Fundamental image processing techniques for high-definition displays

Ye, Wei

2017

Ye, W. (2017). Fundamental image processing techniques for high-definition displays.
Doctoral thesis, Nanyang Technological University, Singapore.

<http://hdl.handle.net/10356/71553>

<https://doi.org/10.32657/10356/71553>

**NANYANG
TECHNOLOGICAL
UNIVERSITY**

SINGAPORE

Fundamental Image Processing Techniques for High-definition Displays

YE WEI

School of Electrical and Electronic Engineering

A thesis submitted to the Nanyang Technological University
in partial fulfillment of the requirement for the degree of
Doctor of Philosophy

2017

Acknowledgments

This thesis would not be possible without the successive support from many people. I would like to take this opportunity to convey my sincerest gratitude to them.

First and foremost, I owe my everlasting appreciation to my supervisor, Prof. Kai-Kuang Ma, for his continuous guidance and support during the past four years. When I was unsure about where to start at the early stage of my Ph.D. study, it was his valuable suggestions that helped me find the research topics that I am passionate about. During our discussions, he always encouraged me to think deeply and investigate those fundamental problems that really matter. His enlightening ideas and insightful comments have given me numerous inspirations and always motivated me to go beyond what I have accomplished. He has also provided great instructions on developing my presentation and writing skills, which would be priceless fortune to my future life and career.

My gratitude also goes to my dear colleagues and friends, Dr. Wei Zhe, Dr. Wang Chen, Dr. Zeng Huanqiang, Dr. Yu Gang, Mr. Dang Kang, Mr. Jiang Yuning, and Mr. He Jiawei, for the helpful discussions, warm encouragements, and all the fun we have had together. Without them, my time at Nanyang Technological University would never be so enjoyable and memorable.

Last but not least, I would like to thank my beloved parents, who raised me with their unconditional love and supported me in all my pursuits. I hope that this work could make them proud of me.

Abstract

Recent development of high-resolution displays (e.g., 4K displays) has brought a plethora of new requirements and expectations on high-resolution and high-quality images. Owing to such trend of demand, many of the existing image processing techniques need to be advanced. The objective of this thesis is fully motivated by this vision, and four fundamental image processing areas are investigated in this thesis; i.e., *image demosaicing*, *image interpolation*, *image enhancement*, and *semantic image content filtering*. Among these techniques, image demosaicing is a critical processing step of the in-camera image processing pipeline that largely determines the quality of the camera’s output image. Image interpolation and image enhancement provide an economical way to improve the resolution and perceptual quality of images, respectively. Finally, semantic image content filtering is a generic image processing tool for extracting subjectively meaningful structures from an image, which is able to facilitate a variety of image processing tasks. Our contributions in these areas are respectively summarized as follows.

For image demosaicing, a novel demosaicing method, called the *iterative residual interpolation* (IRI), is proposed. The key success of our method lies in the developed IRI process that further exploits the recent residual interpolation (RI) based demosaicing strategy to restore the most important green (G) channel of the image. In essence, the proposed IRI process is an iterative refinement process of the estimated missing pixel values on the G channel, which is accomplished by letting all the three color channels (i.e., red, green, and blue) mutually guide each other through the guided filter at each itera-

tion. Extensive experiments conducted on two commonly-used test datasets have clearly demonstrated that the proposed IRI algorithm outperforms existing state-of-the-art demosaicing methods in most cases in terms of both objective and subjective performance evaluations.

For image interpolation, a *fast* contrast-guided interpolation (CGI) algorithm is proposed, which is a significant improvement over the original CGI method. In this approach, the time-consuming *iterative edge diffusion* (IED) scheme of the original CGI algorithm is replaced by a much more efficient *convolutional edge diffusion* (CED) scheme. The proposed CED scheme requires only a single convolution operation to accomplish any targeted amount of edge diffusion, which is controlled by the variance parameter of a Gaussian kernel. This leads to a significant speed-up over the original IED scheme as no iterative optimization is needed. Extensive simulations have shown that the resultant fast CGI algorithm costs only 1/4 of the run time as consumed by the original CGI, while delivering almost the same interpolation performance. This makes it highly appealing for those applications that require real-time performance and low power consumption.

For image enhancement, a novel extension of the classical *unsharp masking* (UM) approach, called the *blurriness-guided unsharp masking* (BUM), is developed. Our approach is motivated by the observation that enhancing a highly-blurred image region is undesirable, because it is unlikely to yield any improvement on image quality while often leading to visually unpleasant artifacts. Based on such observation, the proposed BUM incorporates the local blurriness information into the enhancement process to adapt the enhancement strength at different pixel locations. The key strategy and novel idea is that less enhancement or even no enhancement will be imposed for those highly-blurred regions. Experimental results have shown that the proposed approach is able to consistently deliver enhanced images with superior perceptual quality, compared with the conventional UM method and other existing adaptive UM methods.

For semantic image content filtering (SICF), a novel filtering algorithm, called the *edge-preserving scale-aware filter* (ESF), is developed. The proposed ESF is an extension of the recently-developed scale-aware *rolling guidance filter* (RGF) with significantly improved edge-preserving capability. The key novelty of our approach lies in the deployment of a more effective detail removal scheme for generating the initial guidance image for the follow-up structure recovery stage. For that, a modified *relative total variation* filter is developed and incorporated into the detail removal process. Compared with the original RGF’s detail removal scheme using only a Gaussian filter, the proposed scheme is able to smooth out more small-scale details on the input image while causing much less degradations to the large-scale main structures. Consequently, those main structures can be recovered more effectively in our proposed ESF, using the same structure recovery process as employed in the RGF. Extensive simulations conducted on a variety of test images have demonstrated that the proposed ESF is able to deliver superior SICF results on the aspects of small-scale detail removal and large-scale edge preservation, compared to other state-of-the-art filters for conducting the SICF.

Contents

Acknowledgments	i
Abstract	ii
List of Figures	xvi
List of Tables	xviii
List of Abbreviations	xix
1 Introduction	1
1.1 Motivation and Objective	1
1.2 Our Contributions	7
1.3 Organization	9
2 Color Image Demosaicing Using Iterative Residual Interpolation	11
2.1 Introduction	11
2.2 Residual Interpolation for Demosaicing	17
2.2.1 Stage 1: Component Image Estimation	18
2.2.2 Stage 2: Estimation Error Compensation	21
2.2.3 Residuals Versus Color-component Differences	22
2.3 Proposed Iterative Residual Interpolation Based Demosaicing Algorithm	24

2.3.1	Iterative Residual Interpolation Process	24
2.3.2	Local Window Size Selection for the Guided Filter	27
2.3.3	Stopping Criterion for the IRI Process	29
2.3.4	Final Reconstruction of the G Channel	31
2.3.5	Final Reconstructions of the R and B Channels	32
2.4	Experimental Results	33
2.4.1	Objective Performance Evaluations	33
2.4.2	Subjective Quality Assessment	37
2.4.3	Comparison of Computational Complexity	45
2.4.4	Effectiveness of Proposed Window Size Selection Strategy	46
2.5	Summary	47
3	Fast Contrast-guided Interpolation with Convolutional Edge Diffusion	48
3.1	Introduction	48
3.2	Contrast-guided Image Interpolation (CGI)	51
3.3	Proposed Fast CGI via Convolutional Edge Diffusion (CED)	53
3.3.1	Proposed CED Algorithm	54
3.3.2	Comparison of the Proposed CED and the CGI's IED	56
3.4	Experimental Results	58
3.5	Summary	64
4	Image Sharpness Enhancement via Blurriness-guided Unsharp Mask-	
	ing	67
4.1	Introduction	67
4.2	Literature Review	70
4.3	Proposed Pixel-wise Blurriness Estimation	74
4.3.1	Stage 1: Initial Blurriness Estimation	74

