A SATURATION-BASED IMAGE FUSION TECHNIQUE FOR STATIC SCENES

BY

GELEY PELJOR

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF SCIENCE (ENGINEERING AND TECHNOLOGY)
SIRINDHORN INTERNATIONAL INSTITUTE OF TECHNOLOGY
THAMMASAT UNIVERSITY
ACADEMIC YEAR 2015

A SATURATION-BASED IMAGE FUSION TECHNIQUE FOR STATIC SCENES

BY

GELEY PELJOR

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF SCIENCE (ENGINEERING AND TECHNOLOGY)
SIRINDHORN INTERNATIONAL INSTITUTE OF TECHNOLOGY
THAMMASAT UNIVERSITY
ACADEMIC YEAR 2015

A SATURATION-BASED IMAGE FUSION TECHNIQUE FOR STATIC SCENES

A Thesis Presented

By

GELEY PELJOR

Submitted to

Sirindhorn International Institute of Technology

Thammasat University

In partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE (ENGINEERING AND TECHNOLOGY)

Approved as to style and content by

Advisor and Chairperson of Thesis Committee Toshiaki Kondo

(Assoc. Prof. Dr. Toshiaki Kondo, Ph.D.)

Committee Member and
Chairperson of Examination Committee [illegible]

(Assoc. Prof. Dr. Waree Kongprawechnon, Ph.D.)

Committee Member [illegible]

(Dr. Teera Phatrapornnant, Ph.D.)

DECEMBER 2015

Acknowledgements

First of all, with due respect, my deepest thanks and sincere appreciation goes to my advisor Assoc. Prof. Dr. Toshiaki Kondo for his kind support throughout my master study, without which this work would not have been possible. I genuinely thank him for his patience and for always motivating me. I consider myself as a lucky advisee for having such a humble and kindhearted mentor like him. I can never thank him enough for his kind supervision.

Apart from my advisor, my extreme sincere gratitude and appreciation goes to my thesis committee members: Assoc. Prof. Dr. Waree kongprawechnon and Dr.Teera Phatrapornnant for their insightful comments and encouragement. It is a great honor to work under their guidance.

My heartfelt appreciation goes to SIIT, Thammasat University for providing us with the full thesis support and to National Research University Project, (NRU), Thailand Office of the Higher Education Commission for the financial support that funded parts of the research discussed in this dissertation and to all the staffs of SIIT for the overwhelming hospitality.

Lastly, special thanks to my family for always supporting me and believing in me.

Abstract

A SATURATION-BASED IMAGE FUSION TECHNIQUE FOR STATIC SCENES

by

GELEY PELJOR

B.Sc. IT, Bharathiar University, 2012

In this paper, we present a saturation based image fusion technique for static scenes, which is aimed at generating high dynamic range images. The proposed methods can automatically fuse a set of images of the same stationary scene with different exposure levels that may include overexposed and underexposed images. The input images are blended into one output image automatically as a linear combination of the input images. The weighting factors for blending the images are determined by the saturation values of them. Larger saturation values indicate more vivid colors. Thus, we assign heavier weights to the pixels with larger saturation values, while lighter weights are applied to the pixels with smaller saturation values. Compared with existing methods, the fused images by the proposed method are satisfactory both visually and numerically at a low computational cost. The output images are also fairly free from the halo effect that is a common problem in many image fusion techniques.

Keywords: Saturation, Image Fusion, Exposure Fusion, HDR, Tone mapping.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

With the advancement of communication technology and digital cameras, digital photographs are becoming hugely popular and common all over the world. A wide spread of mobile phones with a digital camera accelerates this trend in recent years. However, digital photography has a limited dynamic range, which comes from the limitation of the image sensors, e.g., CCD and CMOS. Hence, digital photography is often subject to overexposed or underexposed problems because it is difficult to record both bright and dark objects clearly in a single image.

The dynamic range of real-world scenes can be quite high, ratios of 100,000:1 are common in the natural world. An HDR (High Dynamic Range) image stores pixel values that span the whole tonal range of real-world scenes. Thus, High dynamic range (HDR) imaging is a digital photography technique used for creating a picture with a greater dynamic range. However, in our proposed method we follow a quasi–HDR with a traditional quantization levels, 8 bits for each primary color (R, G and B). Thus, our method does not need any special monitor to display the results. HDR images can be produced by blending a multiple images of different exposure levels. Well-exposed regions in the multiple images are merged together to produce a single image that clearly shows both bright and dark objects. This process is known as image fusion. Image fusion is the process of combining two or more images of the same scene into a single image, where the resultant fused image will generate a perfectly well exposed picture with best possible details than the input images. There are some new HDR capture devices being developed, but they are very expensive and still far from becoming commercialized [1-3].

The subsequent chapters of this paper are organized as follows. Chapter 1.1 explains the problem statement. Chapter 1.2 explains the main purpose of the work. Chapter 2 covers the literature review and Chapter 3 explains the proposed procedure of image fusion. Furthermore, Chapter 4 covers the experimental results and analysis. Chapter 5 covers the conclusion, followed by the references.

1.1 Problem Statement

In general, the common problem of generating the HDR images is due to a limited dynamic range of a display device. Thus, when a scene includes both shiny and shadowy objects, it is difficult to capture both clearly in one image.

Generally speaking, there are many published papers with various types of technique for generating the HDR images. However, each technique has their own drawbacks. For Debevec et al. method the camera response curve calibration is needed. Furthermore, the HDR image cannot display on LDR devices directly. To this end, the process known as tone mapping is applied to HDR image. Unfortunately, tone mapping will simply increase the computational time when displaying an image. Goshtasby method cannot deal properly with the boundaries of the object. Wei Zhang, et al., results look unnatural. On the other hand, Raman, et al., method shows unsatisfactory results on color visibility. Recently, a color image synthesization technique is proposed to obtain a complete image detail. Their method gives visually pleasing results. However, their approaches are complicated to implement in a real time applications.

1.2 Purpose of the Research

The main objective of this work is to propose a simple and fully automatic algorithm that provides a high dynamic range image at a low computational cost when used in real applications.

The other objective is to generate a fairly free from the halo effect problems.

Our method generates fully automatically without adjusting any parameters, unlike commercially available software such as, Photoshop.

By using an image fusion method, an overexposed region in one image can be replaced with that in another image and vice versa for the underexposed region.

Chapter 2
Literature Review

2.1 Background

High dynamic range (HDR) images can be acquired from the set of multiple exposure images [4]. Debevec et al. presented a method to recover the HDR radiance maps from the normal photographs [4]. Their method identifies the response curve of the imaging device, which is used to recover the HDR radiance map and then display it on LDR monitors with the help of tone mapping algorithms [5-7]. However, for these types of methods calibrating the camera response curve is needed. Furthermore, the HDR image cannot display on LDR devices directly. To this end, the process known as tone mapping is applied to HDR image. Unfortunately, tone mapping will simply increase the computational time when displaying an image.

Image fusion techniques have been used for many years. For example, multimodal imaging [8] and video enhancement [9]. Burt et al. have used pyramid transform to perform fusion of multi-exposure images [8].

As an alternative, exposure fusion was proposed. Exposure fusion is a technique for blending a series of bracketed exposures of the same scene into a single high-quality image without needing to go for the step of computing a HDR image and tone mapping algorithms, which can be displayed on LDR monitors [10-13]. Goshtasby has used the block based approach to fuse multi-exposure images[10]. Their method first divides the image into same blocks and the best exposure is selected for each block. Then the selected images are blended together using blending functions. Unfortunately, it cannot deal properly with the boundaries of the object. Wei Zhang, et al., [12, 13] used both static and dynamic scenes composed of multiple images with different exposures with the guidance of gradient- based quality assessment. The result looks unnatural as bright and dark effects of the input scene are disregarded. Moreover, there is no proper adjustment in color and sharpness. Exposure fusion method was proposed by Raman, et al., [13] using edge-preserving filters such as bilateral filters. The method shows unsatisfactory results on color visibility. Recently, a color image synthesization technique is proposed to obtain a complete image detail [14]. Their method gives visually pleasing results. However, their approaches are complicated to implement in real time applications.

Mertens *et al.* [11] introduced an exposure fusion method, which consists of two main parts:

1. Computing Weight Map from metrics like Saturation, Contrast and Well Exposedness.

2. Applying the Pyramidal Image Decomposition for multi-resolution blending and for fusing the images.

Malik et al. have extended the Merten et al. approach and used wavelet decomposition instead of pyramidal decomposition as wavelet offers more efficient and robust representation [15].

In the present work, our approach is similar to the Merten et al. approach of Computing Weight Map from metrics, but instead of computing weight map from three quality measures, we compute weight map only from the saturation channel followed by a Gaussian low-pass filter and some post processing techniques to the final fused image. We compare our result with the histogram equalization, which is used for contrast adjustment [16], the pixel-wise selection method, the gradient-based method and finally with the MATLAB HDR built-in function [17]. Our method does not require any information about camera shutting conditions. We assume that the images are strictly aligned and there is no moving object in the scenes. Our technique merely relies on saturation that is defined as,

$$s = 1 - \frac{\min(R,G,B)}{R+G+B} \qquad (1)$$

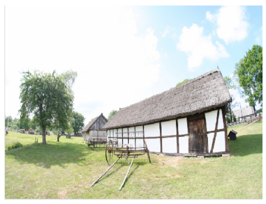
Which prove to be very effective. Our method generates fully automatically without adjusting any parameters, unlike commercially available software, such as, Photoshop.

Chapter 3

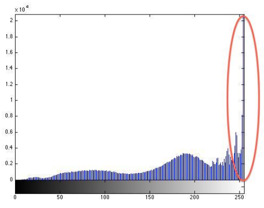
Methodology

3.1 Histogram Equalization

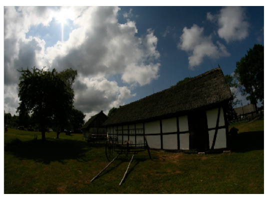
First of all, we apply histogram equalization (HE) to an input image as it is a well known automated technique for enhancing image contrast [16]. Figs. 3.1 and 3.2 show that HE is not always effective in improving image quality. In particular, Fig. 3.1(e) and Fig. 3.1(g) highlight the limitation of HE. HE fails to produce a high-quality image when its histogram has an extreme peak (thin and high) because it produces a discontinuity in its accumulated histogram (Fig. 3.1(b) and Fig. 3.2(d)).



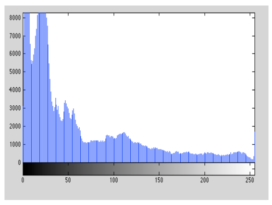
(a)



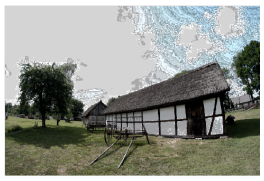
(b)



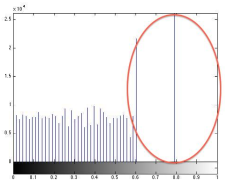
(c)



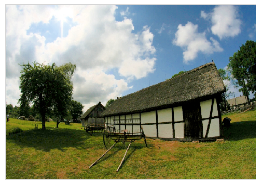
(d)



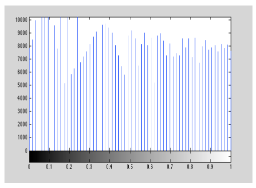
(e)



(f)

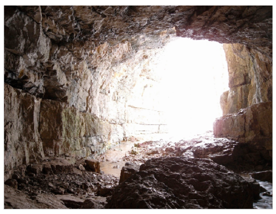


(g)

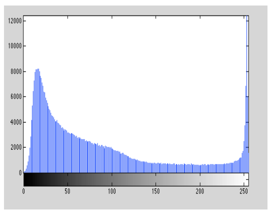


(h)

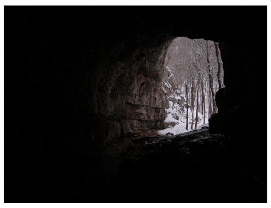
Fig. 3.1. Performance of histogram equalization. (a) Overexposed input image, (b) Histogram of the image in (a), (c) Underexposed input image, (d) Histogram of the image in (c), (e) Histogram equalized image of (a), (f) Histogram of (e), (g) Histogram equalized image of (c), and (h) Histogram of (g).



