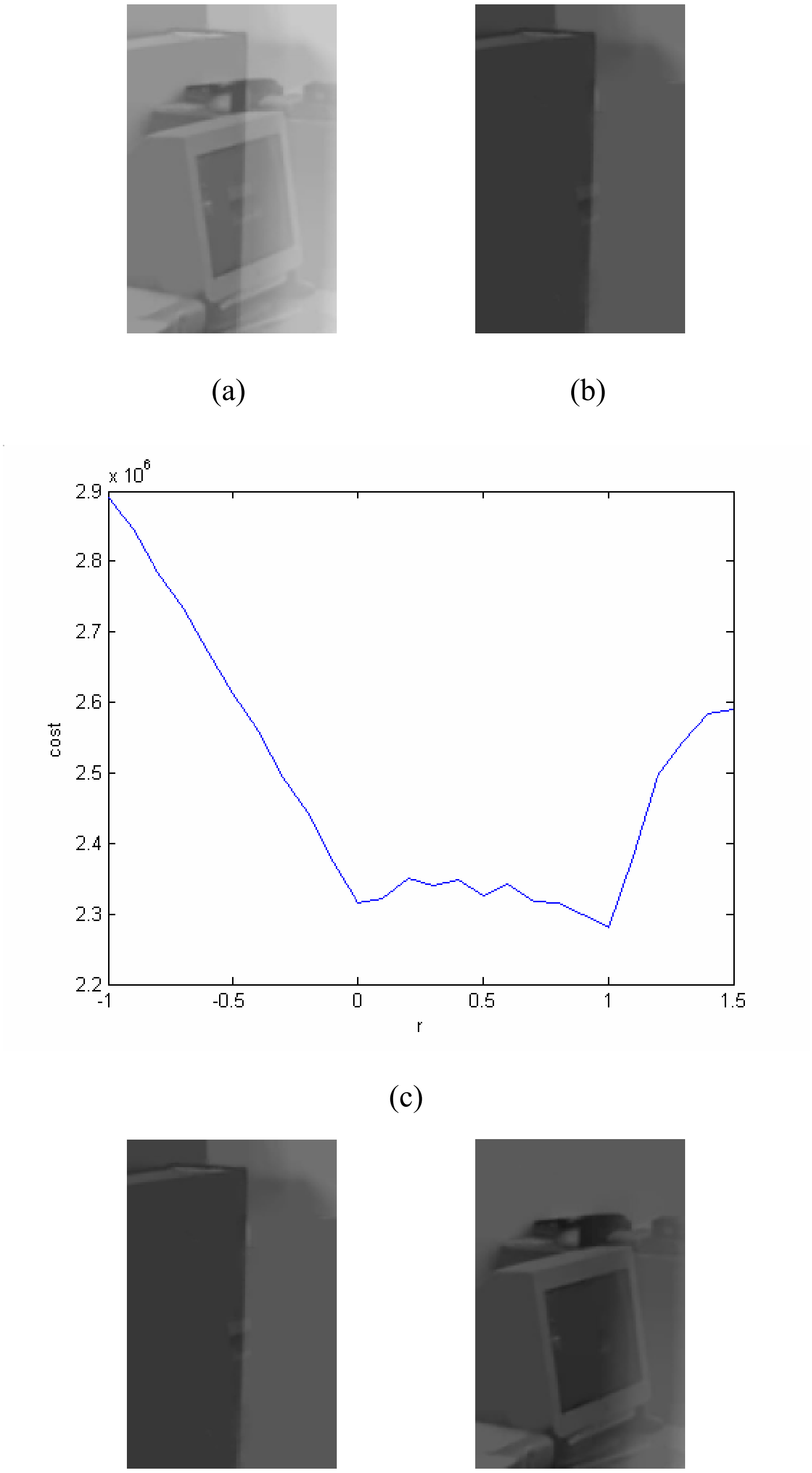


Fig. 4.4. Example 2 for testing a one dimensional subspace of decompositions. (a) Original image *I*. (b) A single layer $s(x,y)$. (c) Diagram of *r* and corresponding cost. (d) The decomposition at the value of *r* with a minimum cost.