4.3.2	Stage 2: Blurriness Map Refinement	76
4.4	Proposed Blurriness-guided Unsharp Masking	81
4.4.1	Case 1: Computation of Λ for Edge-preserving Filter	82
4.4.2	Case 2: Computation of Λ for Non-edge-preserving Filter	84
4.5	Experimental Results	87
4.5.1	Parameter Settings	88
4.5.2	Subjective Evaluations	92
4.5.3	Comparison of Computational Complexity	96
4.6	Summary	97
5	Semantic Image Content Filtering via Edge-preserving Scale-aware Filter	98
5.1	Introduction	98
5.2	Rolling Guidance Filter	104
5.3	Proposed Edge-preserving Scale-aware Filter	108
5.3.1	Stage 1: Detail Removal	109
5.3.2	Stage 2: Structure Recovery	114
5.4	Experimental Results	118
5.4.1	Subjective Evaluations	118
5.4.2	Applications	120
5.4.3	Comparison of Computational Complexity	122
5.5	Summary	123
6	Conclusions and Future Works	127
6.1	Conclusions	127
6.2	Future Works	130

Publications	132
Bibliography	134

List of Figures

1.1	A typical image processing pipeline in digital still cameras (adapted from [2]).	2
2.1	The Bayer CFA pattern.	12
2.2	A comparison of a CD field and a residual field: (a) the full-color test image, “IMAX 5”; (b) the CD field, $R - G$; (c) the residual field, $R - \bar{R}$, where $\bar{R}$ is an estimated R-component image generated by the guided filter [42]. For visual comparison, the original values presented on (b) and (c) have been mapped into the range of $[0, 255]$ using a linear function $y = (x + 255)/2$	15
2.3	Block diagram of the RI-based reconstruction scheme for the R channel as described in the original RI method [54]. The same processing is used for the reconstruction of the B channel, likewise.	17
2.4	The proposed IRI process conducted along the <i>horizontal</i> direction only. For the <i>vertical</i> -direction case, the entire operations will be performed in the same manner, except along the vertical direction. Figures (a)-(f) denote the locations of the pixels that are involved in the processing points corresponding to the labels (a)-(f) as indicated in the block diagram, respectively. Note that the two inputs of the guided filter (i.e., p and d) are exchangeable such that the component images are guiding each other. In total, the filter will be executed four times per iteration.	25

2.5	The $MAD(k)$ values as the iteration number k increases for the <i>horizontal</i> IRI process, where $MAD(k)$ is the difference yielded between the refined G-component images obtained in iteration k and in iteration $k - 1$	30
2.6	Four representative test images from the Kodak and the IMAX datasets.	38
2.7	Visual comparison of the demoaicing results for a region in the “IMAX 5” image.	39
2.8	Visual comparison of the demoaicing results for a region in the “IMAX 17” image.	40
2.9	Visual comparison of the demoaicing results for a region in the “Kodak 8” image.	41
2.10	Visual comparison of the demoaicing results for a region in the “Kodak 19” image.	42
2.11	Visual comparison of the demoaicing results of the LSSC algorithm and the proposed IRI algorithm.	44
3.1	The two-stage interpolation framework as adopted by the CGI method: (a) interpolation of the <i>diagonal</i> pixels $I(2i, 2j)$; and (b) interpolation of the <i>off-diagonal</i> pixels $I(2i - 1, 2j)$ and $I(2i, 2j - 1)$. The black and hollow dots denote the available and missing pixels, respectively.	53
3.2	A comparison of the diffused DV fields (<i>continuous-valued</i>) and the final CDMs (<i>binary-valued</i>), generated by the CGI’s IED and our proposed CED, respectively, using the test image “Lena”: (a) ; (b) u_{0° by CED ; (c) differences between (a) and (b); (d) CDM_{90° by IED; (e) CDM_{90° by CED; and (f) differences between (d) and (e).	57
3.3	The eight test images used in our experiments.	58

3.4	Some PSNR results of our proposed <i>fast</i> CGI algorithm averaged on 8 commonly-used test images using different settings of the filter length N and the standard deviation σ of the Gaussian kernel.	59
3.5	Visual comparison of the image interpolation results obtained from the test image “Wheel”.	61
3.6	Visual comparison of the image interpolation results obtained from the test image “Motorbike”.	61
3.7	Visual comparison of the image interpolation results obtained from the test image “Plane”.	63
3.8	Visual comparison of the image interpolation results obtained from the test image “Butterfly” using a scaling factor of 4.	63
4.1	Enhancement results of image regions at different blur levels by using the conventional UM approach. One can see that enhancing the highly-blurred patch (blue-framed) on the background leads to distinct amplified granular noise, while enhancing the less-blurred patch (red-framed) gives more visually desirable result. However, these two patches have extremely similar local contrast measurement (i.e., the average gradient magnitudes of both patches are approximately 5 intensity levels); this means that it is difficult to distinguish them by the local contrast information.	69

4.2	Results of various blurriness map refinement schemes: (a) original image; (b) the initial blurriness map $\widehat{M}$ generated by using the JNBE algorithm [86]; (c) the confidence weightage map computed according to (4.3); (d) the refined map M by using the bilateral filter [41] with $\sigma_s = 5$ and $\sigma_r = 0.1$; (e) the refined map M obtained by using the WLS-based approach without the confidence weights; (f) the refined map M obtained by using the proposed WLS-based approach with the adaptive confidence weightage map as shown in (c).	77
4.3	The processing pipeline of the proposed blurriness-guided adaptive unsharp masking, conducted on the luminance component L , recorded in the CIELab color space.	80
4.4	An illustration of the the proposed mapping functions for generating the adaptive scaling factor matrix Λ : (a) the blurriness-guided mapping function as defined in (4.10) for Case 1, when an edge-preserving filter is used; (b) the blurriness- and contrast-guided mapping function as defined in (4.12) for Case 2, when non-edge-preserving filter is used. Note that $\rho = 3$, $\eta = 1$ are used for generating the above plots, thus $\Lambda_p \in (1, 4]$ in both cases.	84
4.5	Demonstrations of adaptive scaling factor matrix computation for Case 1 and Case 2: (a) the input image; (b)-(c) the $ T $ obtained via the Bilateral filter (Case 1) and the Gaussian filter (Case 2), respectively; (d)-(e) the Λ computed for Case 1 and Case 2, respectively; (f) an image patch cropped from (a); (g)-(h) enhanced patches in Case 1 and Case 2, respectively; (i) enhanced patch in Case 2 using the mapping function (4.10) (i.e., without incorporating $ T $).	87

4.6	Visual comparison of the enhancement results on the image “old man”; (a) original image; (b) UM with Gaussian filter; (c) GUM with Gaussian filter [39]; (d) AUM with Gaussian filter [34]; (e) CUM [37]; (f) Proposed BUM with Gaussian filter ($\eta = 1$).	90
4.7	Visual comparison of the enhancement results on the image “parrot”; (a) original image; (b) UM with Gaussian filter; (c) GUM with Gaussian filter [39]; (d) AUM with Gaussian filter [34]; (e) CUM [37]; (f) proposed BUM with Gaussian filter ($\eta = 1$).	91
4.8	Visual comparison of the enhancement results on the test image “burger”; (a) original image; (b) UM with bilateral filter; (c) GUM with bilateral filter [39]; (d) modified AUM with bilateral filter [34]; (e) CUM [37]; (f) proposed BUM with bilateral filter ($\eta = 1$).	94
4.9	Visual comparison of the enhancement results on the test image “bear”; (a) original image; (b) UM with bilateral filter; (c) GUM with bilateral filter [39]; (d) modified AUM with bilateral filter [34]; (e) CUM [37]; (f) proposed BUM with bilateral filter ($\eta = 1$).	95
5.1	Top row (from left to right): the input image, the filtered image (or ex- tracted important image structures) by using the proposed SICF algorithm (to be detailed in Section 5.3), and the filtered image by using a low-pass Gaussian filter. Bottom row: the amplitude spectrum of the Fourier trans- form of the respective images on the top row. Note that the zero-frequency component is situated at the center of the spectrum.	99
5.2	Filtering results on (a) the “batman” image, obtained by using (b) the Gaussian filter, (c) the bilateral filter [41], (d) the rolling guidance fil- ter [52], and (e) the proposed edge-preserving scale-aware filter, respectively.	103