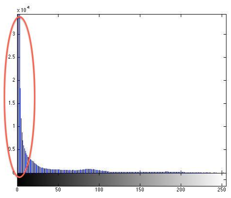
(a)



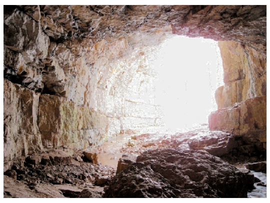
(b)



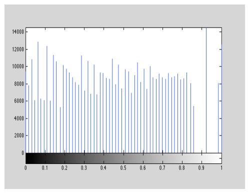
(c)



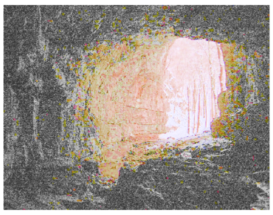
(d)



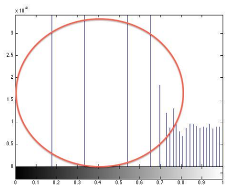
(e)



(f)



(g)



(h)

Fig. 3.2. Performance of histogram equalization. (a) Overexposed input image, (b) Histogram of the image in (a), (c) Underexposed input image, (d) Histogram of the image in (c), (e) Histogram equalized image of (a), (f) Histogram of (e), (g) Histogram equalized image of (c), and (h) Histogram of (g).

3.2 Gradient based image fusion

A popular image fusion method is the one based on image gradients. In general, an image region of greater gradients appear sharper than that of low gradients. Thus, input images can be fused based on their gradient magnitudes. The approach consists of two parts:
1. Obtain the image gradient in the *x* and *y* direction, g*x* and g*y*.
2. Compute the magnitude of the gradients as

$$m = \sqrt{gx^2 + gy^2} \qquad (2)$$

We use the Sobel operator, which comprises a pair of 3×3 convolution masks, *Gx* and *Gy*, one estimating the gradient in the *x*-direction (along columns) and the other estimating the gradient in the *y*-direction (along rows) as shown below:

-1	0	+1
-2	0	+2
-1	0	+1

Gx

+1	+2	+1
0	0	0
-1	-2	-1

Gy

Fig. 3.3 illustrates the methodology flow chart of a gradient-based method. Fig. 3. 3(a) shows the overexposed input image. Fig. 3.3(b) shows the underexposed input image. Fig. 3.3(c) shows the gray image of (a) and Fig. 3.3(d) shows the gray image of (b). Fig. 3.3(e) shows the gradient of (c) in the *x* direction and Fig. 3.3(f) shows the gradient of (c) in the *y* direction as

g*x*=G*x**GrayImage1 (3(c))
g*y*=G*y**GrayImage1(3(c))

Fig. 3.3(g) shows the gradient of (d) in the x direction and Fig. 3.3(h) shows the gradient of (d) in the y direction as

g*x*=G*x**GrayImage2 (3(d))
g*y*=G*y**GrayImage2 (3(d))

Fig. 3.3(i) shows the magnitude of the gradient combined from (e) and (f) using the Eq. (2). Fig. 3.3(j) shows the magnitude of the gradient combined from (g) and (h) using the Eq. (2). Fig. 3.3(k) shows the weight of (a) as

$$w_1(x,y) = \frac{m_1(x,y)}{m_1(x,y)+m_2(x,y)} \qquad (3)$$

Fig. 3.3(l) shows the weight of (b) as

$$w_2(x,y) = \frac{m_2(x,y)}{m_1(x,y)+m_2(x,y)} \qquad (4)$$

Finally, Fig. 3.3(m) shows the final fused image. As we see that the final fused image produces a vivid halo effect, which is normally appeared as a white glow around the subject, when the darker pixels meet the lighter pixels.

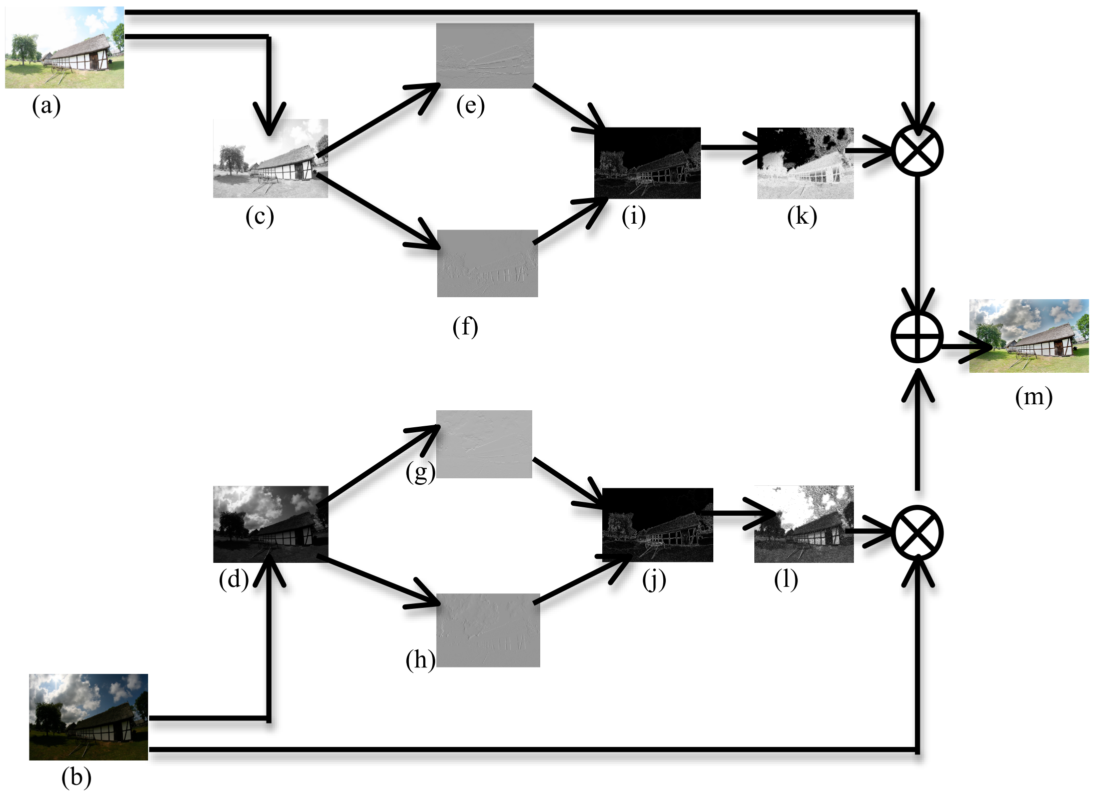


Fig. 3.3. Gradient based methodology flow chart. (a) Overexposed input image, (b) underexposed input image, (c) gray image of (a), (d), gray image of (b), (e) gradient of (c) in the x direction. (f) gradient of (c) in the y direction. (g) gradient of (d) in the x direction, (h) gradient of (d) in the y direction, (i) magnitude of the gradient combined from (e) and (f), (j) magnitude of the gradient combined from (g) and (h), (k) weight of (a), (l) weight for (b) and (m) final fused image.

3.3. MATLAB built-in function

The Image Processing Toolbox™ supports various set of image types including high dynamic range as well [17]. The toolbox includes MATLAB built-in functions, such as for reading, creating, and writing the HDR images into the MATLAB workspace. On top of that, it also includes a tone-map operator, which compresses the dynamic range of the image to fit on a display device. The syntax is given below:

HDR=makehdr(files) creates the HDR image from the set of registered low dynamic range images listed in the files .
HDR = makehdr(files, param1, val1,...) creates the HDR image from the low dynamic range images in files, defining parameters and values that control various aspects of the image creation. Hence, in chapter 4, we compare our results with the result of the MATLAB built-in function.

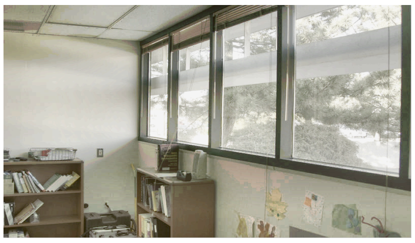


Fig. 3.4. MATLAB built-in function

3.4. Saturation based pixel-wise selection

In order to justify the need of a smooth blending in the proposed method, we use a saturation based pixel-wise selection method. In this approach, we simply select a pixel with higher saturation from the two input images and a pixel with lower saturation is discarded. In this way, two images are cropped and stitched together to be one output image.

Fig. 3.5 demonstrates the saturation based pixel-wise selection methodology flow chart. It is explicit to see that the final fused image (with post processing), produce a poor quality of image with unwanted patterns along the ceiling and the wall side regardless fusing the input RGB images with the pixels with higher saturation. Hence, a pixel-wise selection method fails to generate optimum results.

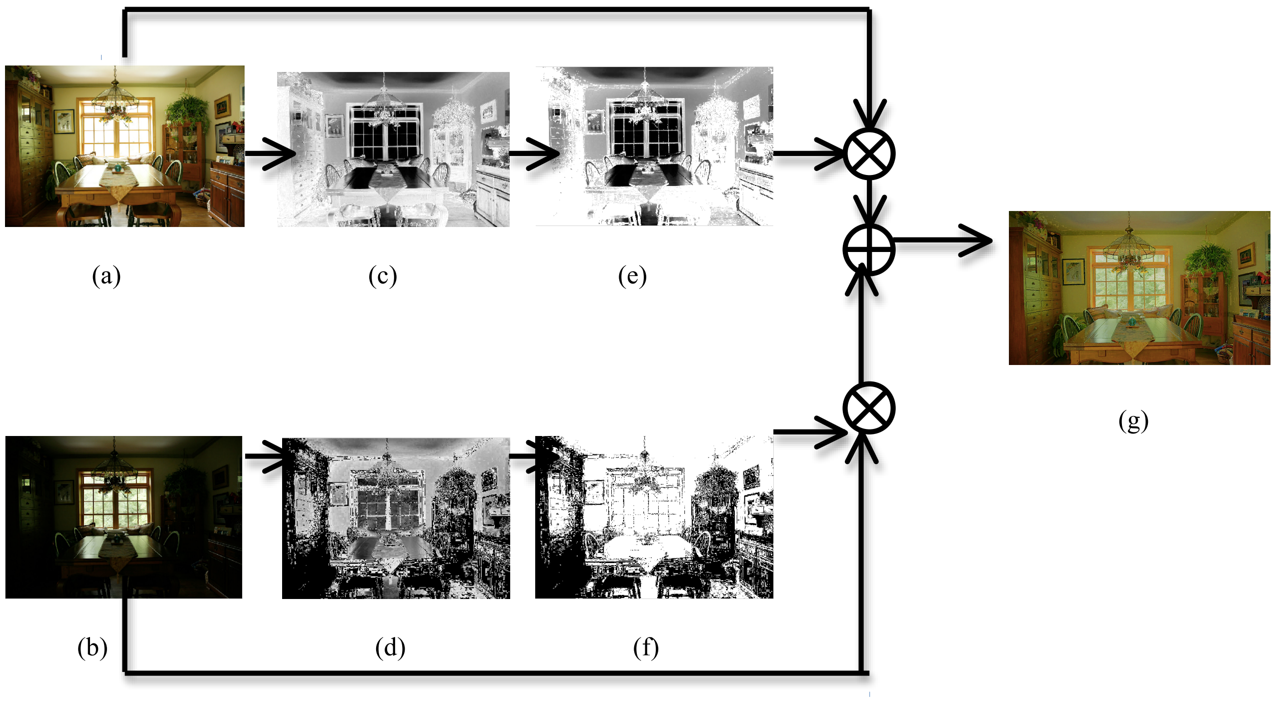


Fig. 3.5. Saturation based pixel-wise selection methodology flow chart. (a) Overexposed input image, (b) underexposed input image, (c) the saturation channel of the image (a), (d) the saturation channel of the image (b), (e) a pixel with higher saturation from (a), (f) a pixel with higher saturation from (b) and (g) final fused image with post processing.