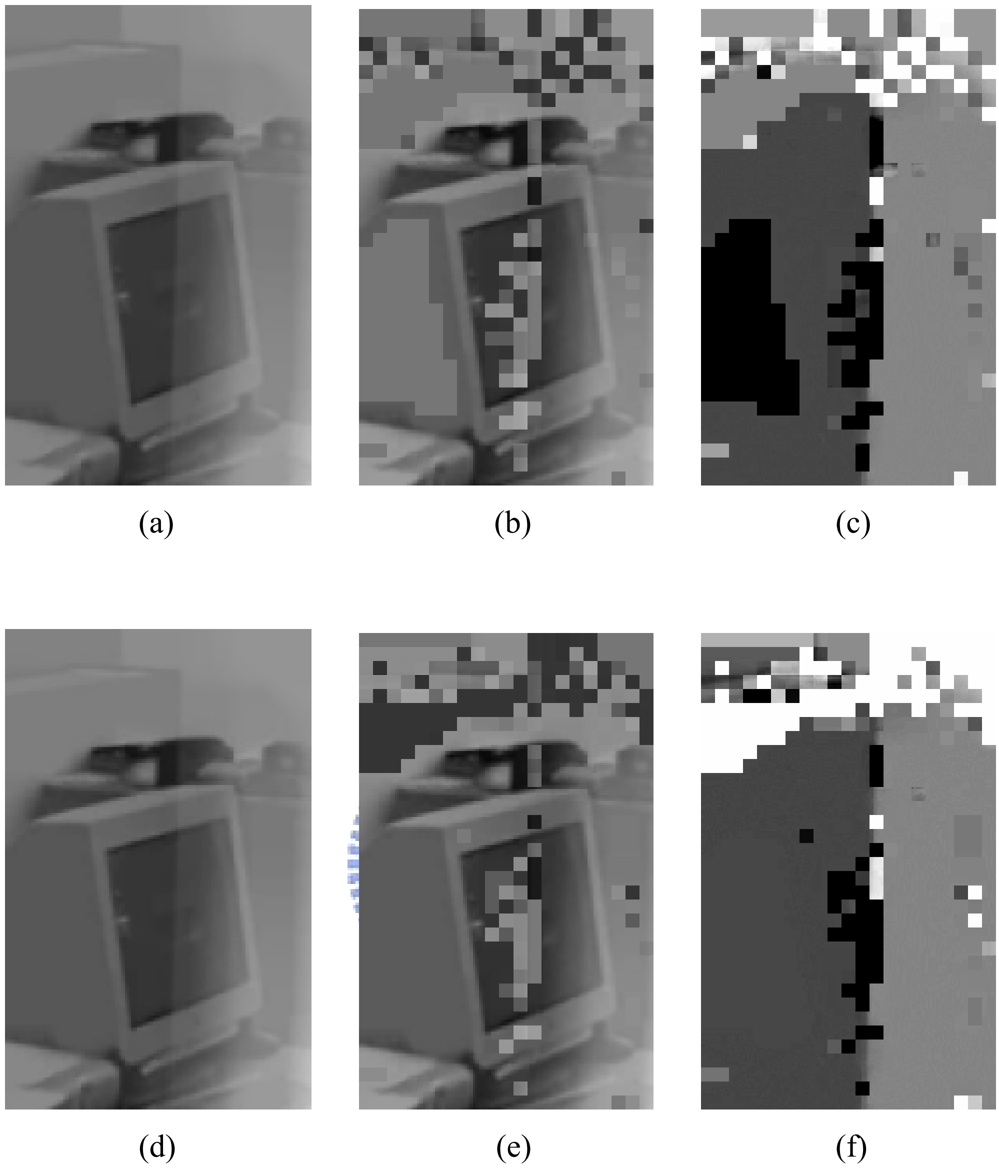
(a)

(b)

(c)

(d)