5.3	The flowchart of the <i>rolling guidance filter</i> (RGF) [52]. Note that R^0 is a Gaussian-blurred version of the input image I , where the superscript denotes the iteration index.	105
5.4	Illustration of the gradient directions in different types of patches: (a) input image; (b)-(c) gradient direction maps of a patch containing large-scale structure (red-framed) and another containing only small-scale textures (green-framed), respectively.	110
5.5	Different Gaussian weighting kernels as employed in our approach: (a) the isotropic Gaussian kernel with standard deviation σ_v ; (b)-(e) the four anisotropic Gaussian kernels along the 0° , 90° , 45° , and 135° directions, respectively. For each anisotropic kernel, the standard deviations along the major and minor axes are σ_v and $\sigma_v/2$, respectively.	111
5.6	Filtering results of different detail-smoothing filters: (a) input image; (b)-(d) smoothed images by using the (b) Gaussian filter, (c) the original RTV filter [49], and (d) the proposed AWRTV filter.	113
5.7	A demonstration of the input and intermediate images involved in the proposed ESF: (a) the input image I ; (b) the filtered image S obtained by using the proposed AWRTV; (c) the further smoothed version of S , i.e., $G * S$; (d)-(f) filtered images after the 1st, 2nd, and 10th iteration of the structure recovery process. One can see that the main image structure becomes well-recovered after only 2 iterations.	115
5.8	The mean absolute difference between the filtered images R^k and R^{k-1} (denoted by $MAD(k)$) as measured on the “dog” image in Fig. 5.7 (a) by applying the proposed ESF and the RGF, respectively.	116

5.9	A comparison of the structure recovery results R^K using different initial guidance image R^0 . Top row: the input image I . Middle row: the R^0 obtained by applying (b) the Gaussian filter (with $\sigma = 6$) as practiced in the RGF [52], (c) the developed AWRTV filter only, (d) the developed AWRTV followed by a Gaussian filter (with $\sigma = 2$). Bottom row: the R^K obtained by using the respective R^0 from the above row as the initial guidance image. The parameters for the structure recovery process are set to $\sigma_r = 0.05$ and $\sigma_s = 6$ for all three cases.	117
5.10	Detail removal results of different detail-smoothing filters: (a) original image; (b) BF ($\sigma_s = 4$, $\sigma_r = 0.5$); (c) WLS ($\lambda = 4$, $\alpha = 2$); (d) TF ($\sigma = 0.04$, $\sigma_s = 4$, $\sigma_r = 0.1$, with 2 iterations); (e) SD ($\lambda = 500$, $\mu = 50$, $\nu = 300$, with 5 iterations); (f) RTV ($\lambda = 0.01$, $\sigma = 4$, with 4 iterations); (g) RGF ($\sigma_s = 4$, $\sigma_r = 0.1$); (h) proposed ESF ($\lambda = 0.02$, $\sigma_v = 4$, $\sigma_s = 8$, $\sigma_r = 0.05$).	119
5.11	Detail removal results of different detail-smoothing filters: (a) original image; (b) BF ($\sigma_s = 4$, $\sigma_r = 0.5$); (c) WLS ($\lambda = 4$, $\alpha = 2$); (d) TF ($\sigma = 0.04$, $\sigma_s = 4$, $\sigma_r = 0.1$, with 2 iterations); (e) SD ($\lambda = 500$, $\mu = 200$, $\nu = 200$, with 5 iterations); (f) RTV ($\lambda = 0.02$, $\sigma = 4$, with 4 iterations); (g) RGF ($\sigma_s = 4$, $\sigma_r = 0.1$); (h) proposed ESF ($\lambda = 0.02$, $\sigma_v = 4$, $\sigma_s = 8$, $\sigma_r = 0.05$).	124
5.12	Image enhancement results using different layer-decomposition filters: (a) the input color image (bottom) and its luminance component (top) in the CIELab color space; (b)-(d) the extracted base layer images (top) and the resultant enhanced images (bottom) obtained by using (b) the BF ($\sigma_s = 5$, $\sigma_r = 0.2$), (c) the RGF ($\sigma_s = 5$, $\sigma_r = 0.1$), and (d) the proposed ESF ($\lambda = 0.04$, $\sigma_v = 8$, $\sigma_s = 5$, $\sigma_r = 0.1$), respectively.	125

5.13	Texture replacement results by using the proposed ESF: (a) the input image; (b) the extracted base layer by applying the proposed ESF ($\lambda = 0.03$, $\sigma_v = 4$, $\sigma_s = 4$, $\sigma_r = 0.08$); (c) the synthetic image by adding another texture image to (b).	125
5.14	Image de-halftoning results using different filters: (a) the input color image; (b)-(d) the reconstructed continuous-tone image obtained by using (b) the BF ($\sigma_s = 4$, $\sigma_r = 0.2$), (c) the RGF ($\sigma_s = 4$, $\sigma_r = 0.08$), and (d) the proposed ESF ($\lambda = 0.03$, $\sigma_v = 4$, $\sigma_s = 4$, $\sigma_r = 0.08$), respectively. . .	126
5.15	Edge detection results using a canny edge detector. Top row: the input image and its smoothed versions by using the BF ($\sigma_s = 10$, $\sigma_r = 0.2$), the RGF ($\sigma_s = 10$, $\sigma_r = 0.1$), and the proposed ESF ($\lambda = 0.06$, $\sigma_v = 5$, $\sigma_s = 10$, $\sigma_r = 0.1$), respectively. Bottom row: the detected edge maps for the respective images in the top row.	126

List of Tables

2.1	Average smoothness measured over the residual fields and the CD fields, respectively. These measurements are conducted for the Kodak image set and the IMAX image set, separately.	23
2.2	Average PSNR and CPSNR results (in dB) on the Kodak and the IMAX datasets. The highest average PSNR result in each column is highlighted in boldface.	35
2.3	Average SSIM results on the Kodak and the IMAX datasets. The highest average SSIM result in each column is highlighted in boldface.	36
2.4	Average running time (in seconds) of different methods for demosaicing one image on the Kodak and the IMAX datasets. The spatial resolution of test images in each dataset is indicated below the corresponding dataset name.	45
2.5	Average PSNR and SSIM results (for the G channel) obtained by using different window size configurations	46
3.1	Average PSNR (in dB) results. The highest PSNR for each test image are highlighted in boldface.	60
3.2	Average SSIM results. The highest SSIM for each test image are highlighted in boldface.	60
3.3	Total run time (in seconds) for interpolating 8 test images	64

4.1	Average running time (in seconds) of different methods for enhancing an 800 × 533 image	97
5.1	Average running time (in seconds) of different methods for filtering a 500 × 375 image	122

List of Abbreviations

BUM	Blurriness-guided unsharp masking
CD	Color-component difference
CDI	Color-component difference interpolation
CDM	Contrast-guided decision map
CED	Convolutional edge diffusion
CFA	Color filter array
CGI	Contrast-guided interpolation
DV	Directional variation
EGI	Edge-guided interpolation
ESF	Edge-preserving scale-aware filter
HD	High-definition
HR	High-resolution
IED	Iterative edge diffusion
IRI	Iterative residual interpolation
LR	Low-resolution
LSI	Linear shift-invariant
MAD	Mean absolute difference
PSNR	Peak signal to noise ratio
RGF	Rolling guidance filter
RI	Residual interpolation
RTV	Relative total variation
SICF	Semantic image content filtering
SSIM	Structural similarity
UM	Unsharp masking

Chapter 1

Introduction

1.1 Motivation and Objective

With the development of display technologies, the high-definition (HD) displays have become more and more prevalent nowadays in the consumer electronics market. Besides the current mainstream “Full HD” displays that have a resolution of 1920×1080 , the next-generation “Ultra HD” displays, as coined by the Consumer Electronics Association in 2012, have also been emerging rapidly in recent years, with resolutions reaching up to 3840×2160 (“4K Ultra HD”) or even to 7680×4320 (“8K Ultra HD”) [1]. The high resolutions of HD displays allow them to present much more fine details of the imagery, thus creating much superior visual sensations to that provided by other displays with lower resolutions. Obviously, any flaws of the imagery will also be more easily noticeable on a HD display. Therefore, for the consumers to fully appreciate and enjoy the viewing experience brought by the HD displays, imagery with higher resolution as well as higher quality in various aspects, such as resolution, sharpness, contrast, dynamic range, noise level, and so on, are clearly needed to go along with the HD display technology.