3.5. Proposed Method

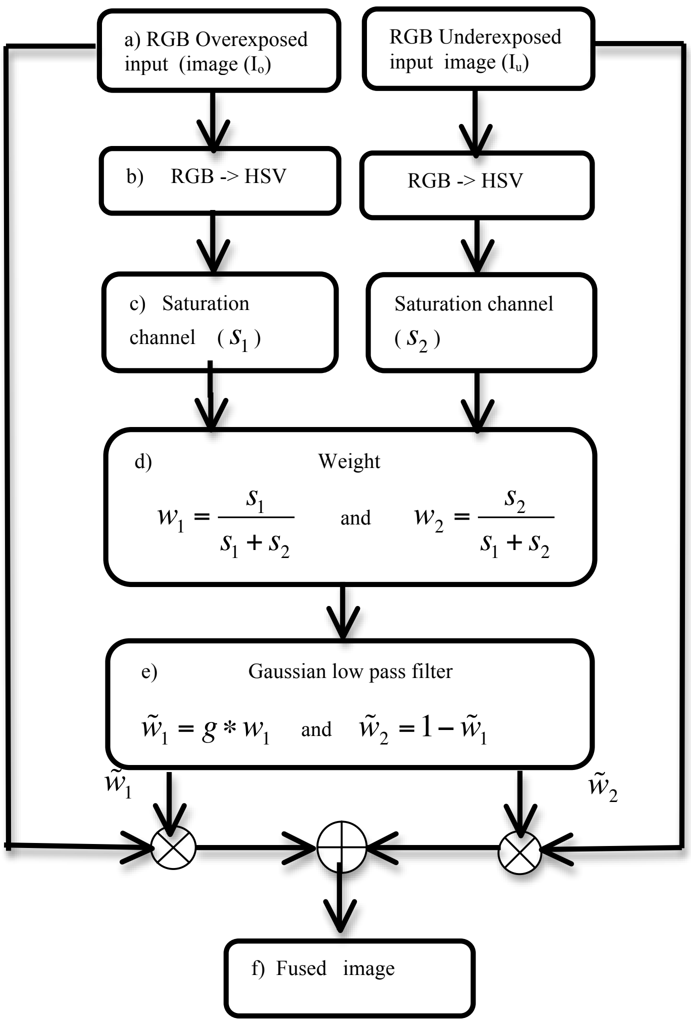
Fig. 3.6 shows the procedure of the proposed method. More details are given below:

1. We load a pair of static input images, I_o (overexposed) and I_u (underexposed) (Fig. 3.6 (a)).
2. Input images I_o and I_u are converted from the RGB to the HSV color space, respectively (Fig. 3.6 (b)).
3. Then, we extract the saturation channels from the HSV color space of the input images, respectively as s_1 (saturation of overexposed) and s_2 (saturation of underexposed) (Fig. 3.6 (c)).
4. Next, we take the sum of the two saturation images together as $s = s_1 + s_2$.
5. The pixels whose saturation is equal to 1 or close to 1 (Eq.1) are often underexposed and colorless. Hence, we replace the saturation value 1 or close to it with 0 to exclude those pixels from the image blending. Note that the pixels with saturation 0 are automatically disabled from the blending as well.

6. Then, we define the weighting factor on the overexposed image as $w_1 = \frac{s_1}{s_1 + s_2}$ and the one on the underexposed image as $w_2 = \frac{s_2}{s_1 + s_2}$ (Fig. 3.6 (d)). In the case of $s_1 + s_2 = 0$, we assign 0.5 to both w_1 and w_2.
7. We apply a Gaussian low-pass filter (LPF), g, to the weighting factor map (w_1) to smoothen it and stored as $\tilde{w}_1$. $\tilde{w}_2$ is obtained by $\tilde{w}_2 = 1 - \tilde{w}_1$ (Fig. 3.6 (e)). The size of the LPF is the one tenth of the vertical size of the input image.
8. We blend two input RGB full color images using the weighting factors $\tilde{w}_1$ and $\tilde{w}_2$.
9. We apply post processing to the fused image. First, we sharpen the final fused image using the unsharp masking technique, though this is optional. Then we convert the final fused sharpened image to the HSV color space. We then apply the gamma transform to the image intensity and saturation values where we currently use the fixed gamma value 0.7 for both. This step helps to brighten the output images.
Finally, the images are converted back to the RGB space (Fig. 3.6 (f)).

Weighted averaging= $\tilde{w}_1 * I_0 + \tilde{w}_2 * I_u$, where,

$\tilde{w}_1$ =Weighting factor
$\tilde{w}_2 = 1 - \tilde{w}_1$
I_0 =Overexposed input image
I_u =Underexposed input image



a) RGB Overexposed input (image (I_o)
RGB Underexposed input image (I_u)
b) RGB -> HSV
RGB -> HSV
c) Saturation channel (s_1)
Saturation channel (s_2)
d) Weight
$w_1 = \frac{s_1}{s_1 + s_2}$ and $w_2 = \frac{s_2}{s_1 + s_2}$
e) Gaussian low pass filter
$\tilde{w}_1 = g * w_1$ and $\tilde{w}_2 = 1 - \tilde{w}_1$
$\tilde{w}_1$
$\tilde{w}_2$
f) Fused image

Fig. 3.6. Proposed Method flow chart

Chapter 4
Results and Discussions

4.1. Experimental results

Figs.4.1-4.10 shows the comparison of our method with the histogram equalization techniques, the gradient-based method, the MATLAB HDR built-in function [17] and the saturation based pixel-wise selection method. We have obtained the sample images from the website [18] and [19]. Figs. 4.1(a)-4.10(a) shows the overexposed input image. Figs. 4.1(b)-4.10(b) shows the underexposed input image. Figs. 4.1(c)-4.10(c) shows the histogram equalized image of (a), as we can see the image produces lots of false contours in some parts of the region such as , at the wall, sky and the window side. Figs. 4.1(d)-4.10(d) shows the histogram equalized image of (b). The image produces good color contrast, but with less information in some parts of the region with discontinuities. Figs. 4.1(e)-4.10(e) shows the output by the gradient-based method. The image appears bright, but the color information is not evenly distributed and thus, darkened in some parts of the region. Moreover, it introduces halos around the edges. We can reduce the halo effect problem by applying the extremely large Gaussian low pass filter, but that will increase the computational cost. Figs. 4.1(f)-4.10(f) shows the output by the MATLAB function 'makehdr'. The image looks faded and unnatural. Figs. 4.1(g)-4.10(g) shows the output by the pixel-wise selection method. The image produce poor picture quality. Figs. 4.1(h)-4.10(h) shows the output by the proposed method with post processing. It is evident from the results that the image provides complete details about the scene. The color information is evenly distributed throughout the image and it has realistic appearance as well.

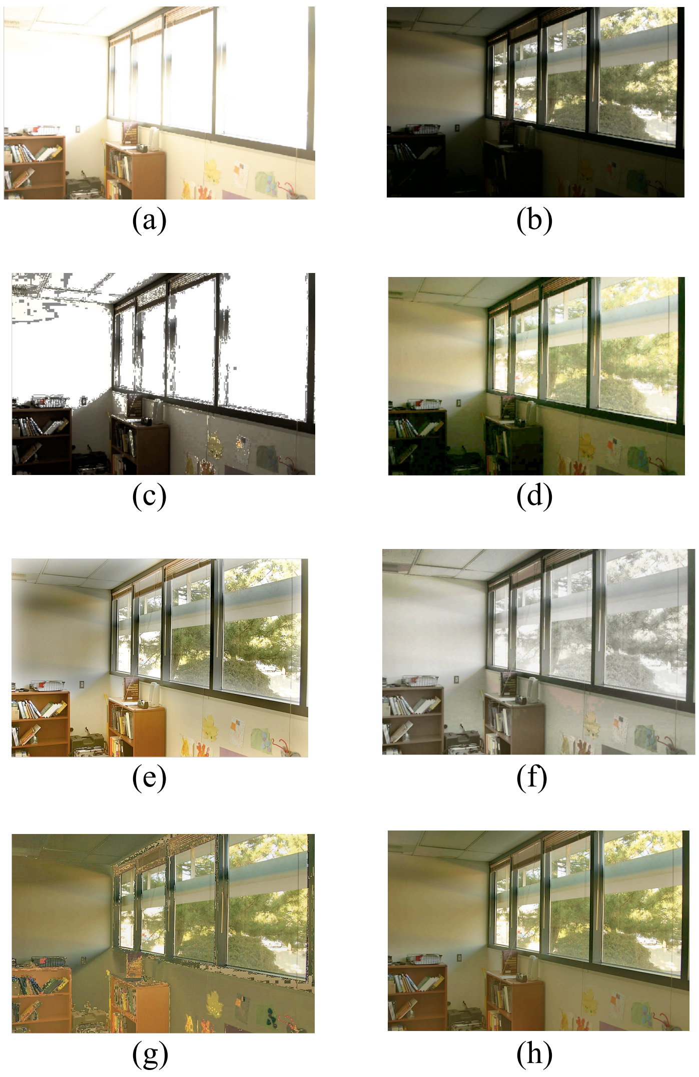
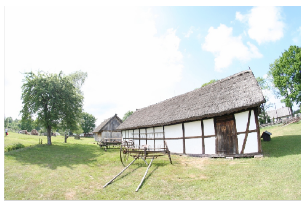
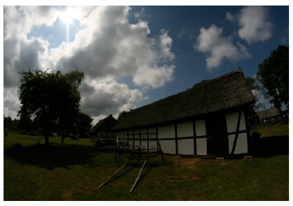
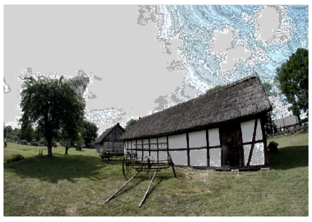
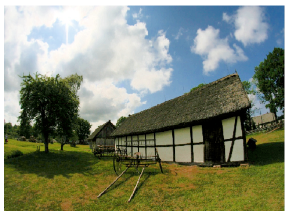


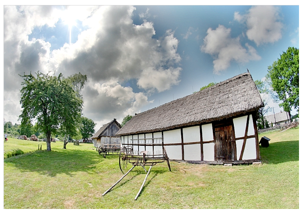
Fig. 4.1. Comparison. (a) Overexposed input image, (b) underexposed input image, (c) the histogram equalized image of (a), (d) the histogram equalized image of (b), (e) output by the gradient-based method, (f) output by the MATLAB function 'makehdr', (g) output by the saturation based pixel-wise selection method, and (h) output by the proposed method with post processing.

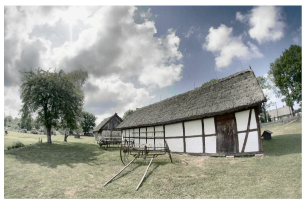

(a)

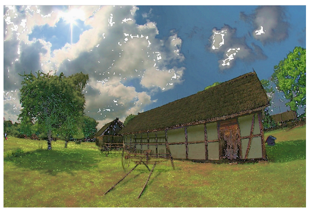

(b)


(c)


(d)


(e)


(f)


(g)

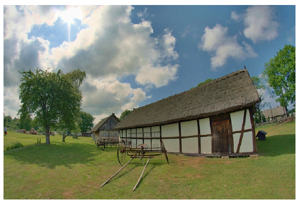

(h)

Fig. 4.2. Comparison. (a) Overexposed input image, (b) underexposed input image, (c) the histogram equalized image of (a), (d) the histogram equalized image of (b), (e) output by the gradient-based method, (f) output by the MATLAB function 'makehdr', (g) output by the pixel-wise selection method, and (h) output by the proposed method with post processing.

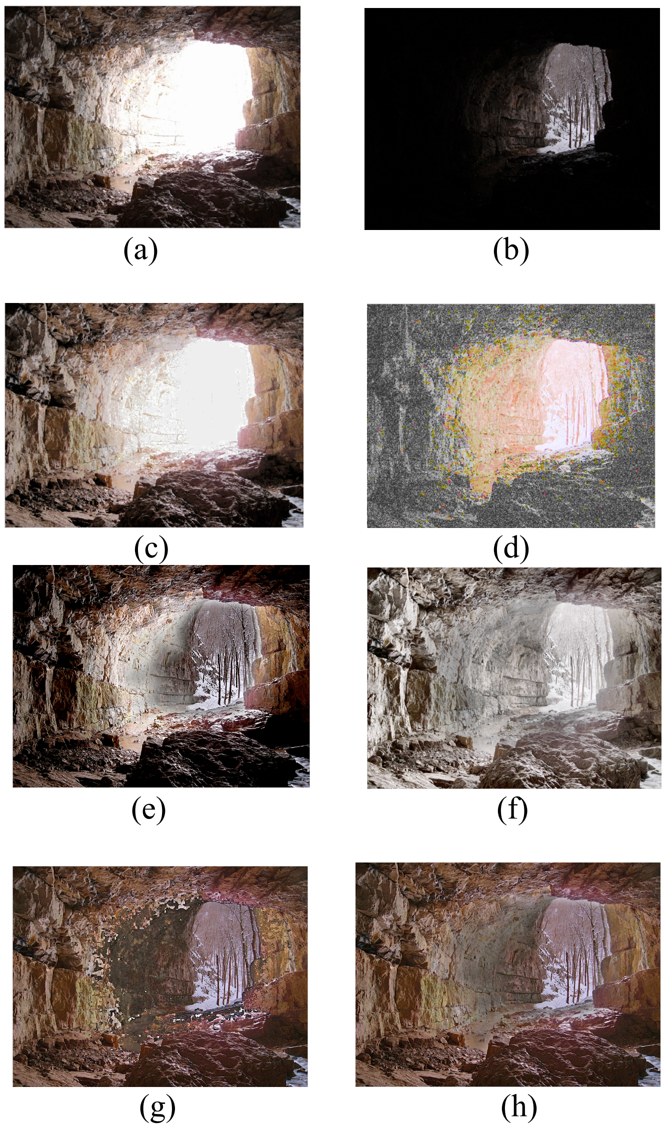


Fig. 4.3. Comparison. (a) Overexposed input image, (b) underexposed input image, (c) the histogram equalized image of (a), (d) the histogram equalized image of (b), (e) output by the gradient-based method, (f) output by the MATLAB function 'makehdr', (g) output by the pixel-wise selection method, and (h) output by the proposed method with post processing.