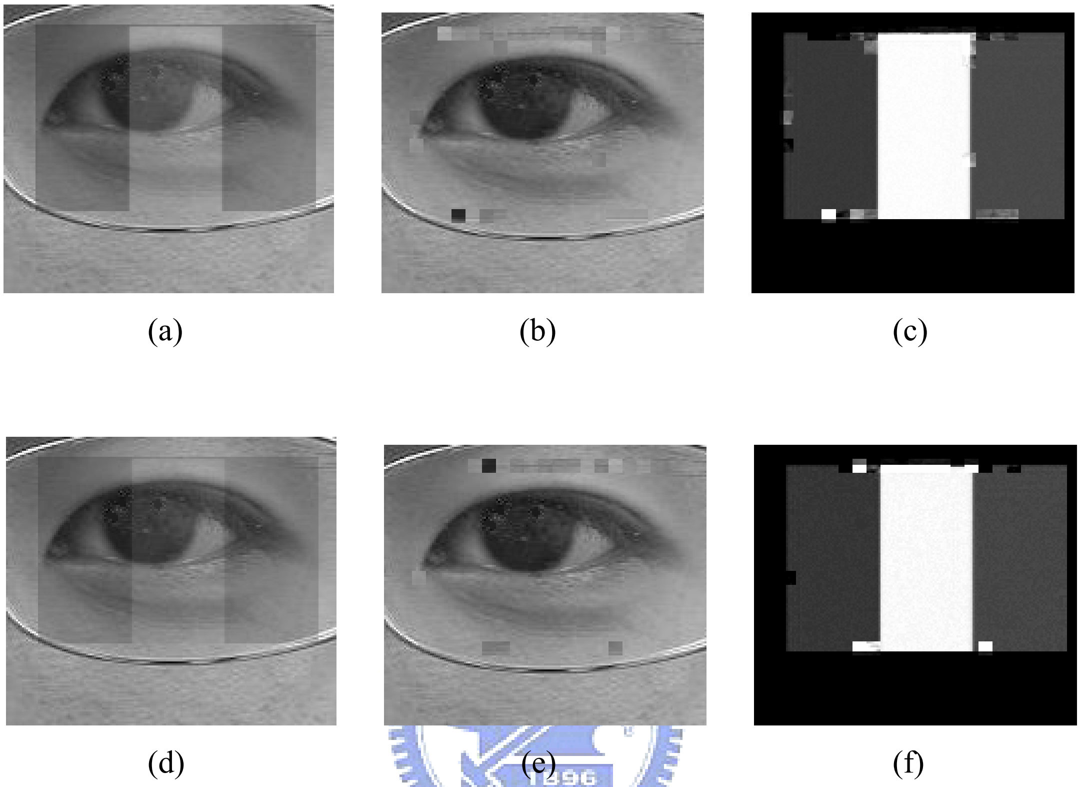
Fig. 4.5. Example 3 for testing a one dimensional subspace of decompositions. (a) Original image *I*. (b) A single layer $s(x,y)$. (c) Diagram of *r* and corresponding cost. (d) The decomposition at the value of *r* with a minimum cost.



(a) (b) (c)

(d) (e) (f)