To fulfill such demand, two typical approaches have been commonly practiced: 1) developing more advanced digital cameras, and 2) conducting resolution and/or quality improvement to existing imagery. In either approach, various fundamental image

Figure 1.1: A typical image processing pipeline in digital still cameras (adapted from [2]).

processing techniques need to be developed and incorporated. To be more specific, the first approach requires not only the upgrade of hardware (such as camera lenses and image sensors), but also the development of software (i.e., the built-in image processing algorithms). A typical image processing pipeline of digital still cameras is depicted in Fig. 1.1 [2], from which one can see that a series of image processing operations are conducted between the capture of incoming light and the final image output¹. On the other hand, the second approach is purely a software approach based on image processing techniques, such as image interpolation and image enhancement. Both approaches have their own strength and weakness. Despite that the first approach ‘starts from the scratch’ to meet the emerging demand on high-resolution and high-quality images, it comes at a fairly high cost because of the required hardware upgrade. Moreover, transmitting and storing the acquired high-resolution imagery (especially those with “Ultra HD” resolutions) are still difficult nowadays due to their extraordinarily large file size. In contrast, the second approach offers a much more economical and practical way to

¹Note that the involved image processing operations in today’s digital cameras may be slightly different from what shown in Fig. 1.1, and they could also be different from manufacturer to manufacturer. Refer to [2] for more detailed introduction on the in-camera image processing pipeline.

generate the desired imagery without demanding costly upgrade on camera hardware, network, or storage devices. Nevertheless, generally speaking, its yielded imagery quality is not as good as that of the first approach.

Owing to the continuous evolution of display devices and the resultant higher requirements on image resolution and quality, there is no doubt that the involved image processing techniques in the above-mentioned two approaches need to be further advanced. Our research is fully motivated by this vision, and four fundamental image processing techniques are investigated in this thesis, i.e., *image demosaicing*, *image interpolation*, *image enhancement*, and *semantic image content filtering*. In what follows, a brief introduction of the research problems in each field is provided, while our contributions will be highlighted in Section 1.2.

Image Demosaicing In order to reduce the production cost and physical size of the camera, almost all consumer digital cameras use only one image sensor, which is overlaid by a color filter array (CFA) as shown in Fig. 1.1, to capture only *one* of the three RGB color-component values at each pixel location, and such captured image is commonly known as a *mosaiced* image. Image *demosaicing* is the reverse process of mosaicing, which estimates the two missing color-component values at each pixel location to restore a full-color RGB image from the mosaiced image. If the demosaicing process is not properly carried out, various color artifacts (e.g., aliasing, false color, zipper artifacts, etc.) could be imposed in the restored image [3]. Therefore, demosaicing is widely considered as one of the most important processing steps of the in-camera image processing pipeline. Certainly, it is critical for acquiring the demanded high-quality images for the HD displays. In the literature, a popular class of demosaicing methods is known as the *color-component difference interpolation* (CDI) (e.g., [4]–[19]). Its development is based on the key observation that the correlations among different color channels (i.e., R, G,

and B) are usually strong and thus their corresponding *color-component difference* (CD) fields (e.g., R–G) are usually very smooth. However, it has been observed that the demosaicing performance of this class of approaches starts to degrade significantly when the correlations among three channels become weak. This typically happens when the image contains sharp color transitions, in which case distinct edges will be incurred in the CD fields and cause the CDI yielding large interpolation errors. Since sharp color transitions are very common in natural images (especially in those images with vivid colors), it is highly desirable to investigate color demosaicing algorithms that are effective in such cases (i.e., robust to the amount of channel correlations).

Image Interpolation The objective of this processing task is to generate a higher-resolution image from a corresponding lower-resolution version. It is another particularly important image processing technique for the HD displays, as it provides a much more economical way to acquire high-resolution imagery compared with the approach via camera hardware upgrade as mentioned above. Although image interpolation has been an active area of research for many years, it remains a very challenging problem. Due to the high resolution and large size of HD displays, this task becomes even more difficult because various visually unpleasant artifacts caused by interpolation will be more easily noticeable on the screen. Conventional interpolation algorithms, such as bilinear [20], bicubic spline [21], and bicubic convolution [22], are computationally efficient and easy to implement, but they often suffer from certain forms of artifacts, such as blurring, aliasing, ringing, jaggy edges, etc. More sophisticated *edge-guided interpolation* (EGI) methods (e.g., [23]–[31]) are able to significantly improve the interpolated image quality. The key idea of this approach is to identify edges first and then conduct *directional* interpolation for the missing (i.e., to-be-interpolated) pixels situated on an edge. More recently, a novel interpolation method, called the *contrast-guided interpolation* (CGI) [32], has been

developed and shown state-of-the-art interpolation performance on image quality. The key novelty of the CGI method lies in the further re-classification of some edge-nearby non-edge pixels as ‘edge’ ones to conduct directional interpolation. Thus, the CGI can be viewed as a powerful extension of the EGI approach. Despite the attractive performance of the EGI and CGI methods, it has been noticed that they all incur fairly high computational complexity and algorithmic complexity, which greatly hinders their use in many practical applications (e.g., the built-in image upscaling module of the HD displays) that require real-time performance and low power consumption. Therefore, one major objective of this thesis is to develop an interpolation algorithm that is not only capable of delivering high-quality interpolated images, but also computationally efficient and algorithmically simple.

Image Enhancement Unlike the above-mentioned image demosaicing and interpolation techniques, image enhancement is not a process of restoring the missing information of the image. Instead, it is an operation of altering the information that is already captured or recorded on the image. The term “image enhancement” covers a broad range of techniques, which are applied to modify one or more important attributes of the image, such as sharpness, contrast, and so on. Since image enhancement techniques are able to effectively improve the perceptual quality of images, they are often built in the image processing module of cameras and displays. In this thesis, we focus on the enhancement of image sharpness, for which the most well-known technique is called the *unsharp masking* (UM) [33]. In the UM algorithm, the sharpness enhancement is accomplished by amplifying the extracted high-frequency components of the input image by a constant scaling factor. Although the UM algorithm is simple and works well in many cases, it has been found that it often introduces two types of visually unpleasant artifacts in the enhanced image, i.e., amplified noise and overshoot artifacts, due to the fixed en-

hancement strength imposed for all pixels without considering the image content [34]. Various adaptive UM algorithms (e.g., [34]–[39]) have been attempted to overcome the shortcomings of the UM algorithm by utilizing the local *contrast* information to adapt the enhancement strength of different pixels. However, it has been observed that these contrast-based methods could still produce the above-mentioned artifacts, and some of them sometimes will even lead to the under-enhancement issue (i.e., insufficient amount of enhancement). This is mainly because the local contrast is unable to capture the high-level semantic clues that are needed to properly determine how much enhancement should be imposed at a each pixel. Therefore, one of the main objectives of this thesis is to extract and utilize a more effective image attribute to guide the UM process and avoid all the above-mentioned issues.