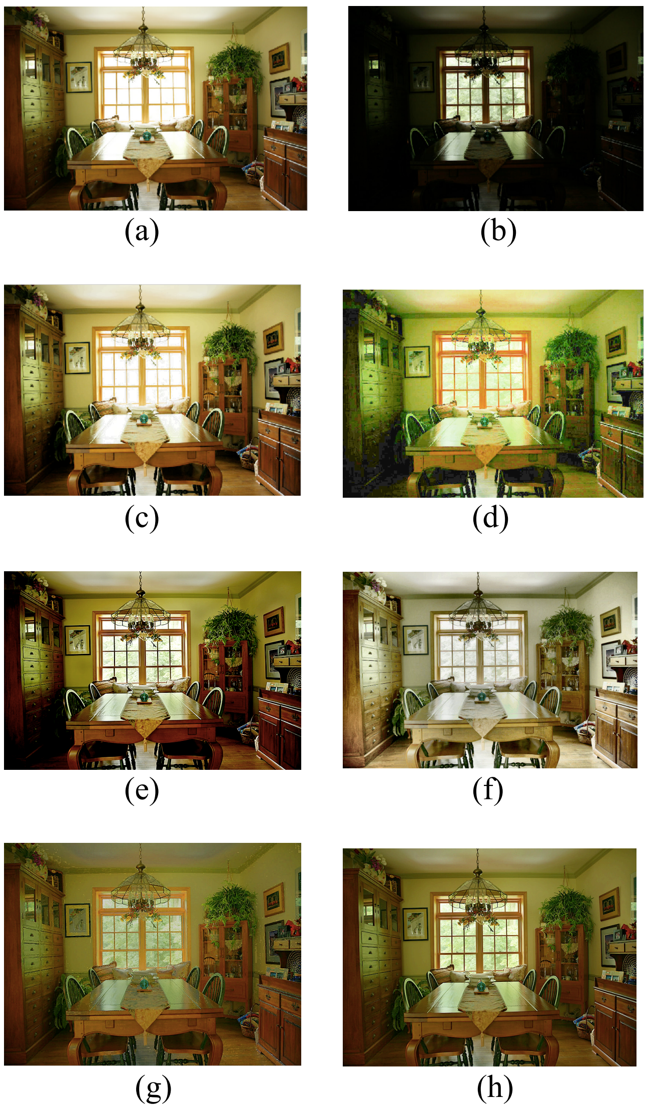


Fig. 4.4. Comparison. (a) Overexposed input image, (b) underexposed input image, (c) the histogram equalized image of (a), (d) the histogram equalized image of (b), (e) output by the gradient-based method, (f) output by the MATLAB function 'makehdr', (g) output by the pixel-wise selection method, and (h) output by the proposed method with post processing.

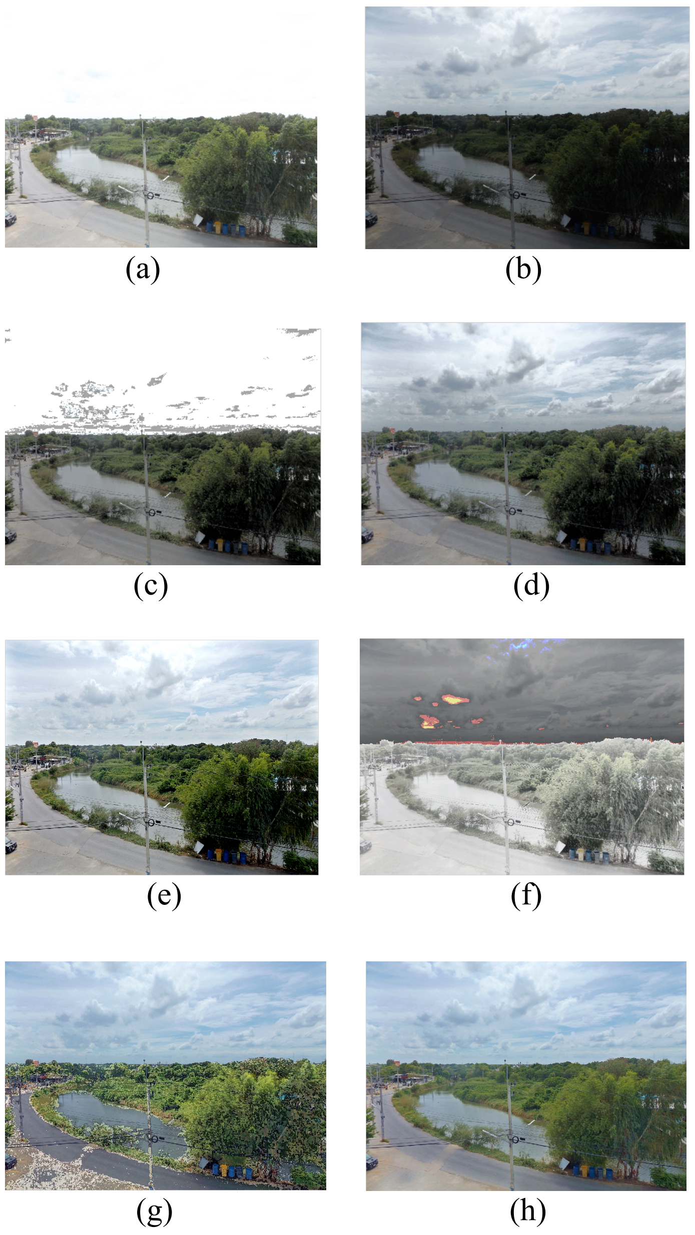


Fig. 4.5. Comparison. (a) Overexposed input image, (b) underexposed input image, (c) the histogram equalized image of (a), (d) the histogram equalized image of (b), (e) output by the gradient-based method, (f) output by the MATLAB function 'makehdr', (g) output by the pixel-wise selection method, and (h) output by the proposed method with post processing.

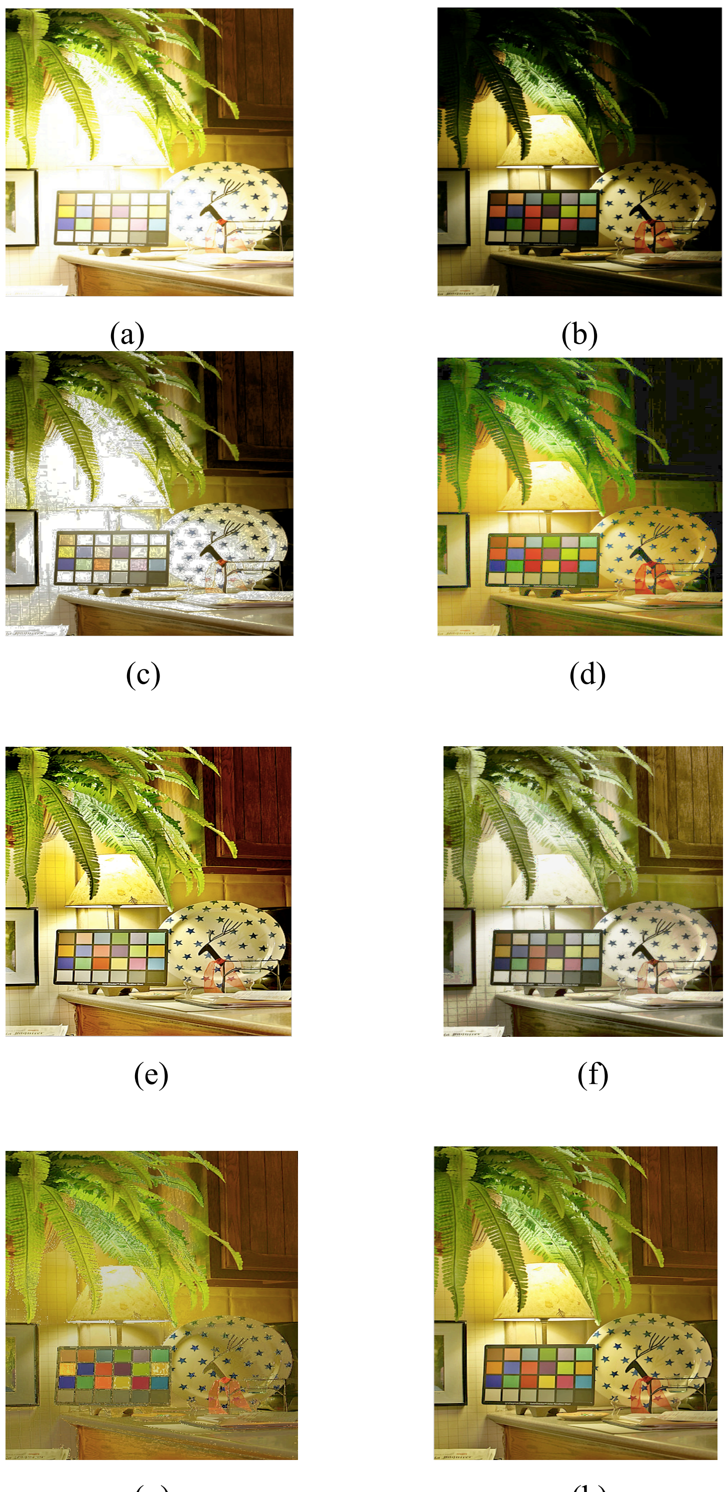


Fig. 4.6. Comparison. (a) Overexposed input image, (b) underexposed input image, (c) the histogram equalized image of (a), (d) the histogram equalized image of (b), (e) output by the gradient-based method, (f) output by the MATLAB function ‘makehdr’, (g) output by the pixel-wise selection method, and (h) output by the proposed method with post processing.

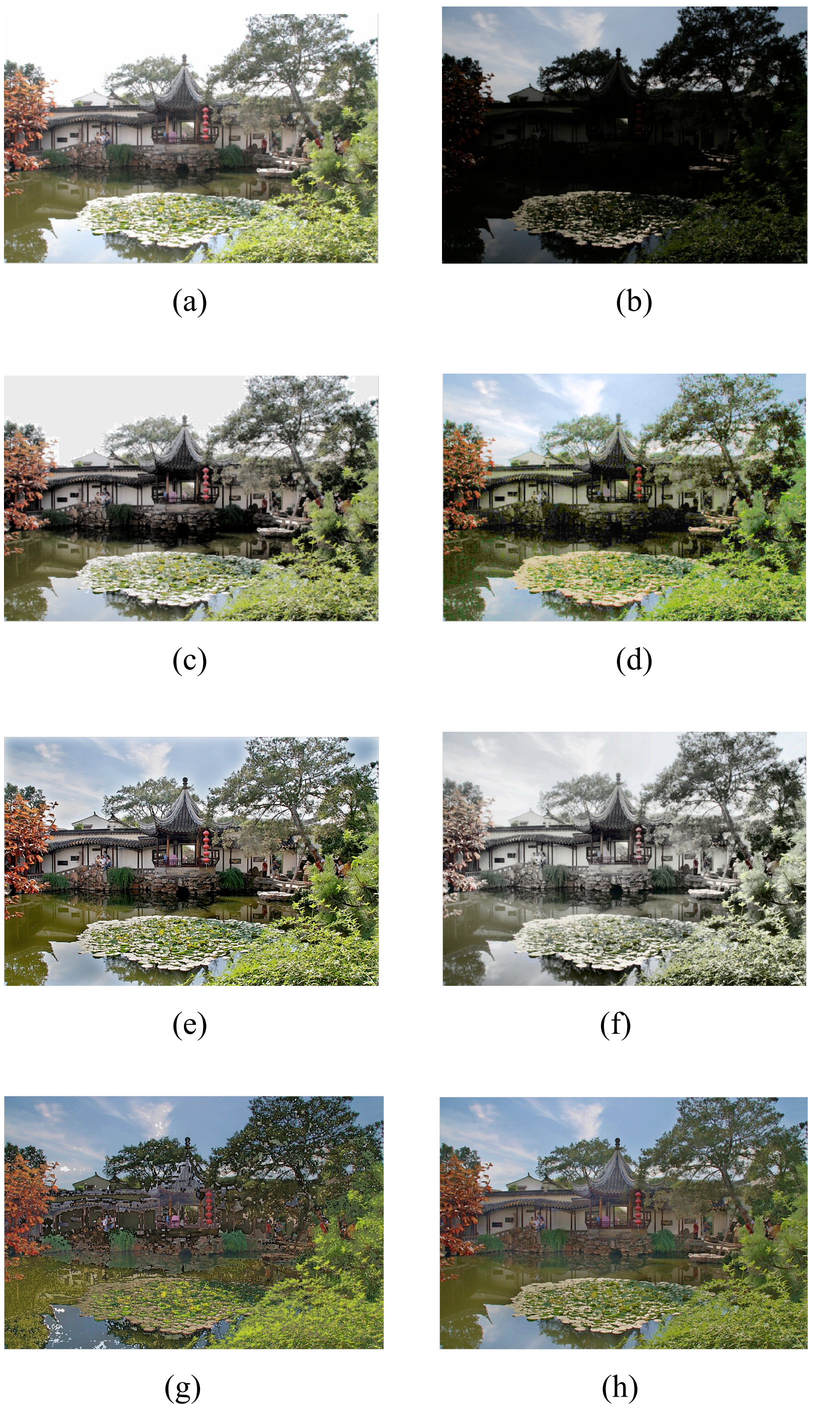


Fig. 4.7. Comparison. (a) Overexposed input image, (b) underexposed input image, (c) the histogram equalized image of (a), (d) the histogram equalized image of (b), (e) output by the gradient-based method, (f) output by the MATLAB function 'makehdr', (g) output by the pixel-wise selection method, and (h) output by the proposed method with post processing.