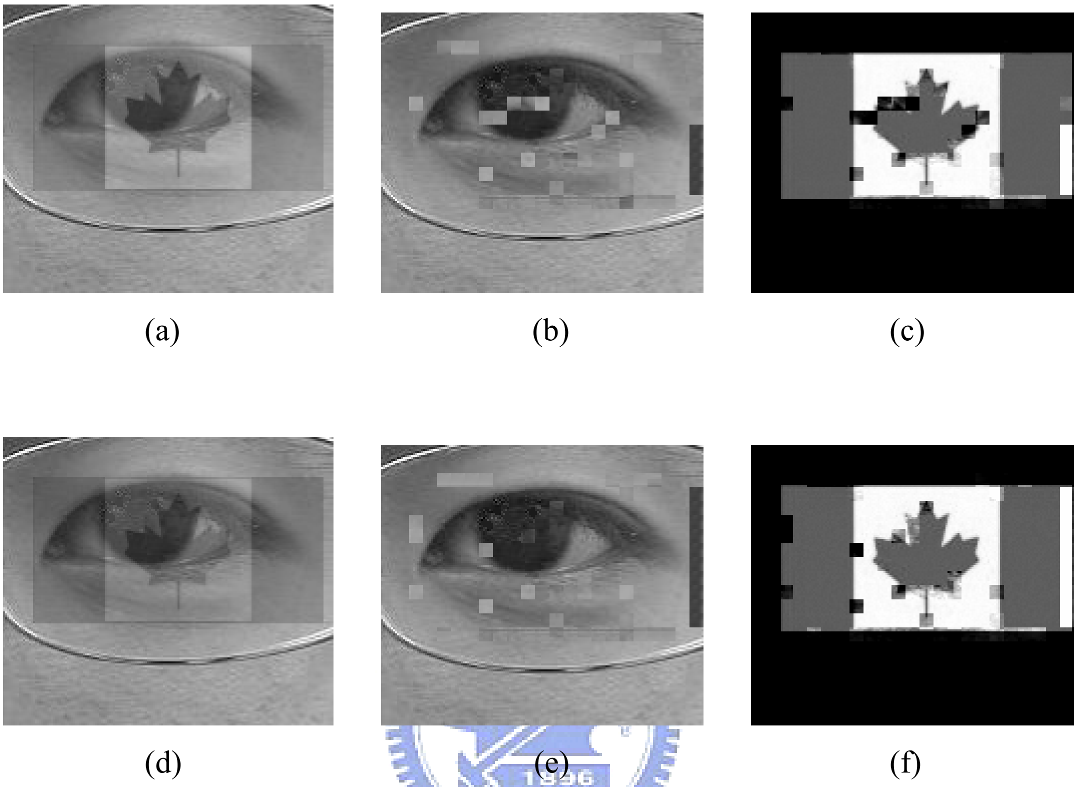
Fig. 4.6. Example 1 of separation results of images with reflections using discretization. (a) The input image consisting of the foreground image multiplied by 0.8 and the reflection image multiplied by 0.2. (b) Separated foreground layer image of (a). (c) Separated reflection layer image of (a). (d) The input image consisting of the foreground image multiplied by 0.85 and the reflection image multiplied by 0.15. (e) Separated foreground layer image of (d). (f) Separated reflection layer image of (d).



(a) (b) (c)

(d) (e) (f)

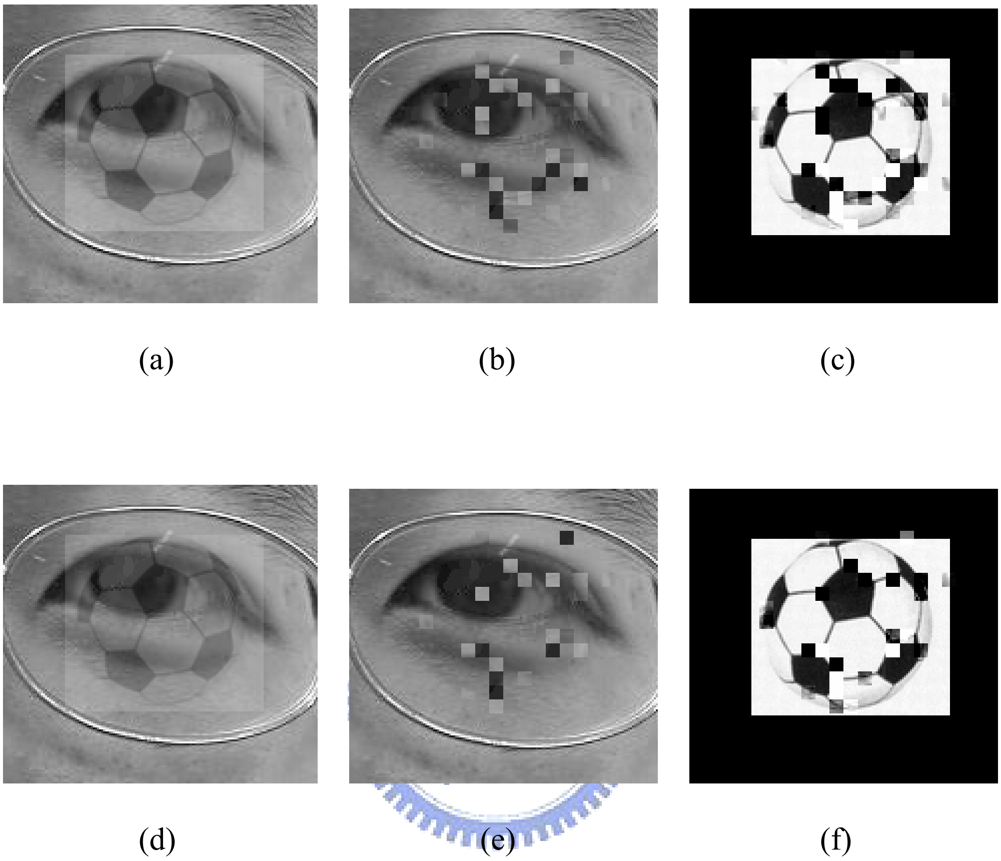
Fig. 4.7. Example 2 of separation results of images with reflections using discretization. (a) The input image consisting of the foreground image multiplied by 0.8 and the reflection image multiplied by 0.2. (b) Separated foreground layer image of (a). (c) Separated reflection layer image of (a). (d) The input image consisting of the foreground image multiplied by 0.85 and the reflection image multiplied by 0.15. (e) Separated foreground layer image of (d). (f) Separated reflection layer image of (d).