Semantic Image Content Filtering In many practical image processing applications, such as noise suppression, edge detection, segmentation, and so on, it is often highly desirable to remove the subjectively insignificant contents from the input image, while only preserving the semantically meaningful ones. In this thesis, this task is coined as *semantic image content filtering* (SICF). Unfortunately, the conventional linear shift-invariant (LSI) filters, which process an image mainly from a frequency point of view, are unable to effectively accomplish such task. This is because the semantically important and unimportant image contents cannot be discriminated simply based on the frequencies. Over the last decades, the *edge-aware* or *edge-preserving* smoothing filters (e.g., [40]–[47]) have emerged and widely exploited for conducting SICF. However, they often fail to suppress those high-contrast details or preserve those low-contrast but important structures, because they determine the semantic importance of image contents based on the *contrast*. Recently, it has been pointed out in [48] that the semantically insignificant (or unwanted) image details should be identified based on the their *scale*

rather than the contrast. Inspired by this insight, several pioneer works (e.g., [48]–[53]) have been proposed that incorporate the scale of image structures into the filtering process. In view of the practical usefulness of the SICF and the promising filtering results of the scale-aware filters, one major objective of this thesis is to develop a more effective scale-aware filter to conduct the desired SICF.

1.2 Our Contributions

In this thesis, four contributions have been made in the above-mentioned four research areas, respectively, which are summarized as follows.

For image demosaicing, a novel demosaicing method, called the *iterative residual interpolation* (IRI) is proposed. Our method is a novel extension of the recently-developed *residual interpolation* (RI) [54] based demosaicing strategy that has demonstrated superior performance over the conventional color-component difference interpolation (CDI) approach. The key novelty of the proposed IRI process lies in the developed IRI process that further exploits the RI-based strategy to the most important G channel of the image. In the IRI process, an iterative refinement of the estimated missing pixel values of the G channel is accomplished by letting all the three channels guiding each other via the guided filter. The proposed IRI algorithm is able to restore a highly accurate G channel, with which the reconstructions of the R and B channels using the original non-iterative RI process are greatly benefited as well. Extensive simulations conducted on two commonly-used test datasets for demosaicing algorithms have demonstrated that our algorithm has achieved the best performance in most cases, compared to the existing state-of-the-art demosaicing methods on both objective and subjective performance evaluations.

For image interpolation, a *fast* contrast-guided image interpolation algorithm is pro-

posed, which is a significant improvement over the state-of-the-art contrast-guided interpolation (CGI) method [32]. In this approach, the time-consuming *iterative edge diffusion* (IED) scheme of the original CGI algorithm is replaced by a much more efficient *convolutional edge diffusion* (CED) scheme. The proposed CED scheme requires only a single convolution operation to accomplish any targeted amount of edge diffusion, which is controlled by the variance parameter of a Gaussian kernel. This leads to a significant speed-up over the original IED scheme as no iterative optimization is needed. Extensive simulations have shown that the resultant fast CGI algorithm costs only 1/4 of the run time as consumed by the original CGI, while delivering almost the same interpolation performance. This makes it highly appealing for those applications that require real-time performance and low power consumption.

For image enhancement, a novel content-adaptive sharpness enhancement approach, called the *blurriness-guided unsharp masking* (BUM), is developed. Our approach is motivated by the observation that enhancing a highly-blurred image region is undesirable, because it is unlikely to yield any quality improvement while often leading to visually unpleasant artifacts. Based on such observation, a two-stage pixel-wise blurriness estimation approach is developed to generate a highly-accurate pixel-wise blurriness map for each input image, followed by conducting a mapping process to translate each estimated blurriness value to a corresponding scaling factor. The key strategy of our approach is that those pixels in highly-blurred regions will receive smaller scaling factors, such that they are less enhanced or not enhanced at all. Extensive simulations on various test images have demonstrated that the proposed method is able to deliver superior perceptual quality of the enhanced image compared with the conventional UM and other existing adaptive UM methods.

For semantic image content filtering, a novel *edge-preserving scale-aware filter* (ESF) is developed. The proposed ESF is inspired by the recently-developed scale-aware *rolling*

guidance filter [52], but with a significant improvement on the aspect on edge preserving. The key novelty of our approach lies in the proposed detail removal scheme for generating a much improved initial guidance image for the follow-up structure recovery stage to start with. Compared with the original RGF’s detail removal scheme using only a Gaussian filter, the proposed scheme is able to smooth out more small-scale details on the input image while causing much less degradations to the large-scale main structures. Consequently, those main structures can be recovered more effectively in our proposed ESF via the same structure recovery process as employed in the RGF. Extensive simulations conducted on a variety of test images have clearly demonstrated that the proposed ESF outperforms other state-of-the-art filters for conducting the SICF, on the aspects of small-scale details removal and large-scale structures preservation. This makes our approach highly desirable in many practical image processing tasks that can be greatly benefited by an effective SICF process.

1.3 Organization

The rest of this thesis is organized as follows. In Chapter 2, a detailed introduction of the RI-based demosaicing strategy as well as the proposed demosaicing algorithm based on the *iterative residual interpolation* (IRI) is presented. Experimental results obtained from extensive simulations are provided to show that the proposed IRI algorithm clearly outperforms existing state-of-the-art demosaicing methods, in terms of objective evaluation on PSNR as well as subjective assessment on perceptual quality.

In Chapter 3, the proposed fast contrast-guided interpolation algorithm using the *convolutional edge diffusion* (CED) is elaborated at length. Performance evaluations of our fast CGI algorithm in terms of both interpolated-image quality and computational efficiency are provided, which clearly demonstrate that the proposed fast CGI algorithm

is able to deliver state-of-the-art interpolation performance while having significant advantage in terms of algorithmic simplicity and computational complexity.

In Chapter 4, the proposed *blurriness-guided unsharp masking* (BUM) method is described in detail. A novel two-stage pixel-wise blurriness estimation method is developed and introduced, which is able to generate highly accurate blurriness map of the input image for guiding the follow-up enhancement process. Subjective evaluations conducted on various testing images are provided to demonstrate the superior enhanced-image quality compared with other existing UM methods.

In Chapter 5, the proposed *edge-preserving scale-aware filter* (ESF) for conducting semantic image content filtering (SICF) is introduced. Experimental results obtained from extensive simulations are provided to show the superiority of the proposed ESF over other state-of-the-art edge-aware or scale-aware filters on conducting the SICF. Furthermore, several application examples are demonstrated to show the effectiveness and usefulness of the proposed ESF algorithm.

Finally, Chapter 6 summarizes this thesis and discusses several promising future research directions.

Chapter 2

Color Image Demosaicing Using Iterative Residual Interpolation

2.1 Introduction

To produce a full-color RGB image, a digital camera need to record three color-component values (i.e., R, G, and B) at each pixel location. One approach to accomplish this is to use beam-splitters along the optical path to project the input light onto three separate image sensors, such as charge-coupled devices (CCD) sensors or complementary metal-oxide-semiconductor (CMOS) sensors. Such cameras are known as the 3CCD and 3CMOS cameras, respectively. However, this approach is costly as it requires three image sensors and a large bandwidth to transmit the acquired full-color image. Due to the consideration of cost, most digital cameras in consumer electronics exploit a single image sensor, which is overlaid with a *color filter array* (CFA), to acquire color images. As a result, only one of the three primary color components (i.e., R, G, or B) is recorded at each pixel location by the sensor, and such captured image is commonly termed as a *mosaiced* image. The arrangement of the recorded color components on the mosaiced image is determined by the selected CFA, and the most widely-used CFA pattern is the so-called *Bayer pattern* [55], as shown in Fig. 2.1. *Demosaicing* is the reverse process of mosaicing. It estimates the two missing color components at each pixel location to restore a *full-color*

Figure 2.1: The Bayer CFA pattern.

image from the mosaiced image. If the demosaicing process is not properly carried out, severe color artifacts will be incurred on the restored full-color image.