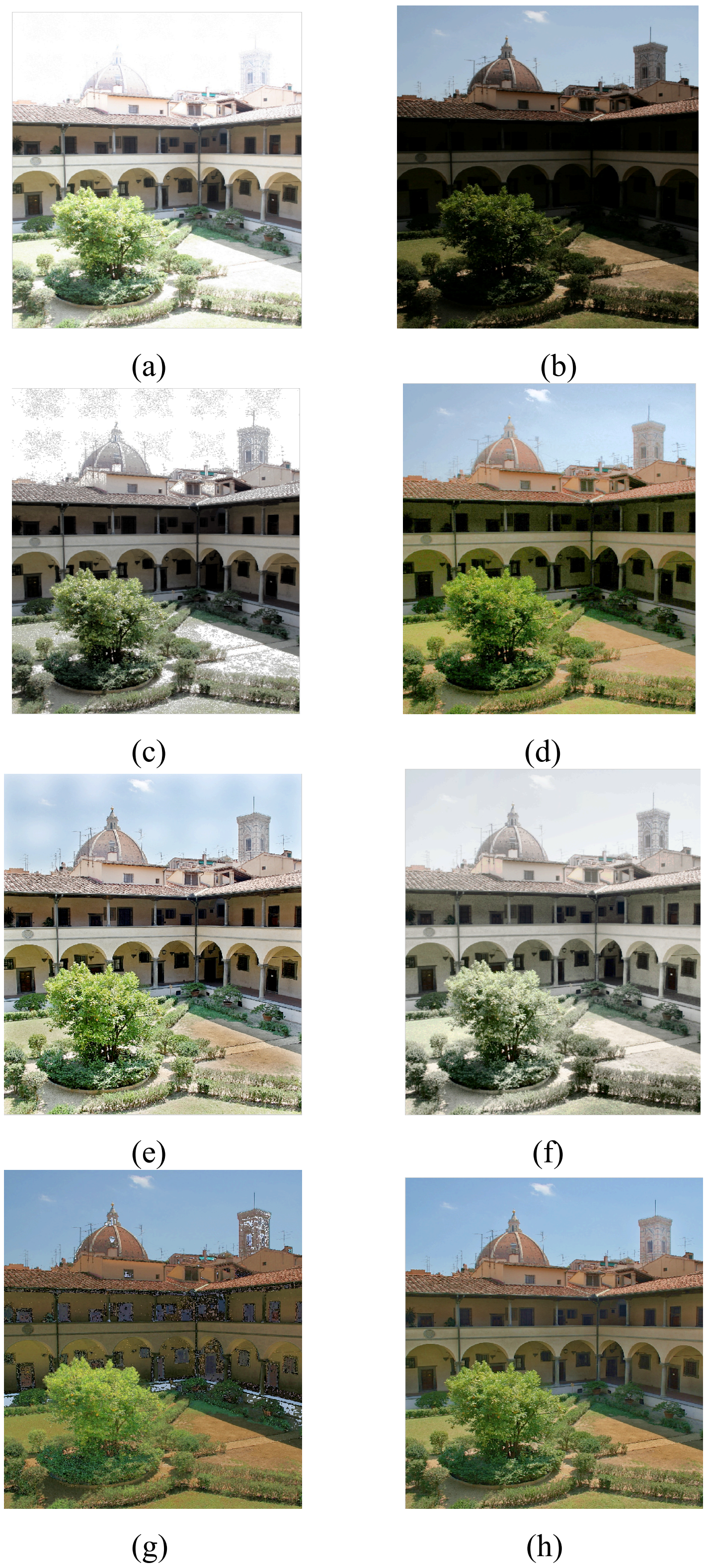
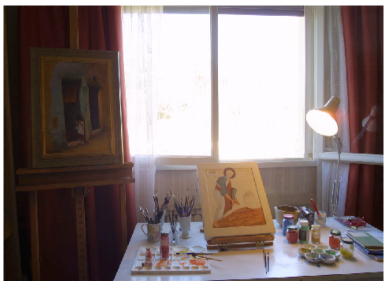
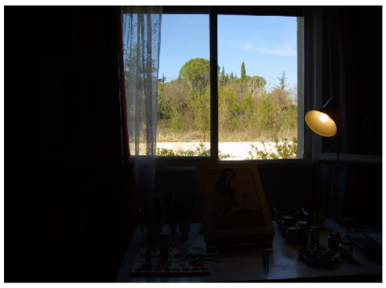
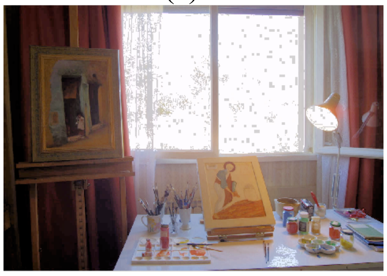
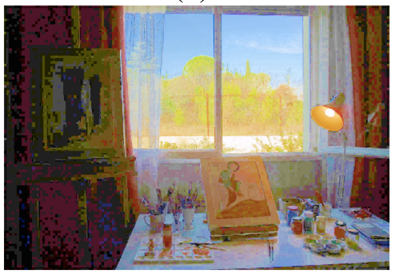


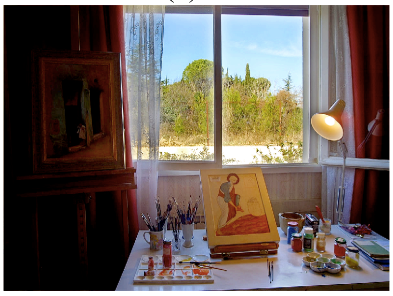
Fig. 4.8. Comparison. (a) Overexposed input image, (b) underexposed input image, (c) the histogram equalized image of (a), (d) the histogram equalized image of (b), (e) output by the gradient-based method, (f) output by the MATLAB function 'makehdr', (g) output by the pixel-wise selection method, and (h) output by the proposed method with post processing.

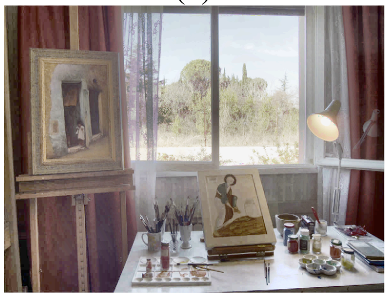

(a)

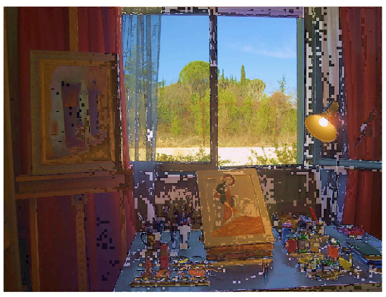

(b)


(c)


(d)


(e)


(f)


(g)

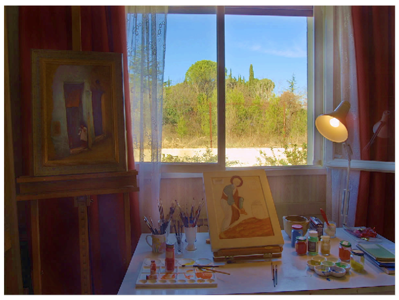

(h)

Fig. 4.9. Comparison. (a) Overexposed input image, (b) underexposed input image, (c) the histogram equalized image of (a), (d) the histogram equalized image of (b), (e) output by the gradient-based method, (f) output by the MATLAB function 'makehdr', (g) output by the pixel-wise selection method, and (h) output by the proposed method with post processing.

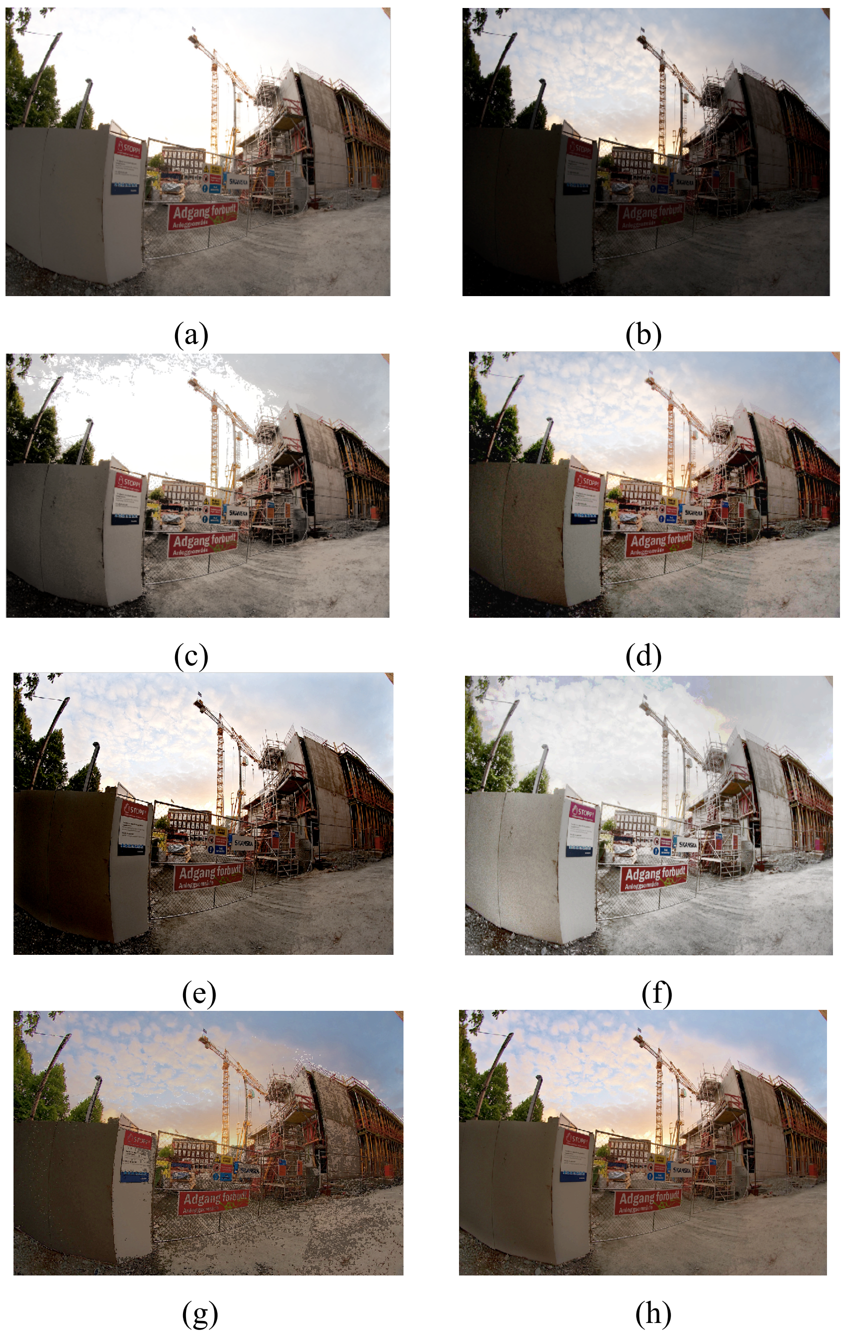


Fig. 4.10. Comparison. (a) Overexposed input image, (b) underexposed input image, (c) the histogram equalized image of (a), (d) the histogram equalized image of (b), (e) output by the gradient-based method, (f) output by the MATLAB function 'makehdr', (g) output by the pixel-wise selection method, and (h) output by the proposed method with post processing.