(a) (b) (c)

(d) (e) (f)

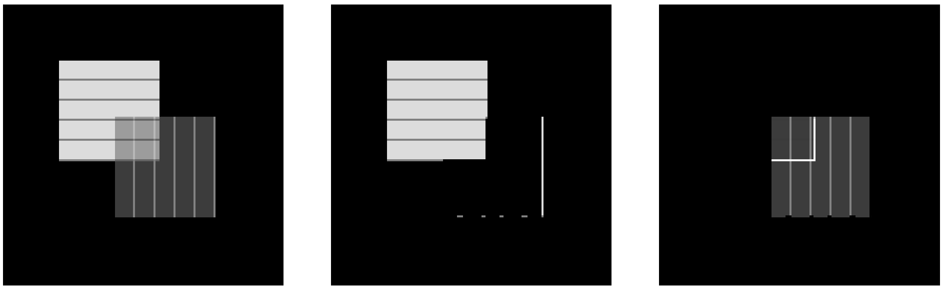
Fig. 4.8. Example 3 of separation results of images with reflections using discretization. (a) The input image consisting of the foreground image multiplied by 0.8 and the reflection image multiplied by 0.2. (b) Separated foreground layer image of (a). (c) Separated reflection layer image of (a). (d) The input image consisting of the foreground image multiplied by 0.85 and the reflection image multiplied by 0.15. (e) Separated foreground layer image of (d). (f) Separated reflection layer image of (d).



(a) (b) (c)

(d) (e) (f)

Fig. 4.9. Example 4 of separation results of images with reflections using discretization. (a) The input image consisting of the foreground image multiplied by 0.8 and the reflection image multiplied by 0.2. (b) Separated foreground layer image of (a). (c) Separated reflection layer image of (a). (d) The input image consisting of the foreground image multiplied by 0.85 and the reflection image multiplied by 0.15. (e) Separated foreground layer image of (d). (f) Separated reflection layer image of (d).



(a) (b) (c)

Fig. 4.10. Example 5 of separation result of a simple image with reflection using discretization. (a) The input image consisting of the foreground image multiplied by 0.8 and the reflection image multiplied by 0.2. (b) Separated foreground layer image. (c) Separated reflection layer image.

Chapter 5

Conclusion

This thesis first presents a method for eye detection, which is also applicable to wearing glasses or sunglasses cases. The eye detection module consists of three stages, face detection, sunglasses region enhancement, and eyeball extraction. These tasks are achieved by image processing techniques such as image segmentation, image enhancement, and edge detection. Experimental results showed that the color space chosen in this thesis is quite effective in face region detection, and the image enhancement technique, MSRCR, deals with the sunglasses cases well. Secondly a method for separating reflections was introduced and adopted to attempt to overcome the interference caused by reflections of glasses or sunglasses. The method find the perceptually correct separation by searching the decomposition with the smallest cost value, i.e., with the least number of edges and corners. It works effectively for a one dimensional separation. A discretization implemented by making use of patches can achieve more possible decompositions for an image with reflection, and then a filter bank is helpful to find the more correct decomposition. For paler reflections, this method is somewhat sufficient.