Through the *Bayer pattern* as shown in Fig. 2.1, the number of initially-acquired values on the G channel is exactly twice that of the R channel and of the B channel, respectively. Therefore, the G channel is commonly considered as the most important one to be demosaiced first. The reconstructed full-resolution G channel is then exploited to facilitate the reconstruction of the remaining two channels. Apparently, the quality of the reconstructed G channel is fairly critical to the overall demosaicing performance, since any reconstruction error introduced in it will be inevitably propagated to the R channel and the B channel in the subsequent processing steps.

Among the existing demosaicing methods (e.g., [4]–[19]), one popular strategy is the so-called *color-component difference interpolation* (CDI)¹. However, the demosaicing performance yielded by this approach is sensitive to the amount of *spectral correlations* existing among the three color channels—i.e., the higher the correlations presented, the better the performance to be delivered. The CDI-based demosaicing approach begins with duplicating the value of the known color component (say, G) for the missing component (say, R) at the same pixel location as its initially-estimated ballpark value, followed by adding another estimated amount of adjustment to obtain the final restored value.

¹This term is often referred to as *color difference interpolation* in the literature (e.g., [54], [56]). However, in our view, the term *color difference* is ambiguous and could lead to mis-interpretation. Therefore, a more accurate term, called *color-component difference*, is suggested and used throughout this thesis.

It turns out that the estimation of adjustment amount can be effectively obtained by conducting interpolation on the *color-component difference* (CD) field (i.e., $R - G$, in this case), since the CD field tends to be fairly smooth over each local region or patch, especially when the correlation between the color channels is strong enough. Therefore, the CDI-based approach is able to restore the missing color-component values more accurately than simply performing interpolation on individual color channel of the mosaiced image.

Besides the CDI-based methods, many other demosaicing approaches have also been proposed in the past, including the wavelet-domain approaches (e.g., [57]), the frequency-domain approaches (e.g., [58]–[60]), and the sparse-representation-based approaches (e.g., [61]–[63]). Comprehensive surveys of demosaicing algorithms can be found in [64] and [3], for example. Some representative demosaicing algorithms that are most related to our work are succinctly highlighted as follows.

In [4], a simple directional filter is exploited to restore each missing color component through a 1-D linear interpolation on the CD field. Due to its attractive performance and algorithmic simplicity, this filter and its variants have been adopted in many subsequently developed algorithms (e.g., [5]–[17]). In order to better preserve edges and suppress color artifacts, various decision schemes have been proposed and incorporated with this filter for selecting the optimal direction to conduct interpolation (i.e., a *hard*-decision approach) (e.g., [4]–[10]), or adaptively combining the interpolation results individually obtained at different directions (i.e., a *soft*-decision approach) (e.g., [11]–[17]). A common practice for making such decision is to take the gradients computed on the CD field into account, since the smoother the CD field is, the more accurate the adjustment amount can be estimated for refining the restored value of each missing component. For example, Menon *et al.* [8] proposed a hard-decision scheme where the optimal interpolation direction is selected according to which direction has yielded the smallest CD gradient

value. Chung *et al.* [9] later developed a similar scheme, except using a hybrid gradient measurement that combines the gradients computed on the CD fields and the ones obtained from the mosaiced image itself. As to the soft-decision schemes, Pekkucuksen *et al.* [13], [16] computed the weight of each directional interpolation result as the inverse of the square of CD gradient, that is, the larger the gradient value resulted, the less the weight will be imposed.

In addition to the above-mentioned methods, some CDI-based approaches exploited an iterative refinement strategy to improve the initial estimation of each channel. For example, Li [18] proposed a *successive approximation* (SA) scheme to refine the R channel and the B channel individually based on an interpolated G channel, followed by conducting the refinement of the G channel with the help of the refined R and B channels. This process will be repeated until a stopping criterion is satisfied. Su *et al.* [19] further improved this technique by incorporating an edge-based weighted interpolation to obtain a more accurate initial estimation of the G channel.

Despite the strong spectral correlations existing among the three channels, it has been observed that when two regions adjacent to each other have a sharp *color transition*, an abrupt change will be incurred between them on the CD field. Consequently, the interpolation conducted in the neighborhood of the boundary tends to yield large estimation errors and lead to more distinct color artifacts on the demosaiced image. To alleviate this problem, several techniques have been developed. For example, the non-local redundancy, which is widely used for image denoising [65], is utilized for enhancing the quality of the reconstructed image in [14], [66], and [67]. In [68], a hybrid demosaicing approach is proposed that adaptively combines the reconstruction results given by a CDI-based algorithm with that by another image interpolation method.

Recently, a new demosaicing approach is proposed, called the *residual interpolation* (RI) [54]. Instead of conducting interpolation on the CD fields, the RI-based algorithm

Figure 2.2: A comparison of a CD field and a residual field: (a) the full-color test image, “IMAX 5”; (b) the CD field, $R - G$; (c) the residual field, $R - \bar{R}$, where $\bar{R}$ is an estimated R-component image generated by the guided filter [42]. For visual comparison, the original values presented on (b) and (c) have been mapped into the range of $[0, 255]$ using a linear function $y = (x + 255)/2$.

performs interpolation on the so-called *residual fields*, where a *residual* is defined as the difference yielded between the initially-acquired color-component value (i.e., the ground truth) and its estimated value obtained by applying the *guided filter* [42]. As implicated by the name of the guided filter, the estimation of the color-component values in one channel is conducted under the *guidance* of another channel via local linear regression. It turns out that such estimated values are usually more accurate than that of the CDI-based approach, in which the estimated values of the missing components at each pixel location are obtained by simply making a duplication from the only known color-component value. Consequently, the resulted residuals are often very close to zero and thus the residual field is smoother than the CD field. This is especially the case around edges and textures, where abrupt variations on one or more color channels tend to occur and thus lead to drastically changing CD values. A comparison between the residual field and CD field is demonstrated in Fig. 2.2, from which one can see that the object boundaries are distinctly yielded on the CD field, while becoming hardly visible on the residual field. However, the algorithm as described in [54] and its improved version as presented in [56] fail to fully

exploit the RI strategy for all the three channels. In fact, only the R and B channels are restored using the RI process, while the reconstruction of the most important G channel mainly relies on a CDI-based approach, called the *gradient based threshold free* demosaicing (GBTf) [13]. Furthermore, these approaches are *non-iterative* and thus lack of an opportunity to conduct refinement for the initially-restored values.

In this chapter, a novel RI-based demosaicing algorithm for restoring a much improved full-resolution G channel is proposed, called the *iterative residual interpolation* (IRI), which applies the RI process to all the three channels in a systematic manner and iteratively for refining their estimated values. During the IRI process, all the three channels will be mutually guiding each other until the iteration stopping criterion is met. Furthermore, the IRI process will be individually conducted twice, one along the *horizontal* direction and the other along the *vertical* direction. The resulted two restored G channels will then be linearly combined using a soft-decision approach to arrive at the final restored G channel, which is to be used to guide the reconstruction of the other two channels through the existing non-iterative RI method, respectively. Extensive simulation experiments conducted on the commonly-used Kodak and IMAX datasets have clearly shown that the proposed demosaicing algorithm has achieved the best performance in most cases, compared to existing state-of-the-art demosaicing methods in terms of both objective and subjective evaluations.

The rest of this chapter is organized as follows. Section 2.2 provides a succinct review of the RI process for demosaicing and clear justifications on the superiority of the smoothness of the residual fields, compared with that of the CD fields. Section 2.3 describes the proposed IRI-based demosaicing algorithm in detail. Section 2.4 conducts performance evaluation of several comparable methods. Section 2.5 summarizes this chapter.

Figure 2.3: Block diagram of the RI-based reconstruction scheme for the R channel as described in the original RI method [54]. The same processing is used for the reconstruction of the B channel, likewise.