Table 4.1. Summary of our comparison results

Histogram equalization of overexposed image	**Histogram equalization of underexposed image**	**Gradient - based method**	**MATLAB function 'makehdr'**	**Saturation based pixel-wise selection method**	**Proposed Method with post processing**
False contours ✗	Good color contrast, but with less information in some parts and false contours ✗	Dark in some parts ✗	Faded color ✗	Poor picture quality with discontinuities ✗	Good color contrast with complete details ✓

4.2 Numerical Evaluations

Visual comparison was conducted in the previous section (Figs.4.1-4.10) and their summary in Table 4.1. The comparison shows the superiority of the proposed method over histogram equalization, a gradient-based approach, a MATLAB function, and saturation-based pixel-wise selection method. In this section, we numerically evaluate the quality of the output images by the proposed method in terms of contrast, saturation and skewness. The quality of the output images is directly compared with their input images.

Thus, we evaluate only for the input images and the output image of the proposed method. To evaluate the contrast of the input images and the output image, we first convert the rgb input images and the output image into the gray color images using the rgb2gray command. We divide the images into several blocks with fixed 5×5 block size and then evaluate the contrast of each block using RMS contrast. Then we evaluate the mean and the standard deviation of the total number of each block.

RMS contrast is defined as the standard deviation of the pixel intensities: [20]

$$\text{RMS contrast} = \sqrt{\frac{1}{MN}\sum_{i=0}^{N-1}\sum_{j=0}^{M-1}\left(I_{ij}-\bar{I}\right)^{2}} \qquad (5)$$

where I_{ij} is the intensity of the image. $\bar{I}$ is the average intensity.

From Table 4.2, it is evident that in most of the cases, the mean of the proposed method is greater than the input images, which indicates that our proposed method can generate higher contrasted images.

To evaluate the saturation of the input images and the output image, we first convert the rgb input images and the output image into the hsv channel using the rgb2hsv command and then extract the saturation channel. We divide the images into several blocks with fixed 5×5 block size. After that, we measure the mean of each block and then evaluate the mean and the standard deviation of the total number of each block.

From Table 4.3, it is evident that in most of the cases, the mean of the proposed method is greater than the input images, which indicates that our proposed method can produce satisfactory results. Thus, the experimental results show that the proposed method can generate the optimum result both visually and numerically.

Table 4.2. Block-wise contrast measurement using the RMS contrast. We deploy 13 sets of images and evaluate the mean of the contrast and the standard deviation of each input (overexposed and underexposed) image and the proposed method after post processing.

	Input overexposed image	Input underexposed image	Proposed method after post processing
1. Mean (SD)	14.54 (17.60)	11.31 (16.74)	**17.27** (18.19)
2. Mean (SD)	5.95 (10.97)	2.12 (3.65)	**6.48** (9.23)
3. Mean (SD)	14.86 (13.24)	4.36 (9.47)	**21.70** (14.75)
4. Mean (SD)	**5.78** (8.39)	2.72 (4.28)	4.90 (5.71)
5. Mean (SD)	**11.27** (14.54)	4.41 (5.14)	9.42 (9.57)
6. Mean (SD)	5.18 (10.82)	8.05 (13.90)	**10.86** (15.24)
7. Mean (SD)	13.33 (15.32)	9.54 (11.81)	**16.25** (16.33)
8. Mean (SD)	5.42 (9.33)	3.59 (6.54)	**6.02** (9.38)
9. Mean (SD)	4.35 (8.02)	3.69 (8.36)	**6.12** (8.57)
10. Mean (SD)	4.52 (8.29)	3.19 (5.35)	**4.68** (7.70)
11. Mean (SD)	5.08 (8.71)	4.19 (8.24)	**5.71** (9.17)
12. Mean (SD)	7.97 (11.34)	2.64 (4.46)	**9.03** (10.92)
13. Mean (SD)	12.16 (15.98)	6.31 (14.33)	**13.55** (17.79)

Table 4.3. Block-wise saturation measurement. We deploy 13 sets of images and evaluate the mean of the saturation and the standard deviation of each input (overexposed and underexposed) image and the proposed method after post processing.

	Input overexposed image	Input underexposed image	Proposed method after post processing
1. Mean (SD)	0.51 (0.36)	0.45 (0.26)	**0.75** (0.18)
2. Mean (SD)	0.16 (0.15)	0.23 (0.20)	**0.33** (0.18)
3. Mean (SD)	0.26 (0.17)	0.08 (0.18)	**0.42** (0.16)
4. Mean (SD)	0.14 (0.09)	**0.34** (0.16)	0.33 (0.14)
5. Mean (SD)	0.16 (0.15)	**0.39** (0.27)	0.37 (0.19)
6. Mean (SD)	0.15 (0.19)	0.33 (0.18)	**0.44** (0.14)
7. Mean (SD)	0.31 (0.22)	0.36 (0.23)	**0.51** (0.20)
8. Mean (SD)	0.14 (0.15)	0.39 (0.17)	**0.41** (0.14)
9. Mean (SD)	0.62 (0.27)	0.37 (0.26)	**0.72** (0.20)
10. Mean (SD)	0.19 (0.13)	0.35 (0.16)	**0.40** (0.15)
11. Mean (SD)	0.20 (0.18)	0.27 (0.19)	**0.37** (0.17)
12. Mean (SD)	0.06 (0.07)	0.18 (0.15)	**0.23** (0.12)
13. Mean (SD)	0.08 (0.09)	0.34 (0.14)	**0.37** (0.10)

To further address the effectiveness of our method, we employ the skewness to measure the symmetry of the data. Negative values of the skewness indicate that the data are skewed to the left. Positive values of the skewness indicate that the data are skewed to the right [21]. The skewness of the normal distribution is zero as shown in Fig. 4.11.

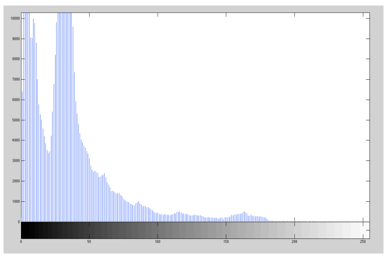
The skewness of a distribution is defined as

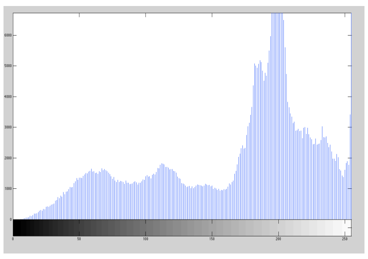
$$s = \frac{E(x-\mu)^3}{\sigma^3} \qquad (6)$$

Where μ is the mean of x, σ is the standard deviation of x, and $E(t)$ represents the expected value of the quantity t.

The histogram is an effective graphical technique for showing the skewness of data set. Thus, we first convert the rgb input images into the gray color images using the rgb2gray command. Then, we use the imhist command to show the histogram of the images. Finally, skewness command is applied to the gray image.

The skew of the histogram of an under-exposed image is normally positive as it is positively skewed (Fig. 4.11(a)). By contrast, the skew of the histogram of an over-exposed image is supposed to be negative as it should be negatively skewed (Fig. 4.11(b)). However, some of the over-exposed images (Nos. 3, 9, and 13 in Table 4.4) show positive skewnesses. In these minor cases, the areas of the over-exposed regions are rather limited, far less than 50% of the entire images, and yet their histograms are positively skewed (Fig. 4.12(a)). If over-exposed regions are corrected by the proposed method, the histograms are more positively skewed or evenly distributed (Fig. 4.12(b)). In these cases, the skewness may not be a positive indicator to show the effectiveness of the proposed method. In general, however, the three statistics (mean, median, and mode) among the skewnesses of 13 images show that the proposed method produces images of better quality as shown in Table 4.4.


(a)


(b)

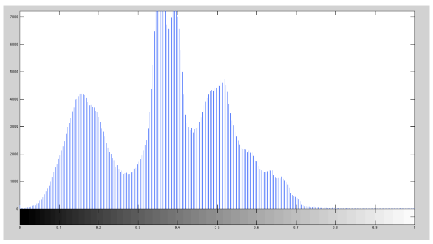
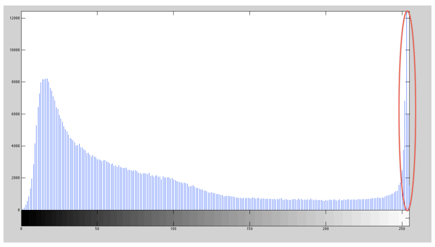

(c)

Fig. 4.11. Skewness of the histogram. (a) Underexposed input image (skewness>0). (b) Overexposed input image (skewness<0). (c) Proposed method (skewness≈0).


(a)

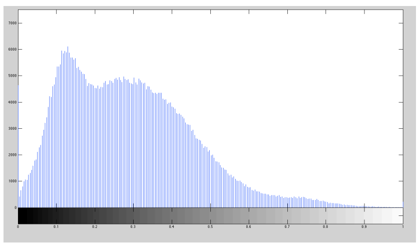

(b)

Fig. 4.12. A minor case of overexposed images with positive skewness. (a) Overexposed input image with positive skewness >0. (b) Proposed method after correction.

Table 4.4. Skewness

S.I	Underexposed>0	Overexposed<0	Proposed method
1.	1.3099	-0.3921	0.4568
2.	1.6267	-0.9886	**-0.3153**
3.	3.4353	0.7249	**0.7204**
4.	2.1222	-0.7044	**-0.0528**
5.	1.4424	-0.9448	**0.7772**
6.	1.1381	-1.4905	**1.0132**
7.	0.3737	-0.3858	**-0.2589**
8.	0.6893	-1.3734	**-0.1168**
9.	1.7657	0.5400	0.9226
10.	0.8689	-0.5456	**0.0355**
11.	1.1482	-0.2733	**0.0876**
12.	0.8100	-0.6041	**0.0159**
13.	2.8574	0.7285	1.2273
Mean	1.5068	0.7458	**0.4616**
Median	1.3099	0.7044	**0.3153**
Mode	0.3737	0.2733	**0.0159**

Chapter 5
Conclusions and Recommendations

This paper presents a simple saturation-based image fusion technique for static scenes, which is designed for generating high dynamic range images fully automatically at a low computational cost. The method produces a high dynamic range image as a linear combination of a set of input images with different exposure levels. The weights for blending the input images are solely dependent on the saturation values of them. Larger saturation values indicate more vivid colors. Thus, we assign heavy weights to the pixels with high saturations, while light weights are applied to the pixels whose saturation values are low. The distribution of the weighting factors is then smoothed using a large Gaussian low-pass filter. This step is effective to suppress the halo effect that can be often seen in the images fused by other techniques. Experimental results show that the proposed method can automatically produce realistic high dynamic range images. The quality of the fused images by the proposed method is satisfactory both visually and numerically. A common limitation of image fusion techniques is that it is difficult to produce a satisfactory result when the inputs images are all overexposed or underexposed. It should be noted that image details must be captured in at least one of the input images.

As future work, we plan to work on image registration in order that the proposed method can be applied to motion pictures.

References

[1] Aggarwal, M. and N. Ahuja. (2004). Split aperture imaging for high dynamic range. *International Journal of Computer Vision*, *58*(1), 7-17.

[2] Seetzen, H., et al. (2004). High dynamic range display systems. *ACM Transactions on Graphics (TOG), 23*(3), 760-768.

[3] Tocci, M.D., et al. (2011). A versatile HDR video production system. *ACM Transactions on Graphics (TOG), 30*(4), 41.

[4] Debevec, P.E. and J. Malik. (2008). Recovering high dynamic range radiance maps from photographs. *ACM SIGGRAPH 2008 classes,* 31.

[5] Durand, F. and J. Dorsey. (2002). Fast bilateral filtering for the display of high-dynamic-range images. *ACM transactions on graphics (TOG), 21*(3), 257-266.

[6] Reinhard, E., et al. (2002). Photographic tone reproduction for digital images. *ACM Transactions on Graphics (TOG), 21*(3), 267-276.

[7] Fattal, R., D. Lischinski, and M. Werman. (2002). Gradient domain high dynamic range compression. *ACM Transactions on Graphics (TOG)*, *21*(3), 249-256.

[8] Burt, P.J. and R.J. Kolczynski. (1993). Enhanced image capture through fusion. *Proceedings., Fourth International Conference,11-14 May 1993* (pp.173-182). Berlin: IEEE.

[9] Raskar, R., A. Ilie, and J. Yu. (2005). Image fusion for context enhancement and video surrealism. *ACM SIGGRAPH 2005 Courses*, 4.