Our current approach for separating reflections is only a first step towards separating arbitrary reflection images from a single image. This algorithm often fails to separate reflections correctly for real images when image textures are more complex. One reason is that the suitable patches for the decomposition of a patch may not be picked by the algorithm, or perhaps they do not exist. For future work, we will try to consider the deeper reflections and more complex images cases. Some modifications for discretization and patches search are required so that the failure in dealing with complex reflections and images may decrease. The contrast of images needs to be further considered as well. For discretization, a further consideration for patches with overlaps may be helpful. For patches search, we may search ten or more candidate decompositions for each patch, and then utilize some optimization techniques such as belief propagation, loopy belief propagation, and graph cuts to find the optimal decomposition.

References

[1] W. W. Wierwille, S. S. Wreggit, C. L. Kim, L. A. Ellsworth, and R. J. Fairbanks, "Research on vehicle-based driver status/performance monitoring: development, validation, and refinement of algorithms for detection of driver drowsiness," *Tech Report,* (No. DOT HS 808 638), Washington, D. C: National Highway Traffic Safety Administration, Dec. 1994.

[2] C. D. Wylie, J. C. Shultz, M. M. Miller, and R. R. Mackie, "Commercial motor vehicle driver fatigue and alertness study," *Project Report* (Report No. FHWAMC-97-002), Washington, D. C: Federal Highway Administration Office of Motor Carroers, Oct. 1996.

[3] W. W. Weirwille, "Overview of research on driver drowsiness definition and driver drowsiness detection," in *Proc. 14th International Technical Conference on Enhanced Safety of Vehicles*, Munich, May 1994.

[4] K. Ogawa and M. Shimotani, "A drowsiness detection system," *Mitsubishi Tech. Reports*, vol. 78, pp. 13–16, 1997.

[5] E. Hjelmas and B. K. Low, "Face detection: A survey," *Computer Vision and Image Understanding*, vol. 83, pp. 236–274, 2001.

[6] D. Chai and K. N. Ngan, "Face segmentation using skin-color map in videophone applications," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 9, pp. 551–564, 1999.

[7] R. L. Hsu, M. A. Mottaleb, and A. K. Jain, "Face detection in color images," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 24, pp. 696–706, 2002.

[8] H. Wu, Q. Chen, and M. Yachida, "Face detection from color images using a fuzzy pattern matching method," *IEEE Trans. Pattern Anal. Machine Intell.*, vol.

21, pp. 557–563, 1999.

[9] J. Yang and A. Waibel, “A real-time face tracker,” in *Proc. 3rd IEEE Workshop on Application of Computer Vision*, 1996, pp. 142–147.

[10] J. Y. Chang and J. L. Chen, “Automated facial expression recognition system using neural networks,” *J. of the Chinese Institute of Engineers*, vol. 24, no. 3, pp. 345–356, 2001.

[11] I. Cohen, N. Sebe, A. Garg, M. S. Lew, and T. S. Huang, “Facial expression recognition from video sequences,” in *Proc. IEEE Multimedia and Expo Conf.*, vol. 2, Aug. 2002.

[12] M. Kirby and L. Sirovich, “Application of the Karhunen-Loeve procedure for the characterization of human faces,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 12, no. 1, pp. 103–108, 1990.

[13] L. B. Wolff and T. Boult, “Constraining Object Features Using Polarization Reflectance Model,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 13, no. 7, pp. 635–657, July 1991.

[14] S. K. Nayar, X. S. Fang, and T. Boult, “Separation of Reflection Components Using Color and Polarization,” *Int’l J. Computer Vision*, vol. 21, no. 3, 1996.

[15] H. Farid and E. H. Adelson, “Separating reflections from images by use of independent components analysis,” *Journal of the optical society of america*, 16(9):2136–2145, 1999.

[16] Y. Shechner, J. Shamir, and N. Kiryati, “Polarization-based decorrelation of transparent layers: The inclination angle of an invisible surface,” in *Proc. ICCV*, pp. 814–819, 1999.

[17] R. Szeliksi, S. Avidan, and P. Anandan, “Layer extraction from multiple images containing reflections and transparency,” in *Proc. IEEE CVPR*, 2000.