2.2 Residual Interpolation for Demosaicing

Residual interpolation (RI) was first introduced in [54] for restoring those missing color-component values on a mosaiced component image p under the guidance of another component image d , as shown in Fig. 2.3. It consists of two stages: 1) component-image *estimation* using the *guided filter* [42], and 2) estimation-error *compensation* via the residual interpolation. In Fig. 2.3, only the reconstruction process of the R channel (or R-component image) is illustrated. For the reconstruction of the B channel (or B-component image), the same approach will be applied, likewise. That is, in Fig. 2.3, $p = B_m$, $\bar{p} = \bar{B}$, $\Delta p = \Delta B$, $\overline{\Delta p} = \overline{\Delta B}$, and $q = \hat{B}$.

Why the RI is a promising approach for conducting image demosaicing? Although the RI process follows a similar processing pipeline as the way performed in the CDI approach, however it is the guided filter [42] that delivers the superiority on the estimation

of the component image under restoration as conducted in the Stage 1 of Fig. 2.3. On the counterpart of this stage, the CDI simply duplicates the pixel values on the G-component image for those unknown (i.e., missing) R-component values at the same pixel locations, respectively. As expected, the estimation errors resulted by the guided filter in the RI process tend to be much smaller, compared with that yielded by the duplication approach as conducted in the CDI process. This means that the resulted *residual field* are usually much smoother than the CD field. Detailed justifications with quantitative measurements on this aspect are provided in Section 2.2.3. Consequently, the amount of adjustment, which is to be compensated to the estimated values obtained via the guided filter, can be more accurately estimated by conducting interpolation on the much smoother residual field in the Stage 2 of Fig. 2.3. In what follows, the two stages of the guided-filter-based RI process will be respectively described in detail.

2.2.1 Stage 1: Component Image Estimation

In Fig. 2.3, the guided filter has two inputs (i.e., R_m and $\hat{G}$) and one output (i.e., $\bar{R}$), where R_m denotes the initially-acquired (i.e., mosaiced) R-component image that is to be restored. In this case, the mosaiced image $p = R_m$ is guided by another component image $d = \hat{G}$, which is a *full*-resolution G-component image obtained by applying the GBTF algorithm [13] to the initially-acquired (i.e., mosaiced) G_m as practiced in [54] (not shown in Fig. 2.3).

The guided filter ‘filters’ the concerned input signal R_m with a guidance from the guiding signal $\hat{G}$. Note that the so-called ‘filtering’ here, in fact, has a very different goal to accomplish in the context of demosaicing task—that is, conducting estimation for *all* the pixels of R_m . This includes both the pixels with ground truth and those with missing values, corresponding to the *shaded* and *unshaded* pixels on R_m as shown in Fig. 2.3, respectively. This estimation is achieved by performing a linear regression

process between R_m and $\hat{G}$ over each formed local region or window. For that, a set of 2-D data points of each $M \times N$ local window, $\Omega_{i,j}$, which is centered at the pixel (i, j) , is formed by individually pairing up the *ground-truth* R values (at those shaded pixels) and their corresponding *interpolated* G values from the same pixel locations, respectively. The dimension of the rectangular window $\Omega_{i,j}$ (i.e., $M \times N$) is pre-determined, and it is fairly important as the window size determines how many data points, as well as what kind of points, will be involved in the linear regression process. Consequently, the quality of the final demosaiced image will be intimately affected by this window size. Therefore, a study of the window size selection for the guided filter has been conducted in this chapter, which is to be discussed in detail in Section 2.3.2.

By conducting the *linear ridge regression* [69] at each pixel location (i, j) based on a set of 2-D data points formed from the local window $\Omega_{i,j}$ as described previously, two real-numbered coefficients, $\alpha(i, j)$ and $\beta(i, j)$, of the linear model will be computed. Specifically, the regression is conducted by minimizing a cost function defined as follows:

$$E(\alpha(i, j), \beta(i, j)) = \sum_{(u,v) \in \Omega_{i,j}} \left(\alpha(i, j) \cdot \hat{G}(u, v) + \beta(i, j) - R(u, v) \right)^2 + \epsilon \cdot a^2(i, j), \quad (2.1)$$

where ϵ is a constant regularization parameter. This regularization term $\epsilon \cdot a^2(i, j)$ penalizes large values of $\alpha(i, j)$ that could be caused by outliers and thus improves the robustness of the regression process. In this work, $\epsilon = 10^{-4}$ is empirically determined. According to [69], the optimal $\alpha(i, j)$ and $\beta(i, j)$ that minimize (2.1) are obtained as follows:

$$\alpha(i, j) = \frac{\mu_{R \cdot \hat{G}}(i, j) - \mu_R(i, j) \cdot \mu_{\hat{G}}(i, j)}{\sigma_R^2(i, j) + \epsilon}, \quad (2.2)$$

$$\beta(i, j) = \mu_R(i, j) - \alpha(i, j) \cdot \mu_{\hat{G}}(i, j), \quad (2.3)$$

where $\mu_R(i, j) = \frac{1}{MN} \sum_{(u,v) \in \Omega_{i,j}} R(u, v)$, $\mu_{\hat{G}}(i, j) = \frac{1}{MN} \sum_{(u,v) \in \Omega_{i,j}} \hat{G}(u, v)$, and $\mu_{R \cdot \hat{G}}(i, j) = \frac{1}{MN} \sum_{(u,v) \in \Omega_{i,j}} R(u, v) \cdot \hat{G}(u, v)$ are the mean pixel values of R , $\hat{G}$, and $R \cdot \hat{G}$ computed

from the $M \times N$ local window $\Omega_{i,j}$ surrounding pixel (i, j) , respectively; $\sigma_R^2(i, j) = \frac{1}{MN} \sum_{(u,v) \in \Omega_{i,j}} (R(u, v) - \mu_R(i, j))^2$ is the variance of R computed from the same local window. Based on the obtained $\alpha(i, j)$ and $\beta(i, j)$, an estimation of the R-component image can be obtained as

$$\bar{R}(i, j) = \alpha(i, j) \cdot \hat{G}(i, j) + \beta(i, j). \quad (2.4)$$

However, for yielding a more robust estimation, such computed linear coefficients α 's and β 's will be further averaged over the local window $\Omega_{i,j}$. That is,

$$a(i, j) = \frac{1}{MN} \sum_{(u,v) \in \Omega_{i,j}} \alpha(u, v), \quad (2.5)$$

$$b(i, j) = \frac{1}{MN} \sum_{(u,v) \in \Omega_{i,j}} \beta(u, v). \quad (2.6)$$

Finally, a more robust estimation $\bar{R}(i, j)$ can be obtained as

$$\bar{R}(i, j) = a(i, j) \cdot \hat{G}(i, j) + b(i, j). \quad (2.7)$$

The simple linear regression model is chosen in the guided filter mainly due to its algorithmic simplicity and robustness to noise. Furthermore, it has been found that this model is able to yield sufficiently good approximations even near edges and textured regions. More complicated models, e.g., polynomial regression with second-order terms of the explanatory variable, has also been experimented in our simulations, but it turns out the performance is even slightly worse as the model is very sensitive to outliers. Moreover, the computational cost is also significantly increased. Finally, for the ease of presentation, the above-described guided filtering process is expressed in symbols as

$$\bar{p} = \mathfrak{R}(p|d), \quad (2.8)$$

and this short-hand notation is to be used throughout this chapter. The symbol “|” indicates that the image d is used to guide the image p . Therefore, the above-described

guided filtering process based on the input image R_m and the guidance image $\hat{G}$ can be denoted as $\bar{R} = \mathfrak{R}(R_m|\hat{G})$.