[10] Goshtasby, A.A. (2005). Fusion of multi-exposure images. *Image and Vision Computing, 23*(6), 611-618.

[11] Mertens, T., J. Kautz, and F. Van Reeth. (2009). Exposure fusion: A simple and practical alternative to high dynamic range photography. *Computer Graphics Forum*, *28*(1), 161-171.

[12] Zhang, W. and W.-K. Cham. (2010). Gradient-directed composition of multi-exposure images. *Computer Vision and Pattern Recognition (CVPR), 13-18 June 2010* (pp.530-536). San Francisco,CA: IEEE.

[13] Raman, S. and S. Chaudhuri. (2009). Bilateral filter based compositing for variable exposure photography. *Proceedings of Eurographics*.

[14] Várkonyi-Kóczy, A.R., A. Rövid, and T. Hashimoto. (2008). Gradient-based synthesized multiple exposure time color HDR image. *Instrumentation and Measurement, IEEE Transactions on*, *57*(8), 1779-1785.

[15] Malik, M.H., S. Asif, and M. Gilani. (2008). Wavelet based exposure fusion.

[16] Gonzalez, R.C. and R.E. Woods. (2002). *Digital image processing*. Upper Saddle River, NJ: Prentice hall.

[17] http://www.mathworks.com/help/images/ index.html

[18] http://www.easyhdr.com/examples/

[19] http://kirkt.smugmug.com/Photography

[20] https://en.wikipedia.org/wiki/Contrast_(vision)

[21] https://www.mathworks.com/help/stats/skewness.html?searchHighlight= skewness

Appendices

Appendix A
Source Code of Proposed Method

```
###------------------image loading---------------------###

% imageload1=imread('overlight.jpg'); image1=imresize(imageload1,1);
% imageload2=imread('underlight.jpg'); image2=imresize(imageload2,1);
% imageload1=imread('overcave.jpg'); image1=imresize(imageload1,1);
% imageload2=imread('undercave.jpg'); image2=imresize(imageload2,1);
% imageload1=imread('overbridge.jpg'); image1=imresize(imageload1,1); %
imageload2=imread('underbridge.jpg');image2=imresize(imageload2,1);
% imageload1=imread('overhouse.jpg'); image1=imresize(imageload1,1);
% imageload2=imread('underhouse.jpg'); image2=imresize(imageload2,1);
% imageload1=imread('overwindow.jpg'); image1=imresize(imageload1,1);
% imageload2=imread('underwin.jpg'); image2=imresize(imageload2,1);
% imageload1=imread('office_6.jpg'); image1=imresize(imageload1,1);
% imageload2=imread('office_2.jpg'); image2=imresize(imageload2,1);
% imageload1=imread('overdoll.jpg'); image1=imresize(imageload1,1);
% imageload2=imread('underdoll.jpg'); image2=imresize(imageload2,1);
% imageload1=imread('overgate.jpg'); image1=imresize(imageload1,1);
% imageload2=imread('undergate.jpg'); image2=imresize(imageload2,1);
% imageload1=imread('overEV+2.jpg'); image1=imresize(imageload1,1);
% imageload2=imread('underEV-2.jpg'); image2=imresize(imageload2,1);
% imageload1=imread('overwater.jpg'); image1=imresize(imageload1,1);
% imageload2=imread('underwater.jpg'); image2=imresize(imageload2,1);
% imageload1=imread('overtable.jpg');image1=imresize(imageload1,1);
% imageload2=imread('undertable.jpg');image2=imresize(imageload2,1);
% imageload1=imread('brightimage.jpg');image1=imresize(imageload1,1);
% imageload2=imread('darkimage.jpg');image2=imresize(imageload2,1);
% imageload1=imread('overtemple.jpg'); image1=imresize(imageload1,1);
% imageload2=imread('undertemple.jpg');image2=imresize(imageload2,1);
 imageload1=imread('overtajmahal.jpg');image1=imresize(imageload1,1);
imageload2=imread('undertajmahal.jpg');image2=imresize(imageload2,1);
% imageload1=imread('over.jpg'); image1=imresize(imageload1,1);
% imageload2=imread('under.jpg'); image2=imresize(imageload2,1);
% imageload1=imread('overwall.jpg'); image1=imresize(imageload1,1);
% imageload2=imread('underwall.jpg'); image2=imresize(imageload2,1);
%imageload1=imread('overbuilding.jpg');image1=imresize(imageload1,1);
%imageload2=imread('underbuilding.jpg');image2=imresize(imageload2,1)
% imageload1=imread('overpillar.jpg'); image1=imresize(imageload1,1);
% imageload2=imread('underpillar.jpg');image2=imresize(imageload2,1);
% imageload1=imread('overnight.jpg'); image1=imresize(imageload1,1);
% imageload2=imread('undernight.jpg'); image2=imresize(imageload2,1);
% imageload1=imread('overalter.jpg'); image1=imresize(imageload1,1);
% imageload2=imread('underalter.jpg'); image2=imresize(imageload2,1);
% imageload1=imread('C.jpg');image1=imresize(imageload1,1);
% imageload2=imread('B.jpg');image2=imresize(imageload2,1);
% imageload1=imread('oversunset.jpg'); image1=imresize(imageload1,1);
% imageload2=imread('undersunset.jpg');image2=imresize(imageload2,1);
% imageload1=imread('overforest.jpg'); image1=imresize(imageload1,1);
% imageload2=imread('underforest.jpg');image2=imresize(imageload2,1);
% imageload1=imread('over.tiff'); image1=imresize(imageload1,1);
% imageload2=imread('under.tiff'); image2=imresize(imageload2,1);
% imageload1=imread('flat.tif'); image1=imresize(imageload1,1);
% imageload2=imread('flatunder.tif'); image2=imresize(imageload2,1);%
% imageload1=imread('geleyover.jpg'); image1=imresize(imageload1,1);
% imageload2=imread('geleyunder.jpg'); image2=imresize(imageload2,1);
% imageload1=imread('overgolfview.jpg');       image1=imresize(imageload1,1/4);
% imageload2=imread('undergolfview.jpg'); image2=imresize(imageload2,1/4);
% imageload1=imread('roomover.jpg'); image1=imresize(imageload1,1/4);
% imageload2=imread('roomunder.jpg');image2=imresize(imageload2,1/4);


###---------------end of image loading-------------------###


[v,h]=size(image1)
h=h/3

figure(1),
subplot(1,2,1),imshow(image1), xlabel('(1) overexposed image');
subplot(1,2,2),imshow(image2),xlabel('underexposed image');

gray1=rgb2gray(image1); gray2=rgb2gray(image2);

###----------histogram equalization of a input gray image---------###

figure(2),
subplot(2,2,1),imshow(gray1),xlabel(' overexposed  gray image');
subplot(2,2,2),imshow(gray2),xlabel(' underexposed  gray image');
subplot(2,2,3),imhist(gray1),title('histogram of overexposed image');
subplot(2,2,4),imhist(gray2),title('histogram of underexposed image');

stretch1=imadjust(gray1);
stretch2=imadjust(gray2);

equalized1=histeq(gray1);
equalized2=histeq(gray2);
```

```
figure(3),
subplot(2,2,1),imshow(stretch1),title(' stretched overexposed  gray image');
subplot(2,2,2),imshow(stretch2),title(' stretched underexposed  gray image');
subplot(2,2,3),imhist(stretch1);
subplot(2,2,4),imhist(stretch2);

figure(4),
subplot(2,2,1),imshow(equalized1),title('histogram equalized of overexposed gray
image');




subplot(2,2,2),imshow(equalized2),title('histogram equalized of underexposed gray
image');
subplot(2,2,3),imhist(equalized1);
subplot(2,2,4),imhist(equalized2);


###------end of histogram equalization of a input gray image-----###


###------extracting saturation channel-------------------###

hsv1=rgb2hsv(image1); hsv2=rgb2hsv(image2);
histogram1=histeq(hsv1(:,:,3));
histogram2=histeq(hsv2(:,:,3));
figure(5),
subplot(2,2,1),imshow(histogram1),title('histogram equalized overexposed image for
intensity channel');
subplot(2,2,2),imshow(histogram2),title('histogram equalized underexposed image for
intensity channel');
subplot(2,2,3),imhist(histogram1),title('histogram of histeq overexposed  intensity
channel image') ;
subplot(2,2,4),imhist(histogram2);title('histogram of histeq underexposed  intensity
channel image');


 hsv1(:,:,3)=histogram1;
 finalimage=hsv2rgb(hsv1);
 hsv2(:,:,3)=histogram2;
 finalimage1=hsv2rgb(hsv2);

 figure(6),
 subplot(1,2,1), imshow(finalimage),xlabel('histogram equalization of over exposed
image');
 subplot(1,2,2), imshow(finalimage1),xlabel('histogram equalization of under exposed
image');

saturationimage1=hsv1(:,:,2); %saturation
saturationimage2=hsv2(:,:,2);

figure(7),
subplot(2,2,1),imshow(saturationimage1),xlabel('(2) saturation of overexposed
image');
subplot(2,2,2),imshow(saturationimage2),xlabel('saturation of underexposed image');

###---------------end of saturation---------------------###

###------------weight of two images--------------------###

 mergeimage=saturationimage1+saturationimage2;

 zeroindex1=find(saturationimage2>0.9); %>0.9
  saturationimage2(zeroindex1)=0;


  figure(7),
  subplot(2,2,3),
  imshow(saturationimage2),xlabel('Saturation of underexposed   image');


 wght11=saturationimage1./mergeimage; % apply weighting averaging to the input
images
 wght22=saturationimage2./mergeimage;


 zeroindex=find(mergeimage==0);
 wght11(zeroindex)=0.5;
 wght22(zeroindex)=0.5;


  figure(8),

  subplot(2,2,1),imshow(mat2gray(wght11)),xlabel('(3) before smoothing weights of
overexposed image');
  subplot(2,2,2),imshow(mat2gray(wght22)),xlabel('before smoothing weights of
underexposed image');


    mask=round(v/10)


lowpassfilter=fspecial('gaussian',mask,mask/4);
```

```
wght111=imfilter(wght11,lowpassfilter, 'replicate');
wght222=1-wght111;


figure(8),


subplot(2,2,3),imshow(mat2gray(wght111)),xlabel('weight for overexposed image');
subplot(2,2,4),imshow(mat2gray(wght222)),xlabel('weight for underexposed image');


image1=double(image1);
image2=double(image2);



hdr(:,:,1)=wght111.*image1(:,:,1)+wght222.*image2(:,:,1);
hdr(:,:,2)=wght111.*image1(:,:,2)+wght222.*image2(:,:,2);
hdr(:,:,3)=wght111.*image1(:,:,3)+wght222.*image2(:,:,3);


figure(9),


 result=mat2gray(hdr);
 imshow(result),xlabel('final image before post processing ');


% %----------------------after applying post processing---------%


  sharp=imsharpen(result);
  hsvimg=rgb2hsv(sharp);

saturation=hsvimg(:,:,2);
intensity=hsvimg(:,:,3);


 hsvimg(:,:,2)=imadjust(saturation,[],[],0.7);
 hsvimg(:,:,3)=imadjust(intensity,[],[],0.7);

 figure(10),
 imshow(hsv2rgb(hsvimg)),xlabel('final HDR image after post processing');
```

```
% -------------------end of post processing -------------------------------------- %


% ---------------------matlab built-in function-------------------------- %

% files={'office_2.jpg','office_6.jpg'};  expTimes = [0.1000, 4.0000];
% files = {'underdoll.jpg', 'overdoll.jpg'};expTimes = [0.0222, 0.3333];
% files = {'underwater.jpg', 'overwater.jpg'};expTimes = [0.0013, 0.0222];
% files = {'underEV-2.jpg', 'overEV+2.jpg'};expTimes = [0.008, 0.125];
% files = {'underlight.jpg', 'overlight.jpg'};expTimes = [0.008, 0.125];
% files={'underlight.jpg','overlight.jpg'}; expTimes=[0.0666,1];
% files={'underwin.jpg','overwindow.jpg'};expTimes=[0.0333,1.0000];
% files={'undertable.jpg','overtable.jpg'};expTimes=[0.125,1.0000];

files = {'overgolfview.jpg', 'undergolfview.jpg'};expTimes = [-2, +2];
%    files = {'underhouse.jpg', 'overhouse.jpg'};expTimes = [-2, +2];
% files = {'underenvironment.tiff', 'overenvironment.tiff'};   expTimes = [-2, +2];
% files = {'undertemple.jpg', 'overtemple.jpg'};
% expTimes = [-2, +2];
%  files = {'undercave.jpg', 'overcave.jpg'};   expTimes = [-2, +2];
% files = {'undertajmahal.jpg', 'overtajmahal.jpg'};
% files = {'underbridge.jpg', 'overbridge.jpg'};  expTimes = [-2, +2];
% files={'underwall.jpg','overwall.jpg'};
% files={'underfactory.jpg','overfactory.jpg'}; expTimes = [-2, +2];
% files={'underpillar.jpg','overpillar.jpg'};expTimes = [-2, +2];
% files={'undernight.jpg','overnight.jpg'};expTimes = [-2, +2];
% files={'underalter.jpg','overalter.jpg'}; expTimes = [-2, +2];
% files={'B.jpg','C.jpg'}; expTimes = [-2, +2];
% files={'undersunset.jpg','oversunset.jpg'};
% files={'flatunder.tif','flat.tif'};expTimes = [4, 30];
% files={'Harborunder.tif','Izmir Harbor.tif'};expTimes = [1, 30];
% files={'underverynight.jpg','oververynight.jpg'};
% files={'under_sand.jpg','over_sand.jpg'};
% files={'The Alley - 04.tif','The Alley - 02.tif'};
% files={'Mountain Shed - 01.tif','Mountain Shed - 03.tif'};

%...............................................end................................. %

%    hdr=makehdr(files);
%    rgb=tonemap(hdr);
      hdr = makehdr(files,'ExposureValues',expTimes);
%      hdr = makehdr(files, 'RelativeExposure', expTimes./expTimes(1));
rgb=tonemap(hdr);
    figure(11); imshow(rgb),xlabel('Matlab built-in function makehdr');

% -----------------------end-----------------------------------%
```