[18] M. Irani and S. Peleg, “Image sequence enhancement using multiple motions

analysis," in *Proc. IEEE Conf. Comput. Vision Pattern Recog.*, pp. 216–221, Champaign, Illinois, June 1992.

[19] Y. Tsin, S. B. Kang, and R. Szeliski, "Stereo matching with reflections and translucency," in *Proc. IEEE Conf. Comput. Vision Pattern Recog.*, pp. 702–709, 2003.

[20] S. W. Lee and R. Bajcsy, "Detection of Specularity Using Color and Multiple Views," *Image and Vision Computing*, vol. 10, pp. 643–653, 1990.

[21] S. Lin, Y. Li, S. B. Kang, X. Tong, and H. Y. Shum, "Diffuse-Specular Separation and Depth Recovery from Image Sequences," in *Proc. European Conf. Computer Vision*, pp. 210–224, 2002.

[22] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *Proc. of the Sixth International Conference on Computer Vision*, Bombay, India, January 1998.

[23] A. Levin and Y. Weiss, "User Assisted Separation of Reflections from a Single Image Using a Sparsity Prior," in *Proc. of the European Conference on Computer Vision (ECCV)*, Prague, May 2004.

[24] A. Levin, A. Zomet, and Y. Weiss, "Separating Reflections from a Single Image Using Local Features," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, June 2004, Washington DC.

[25] S. Kawato and J. Ohya, " Two-step approach for real-time eye tracking with a new filtering technique," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, 2000, vol. 2, pp. 1366–1371.

[26] E. Land, "An alternative technique for the computation of the designator in the retinex theory of color vision", in *Proc. Nat. Acad, Sci.*, vol.83, pp. 3078–3080, 1986.

[27] Z. Rahman, G. A. Woodell, and D. J. Jobson, “A Comparison of the Multiscale Retinex With Other Image Enhancement Techniques,” in *Proc. of the IS&T 50th Annual Conference*, 1997.

[28] D. J. Jobson, Z. Rahman, and G. A. Woodell, “Properties and performance of a center/surround retinex,” *IEEE Trans. Image Processing*, vol. 6, pp. 451–462, Mar. 1997.

[29] D. J. Jobson, Z. Rahman, and G. A. Woodell, "A multi-scale retinex for bridging the gap between color images and the human observation of scenes," *IEEE Tran. Image Processing*, vol. 6, pp. 965-976, July 1997.

[30] D. J. Jobson, Z. Rahman, and G. A. Woodell, "Retinex processing for automatic image enhancement," in *Proc. IS&T/SPIE Electronic Imaging 2002. The Human Vision and Electronic Imaging VII Conference*, 2002, vol. 4662, pp. 390–401.

[31] B. A. Olshausen and D. J. Field, “Emergence of simple-cell receptive field properties by learning a sparse code for natural images,” *Nature*, 381:607–608, 1996.

[32] E. P. Simoncelli, “Statistical models for images: compression restoration and synthesis,” in *Proc. Asilomar Conference on Signals*, *Systems and Computers*, pp. 673–678, 1997.

[33] A. Levin, A. Zomet, and Y. Weiss, “Learning to perceive transparency from the statistics of natural scenes,” S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, 2002.

[34] C. Harris and M. Stephens, “A combined corner and edge detector,” in *Proc. 4th Alvey Vision Conference*, pp. 147–151, 1988.

[35] P. Perona and J. Malik, “Scale space and edge detection using anisotropic diffusion,” *IEEE Trans. on Pattern Analysis and Machine Intelligence,*

12(7):629–639, July 1990.

[36] W. T. Freeman and E. C. Pasztor, “Learning to estimate scenes from images,” in M. S. Kearns, S. A. Solla, and D. A. Cohn, editors, *Adv. Neural Information Processing Systems 11*. MIT Press, 1999.

[37] A. A. Efros and W. T. Freeman, “Image quilting for texture synthesis and transfer,” in *Proc. of SIGGRAPH*, pp. 341–346, August 2001.

[38] J. Malik, S. Belongie, T. Leung, and J. Shi, “Contour and texture analysis for image segmentation,” in K. L. Boyer and S. Sarkar, editors, *Perceptual Organization for artificial vision systems*. Kluwer Academic, 2000.