Appendix B
Source Code of a Gradient-based Method

```
% function HD
% iptsetpref('ImshowAxesVisible','off')
clear all; close all;


% -----------------------------image loading---------------------%

%   org1=imread('overlight.jpg'); img1=imresize(org1,1);
%   org2=imread('underlight.jpg'); img2=imresize(org2,1);
%   org1=imread('overhouse.jpg'); img1=imresize(org1,1);
%   org2=imread('underhouse.jpg'); img2=imresize(org2,1);
%   org1=imread('overtajmahal.jpg'); img1=imresize(org1,1);
%   org2=imread('undertajmahal.jpg'); img2=imresize(org2,1);
%   org1=imread('overbridge.jpg'); img1=imresize(org1,1);
%   org2=imread('underbridge.jpg'); img2=imresize(org2,1);
%   org1=imread('overcave.jpg'); img1=imresize(org1,1);
%   org2=imread('undercave.jpg'); img2=imresize(org2,1);
%   org1=imread('overdoll.jpg'); img1=imresize(org1,1);
%   org2=imread('underdoll.jpg'); img2=imresize(org2,1);
%   org1=imread('overtemple.jpg'); img1=imresize(org1,1);
%   org2=imread('undertemple.jpg'); img2=imresize(org2,1);
%   org1=imread('overwindow.jpg'); img1=imresize(org1,1);
%   org2=imread('underwin.jpg'); img2=imresize(org2,1);
%   org1=imread('overEV+2.jpg'); img1=imresize(org1,1);
%   org2=imread('underEV-2.jpg'); img2=imresize(org2,1);
%   org1=imread('overtable.jpg'); img1=imresize(org1,1);
%   org2=imread('undertable.jpg'); img2=imresize(org2,1);
%   org1=imread('overwater.jpg'); img1=imresize(org1,1);
%   org2=imread('underwater.jpg'); img2=imresize(org2,1);
%   org1=imread('overgate.jpg'); img1=imresize(org1,1);
%   org2=imread('undergate.jpg'); img2=imresize(org2,1);
%   org1=imread('overenvironment.tiff'); img1=imresize(org1,1);
%   org2=imread('underenvironment.tiff'); img2=imresize(org2,1);
%   org1=imread('overwall.jpg'); img1=imresize(org1,1);
%   org2=imread('underwall.jpg'); img2=imresize(org2,1);
    org1=imread('overfactory.jpg'); img1=imresize(org1,1);
    org2=imread('underfactory.jpg'); img2=imresize(org2,1);
```

```
%    org1=imread('Mountain Shed - 02.tif'); img1=imresize(org1,1/4);
%    org2=imread('Mountain Shed - 03.tif'); img2=imresize(org2,1/4);
%    org1=imread('overalter.jpg'); img1=imresize(org1,1);
%    org2=imread('underalter.jpg'); img2=imresize(org2,1);
%    org1=imread('C.jpg'); img1=imresize(org1,1);
%    org2=imread('B.jpg'); img2=imresize(org2,1);
%    org1=imread('oversunset.jpg'); img1=imresize(org1,1);
%    org2=imread('undersunset.jpg'); img2=imresize(org2,1);

%    org1=imread('overgolfview.jpg'); img1=imresize(org1,1/4);
%    org2=imread('undergolfview.jpg'); img2=imresize(org2,1/4);
%    org1=imread('roomover.jpg'); img1=imresize(org1,1/4);
%    org2=imread('roomunder.jpg'); img2=imresize(org2,1/4);

%-------------------juna image ---------------------------------------%

%    org1=imread('10.1.jpg'); img1=imresize(org1,1);% sunset
%    org2=imread('10.8.jpg'); img2=imresize(org2,1);% sunset
%    org1=imread('8.1.jpg'); img1=imresize(org1,1);%  museum
%    org2=imread('8.8.jpg'); img2=imresize(org2,1);%  museum
%    org1=imread('9.1.jpg'); img1=imresize(org1,1);% beach
%    org2=imread('9.7.jpg'); img2=imresize(org2,1);% beach
%    org1=imread('20.1.jpg'); img1=imresize(org1,1);% croud
%    org2=imread('20.2.jpg'); img2=imresize(org2,1);% croud
%    org1=imread('16.1.jpg'); img1=imresize(org1,1);% SIIT
%    org2=imread('16.2.jpg'); img2=imresize(org2,1);% SIIT
%    org1=imread('12.1.jpg'); img1=imresize(org1,1);% road
%    org2=imread('12.2.jpg'); img2=imresize(org2,1);% road
%    org1=imread('14.1.jpg'); img1=imresize(org1,1);% library
%    org2=imread('14.2.jpg'); img2=imresize(org2,1);% library
%    org1=imread('15.1.jpg'); img1=imresize(org1,1); % umbrella
%    org2=imread('15.2.jpg'); img2=imresize(org2,1); % umbrella
%    org1=imread('6.1.jpg'); img1=imresize(org1,1); % learning center
%    org2=imread('6.8.jpg'); img2=imresize(org2,1); % learning center
%    org1=imread('19.1.jpg'); img1=imresize(org1,1/4);% bangkadi
%    org2=imread('19.2.jpg'); img2=imresize(org2,1/4);% bangkadi

% --------------- end of loading----------------------%

figure(15)

subplot(1,2,1),imshow(img1),xlabel('(a) Over-exposed image')
subplot(1,2,2),imshow(img2),xlabel('(b) Under-exposed image')

clear org1 org2
```

```
%-------------- apply histogram equalization to the intensity channel--------------- %

hsv1=rgb2hsv(img1);
hsv2=rgb2hsv(img2);
figure(20)
subplot(1,2,1),imhist(hsv1(:,:,3)),title('(a)histogram of  Over-exposed image')
subplot(1,2,2),imhist(hsv2(:,:,3)),title('(b) histogram of  Under-exposed image')

re_hsv1=cat(3,hsv1(:,:,1),hsv1(:,:,2),histeq(hsv1(:,:,3)));
re_hsv2=cat(3,hsv2(:,:,1),hsv2(:,:,2),histeq(hsv2(:,:,3)));

figure(25)

subplot(2,2,1),imshow(re_hsv1(:,:,3)),xlabel('histogram equalized of overexposed
intensity channel image');
subplot(2,2,2),imshow(re_hsv2(:,:,3)),xlabel('histogram equalized of underexposed
intensity channel image');
subplot(2,2,3),imhist(re_hsv1(:,:,3)),title('(a)histogram of hist equalized of
Overexposed intensity image')
subplot(2,2,4),imhist(re_hsv2(:,:,3)),title('(b)histogram of hist equalized of
Underexposed intensity image')

RGBout1=hsv2rgb(re_hsv1);
RGBout2=hsv2rgb(re_hsv2);

figure(30)

subplot(1,2,1),imshow(RGBout1),xlabel('(a) histogram equalized of RGB Over-
exposed image')
subplot(1,2,2),imshow(RGBout2),xlabel('(b) histogram equalized of RGB Under-
exposed  image')

%------- define filter as sobel (edge detection) for the v and hmask---------%

vmask=fspecial('Sobel'); hmask=vmask';

gray1=rgb2gray(img1); gray2=rgb2gray(img2);
figure(35),
subplot(1,2,1),imshow(gray1),xlabel('(a) Over-exposed  gray image')
subplot(1,2,2),imshow(gray2),xlabel('(b) Under-exposed gray image')

[v,h]=size(gray1);

figure(40)

gx=filter2(vmask,double(gray1),'same'); % sobel operator applied to the x axis
(vertical )
```

```
subplot(2,2,1),imshow(mat2gray(gx)),xlabel('gx1 of overexposed gray image')
gy=filter2(hmask,double(gray1),'same'); % sobel operator applied to the y axis
(horizontal)
subplot(2,2,2),imshow(mat2gray(gy)),xlabel('gy1 of overexposed gray image')
mag1=sqrt(gx.*gx+gy.*gy); % magnitude
subplot(2,2,3),imshow(mat2gray(mag1)),xlabel('Magnitude1 of overexposed gray
image')


figure(45)

gx=filter2(vmask,double(gray2),'same');
subplot(2,2,1),imshow(mat2gray(gx)),xlabel('gx2 of underexposed gray image')
gy=filter2(hmask,double(gray2),'same');
subplot(2,2,2),imshow(mat2gray(gy)),xlabel('gy2 of underexposed gray image')

mag2=sqrt(gx.*gx+gy.*gy);
subplot(2,2,3),imshow(mat2gray(mag2)),xlabel('Magnitude2 of underexposed gray
image')

clear gx gy



%------------------------------- end of filter-------------------------------------%

%-----------------------define weighting factors---------------------------------%

denom=mag1+mag2;
weights1=mag1./denom;
weights2=mag2./denom;

 zeroindex=find(denom==0); % denom<0.1
weights1(zeroindex)=0.5;
weights2(zeroindex)=0.5;


figure(50)

subplot(2,2,1),imshow(mat2gray(weights1)),xlabel('(a) Weight1 of overexposed
image')
subplot(2,2,2),imshow(mat2gray(weights2)),xlabel('(b) Weight2 of underexposed
image')



MKS=round(v/10);  % juna v/4
```

```
%--------------------------end of weighting factors----------------------------%


%---------------------create Gaussian low pass filter--------------------%

LPF=fspecial('Gaussian',MKS,MKS/4);

weights1=imfilter(weights1,LPF, 'replicate');
% weights2=imfilter(weights2,LPF, 'replicate');
weights2=1-weights1;

% weights1=filter2(LPF,weights1,'same');
% %weights2=filter2(LPF,weights2,'same');
% weights2=1-weights1;



subplot(2,2,3),imshow(mat2gray(weights1)),xlabel('Weights for (a) after low pass
filter')
subplot(2,2,4),imshow(mat2gray(weights2)),xlabel('Weights for (b) after low pass
filter')

%------------------------end of LPF-------------------------------------%

%---------------------fused image---------------------------------------%

img1=double(img1); img2=double(img2);


HDR(:,:,1)=weights1.*img1(:,:,1)+weights2.*img2(:,:,1);
HDR(:,:,2)=weights1.*img1(:,:,2)+weights2.*img2(:,:,2);
HDR(:,:,3)=weights1.*img1(:,:,3)+weights2.*img2(:,:,3);


figure(60)


imshow(mat2gray(HDR)),xlabel('HDR image with Gradients before post processing ')

%------------------------end of fused image-----------------------------------%
```

```
% ---------------------juna 2nd post processing---------------------------------------%

HDR2 = imsharpen(mat2gray(HDR),'Radius',1.0,'Amount',1.5);

HDR3 = imadjust(HDR2,[0.1 1],[0 1],0.8);

HSV = rgb2hsv(HDR3); %%figure(80),imhist(HSV(:,:,3));
HSV(:, :, 2) = HSV(:, :, 2) * 1.2;
HSV(:,:,3)=HSV(:,:,3) * 1.1;

RGB = hsv2rgb(HSV);
figure(70);
imshow(RGB),xlabel(' final gradient based image after post processing');


%--------------------- end of post processing---------------------------%
```