2.2.2 Stage 2: Estimation Error Compensation

Once the estimated image $\bar{R}$ with a full resolution is obtained in the Stage 1, the *residuals* (i.e., estimation errors), which reveal the pixel-wise differences between $\bar{R}$ and R_m can be computed by $\Delta R = R_m - \bar{R}$. It should be noted that the subtractions can only be taken place at those pixels with ground-truth values on R_m (i.e., those shaded pixels). The computed residual field ΔR is then subjected to conduct interpolation to obtain the *interpolated* residual field $\overline{\Delta R}$ with a full resolution. The role played by the interpolation is essentially to *predict* the estimation error possibly yielded at those non-shaded pixels on R_m , since their ground-truth values are *not* available. The values on the interpolated residual field $\overline{\Delta R}$ are then treated as the amount of adjustment to be compensated to the estimated image $\bar{R}$ for arriving at the restored image $\hat{R}$, which is an improved version of $\bar{R}$. Based on the assumption that the residual field ΔR is highly smooth, which is to be quantitatively justified in the next sub-section, it would be quite sufficient to exploit a simple interpolation technique (e.g., the bilinear interpolation as exercised in [54] and [56] as well as used in our work) to obtain the interpolated residual field $\overline{\Delta R}$.

It is highly important to note that the initially-acquired ground-truth values on the input image R_m at those shaded pixels are completely preserved in the restored image $\hat{R}$. This is because the bilinear interpolation process is only applied to those non-shaded pixels. That is, the residuals incurred at those shaded pixels on ΔR remain unchanged in $\overline{\Delta R}$ and will be added back to the corresponding pixel values in $\bar{R}$ in the last stage to get back to the original ground-truth values for those shaded pixels.

2.2.3 Residuals Versus Color-component Differences

This section provides quantitative justifications on the superior smoothness of the residual fields, compared with that of the CD fields, using the Kodak and IMAX datasets.

For that, let F denote a 2-D data field, be it a residual field or a CD field. A Laplacian filter $\mathcal{L}$ is then applied to F , followed by taking the absolute value on each filtered value; that is,

$$\delta = |\mathcal{L} * F| = \left| \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} * F \right|, \quad (2.9)$$

where $*$ denotes the convolution operation. The *degree of smoothness* (or *smoothness* in short) of F is defined as the average value computed over δ . Apparently, the smaller the measured average value, the smoother the field F . The computed measurements are documented in Table 2.1, where the symbols R , G , and B denote the ground-truth of three component images of each test image, respectively. Furthermore, $\bar{G}_R$ denotes the estimated G-component image obtained under the guidance of R ; i.e., $\bar{G}_R = \mathfrak{R}(G|R)$. The same interpretation can be applied to $\bar{G}_B = \mathfrak{R}(G|B)$, $\bar{R}_G = \mathfrak{R}(R|G)$, and $\bar{B}_G = \mathfrak{R}(B|G)$ likewise.

An important parameter of the guided filtering process is the size of the local window imposed. For that, we have conducted the guided filtering process using all possible window sizes $M \times N$, where $M, N \in \{3, 5, \dots, 17\}$. The minimum and maximum smoothness measurements obtained for each test image are then averaged over each test image dataset. Therefore, a range of measurements are documented in Table 2.1. For example, in the first entry under Kodak, 3.61 is the minimum, while 4.86 is the maximum. Further note that the smoothness measurements of the CD fields $R - G$ and $G - R$ are exactly the same; likewise, for $G - B$ and $B - G$. Therefore, only one measurement is documented in Table 2.1.

TABLE 2.1: AVERAGE SMOOTHNESS MEASURED OVER THE RESIDUAL FIELDS AND THE CD FIELDS, RESPECTIVELY. THESE MEASUREMENTS ARE CONDUCTED FOR THE KODAK IMAGE SET AND THE IMAX IMAGE SET, SEPARATELY.

Fields		Kodak	IMAX
Residual	$G - \bar{G}_R$	3.61~4.86	6.20~8.25
	$R - \bar{R}_G$	3.64~4.84	6.21~8.51
CD	$G - R$	4.03	9.46
	$R - G$		
Residual	$G - \bar{G}_B$	3.77~5.15	8.11~11.13
	$B - \bar{B}_G$	4.03~5.27	8.10~10.52
CD	$G - B$	4.40	12.97
	$B - G$		

As indicated in Table I, the degree of smoothness of the residual fields and that of the CD fields are fairly close one another on those test images in the Kodak dataset. However, when it comes to the test images in the IMAX dataset, the residual fields are clearly smoother than that of the CD fields. It has been pointed out in several existing works (e.g., [3], [68]) that the test images in the Kodak set somehow have fairly high spectral correlations, while the ones in the IMAX dataset contain more saturated colors and have much lower spectral correlations. It is not clear which test image set can better represent the natural images as highlighted in [67]. In any case, an important observation is that the degree of smoothness incurred in the residual fields is much less sensitive to the degree of spectral correlations among three color component images. Therefore, it is clearly advantageous to conduct interpolation over the residual fields rather than on the CD fields for demosaicing. This justifies why the RI-based approaches are able to provide more robust and consistent demosaicing performance.

2.3 Proposed Iterative Residual Interpolation Based Demosaicing Algorithm

The proposed *iterative residual interpolation* (IRI) process as depicted in Fig. 2.4 aims to restore a much more accurate full-resolution G channel compared with that of using the GBTF [13] as practiced in [54]. The obtained G channel is then exploited to guide the reconstruction of the R channel and of the B channel, respectively, using the RI-based scheme as shown in Fig. 2.3. Note that the IRI process will be conducted twice, one along the *horizontal* direction (with drawing as shown in Fig. 2.4) and the other along the *vertical* direction, which has same processing flow as the one shown in Fig. 2.4 except along the vertical direction. The obtained two restored G channels will then be combined via a soft decision scheme to arrive at the final demosaiced G channel. The IRI process will be described in detail as follows.

2.3.1 Iterative Residual Interpolation Process

Given three initially-acquired mosaiced images R_m , G_m , and B_m , as shown in Fig. 2.4 (a), a simple 1-D linear interpolation conducted along the horizontal direction is first applied to each of them independently to produce three interpolated images, $\tilde{R}^{(k)}$, $\tilde{G}^{(k)}$, and $\tilde{B}^{(k)}$, respectively, as shown in Fig. 2.4 (b), where the superscript (k) denotes the iteration index. As expected, these interpolated values obtained initially (i.e., at the first iteration when $k = 1$) are far from being accurate and thus need to be further refined. For that, the RI process as described in Section 2.2 will be exploited in our work to iteratively refine all the three channels; i.e., the proposed IRI process can be viewed as an image *refinement* process.

The guided filter has two inputs and one output. It conducts the linear regression based on a set of 2-D data points established on each local window imposed on the two

Figure 2.4: The proposed IRI process conducted along the *horizontal* direction only. For the *vertical*-direction case, the entire operations will be performed in the same manner, except along the vertical direction. Figures (a)-(f) denote the locations of the pixels that are involved in the processing points corresponding to the labels (a)-(f) as indicated in the block diagram, respectively. Note that the two inputs of the guided filter (i.e., p and d) are exchangeable such that the component images are guiding each other. In total, the filter will be executed four times per iteration.

input sources. Due to the inherited lattice structure of the Bayer pattern (Fig. 2.1), the *interpolated* G-component image (i.e., $\tilde{G}^{(1)}$) can be evenly split into two non-overlapping parts, respectively denoted as $\tilde{G}_R^{(1)}$ and $\tilde{G}_B^{(1)}$; that is, $\tilde{G}^{(1)} = \tilde{G}_R^{(1)} \cup \tilde{G}_B^{(1)}$. Note that $\tilde{G}_R^{(1)}$ contains all the rows extracted from $\tilde{G}^{(1)}$ that correspond to the *shaded* rows in $\tilde{R}^{(1)}$, while all remaining rows on $\tilde{G}^{(1)}$ correspond to the shaded rows in $\tilde{B}^{(1)}$. Now, the pixels on $\tilde{R}^{(1)}$ and $\tilde{B}^{(1)}$ are ready to be paired up with the pixels from $\tilde{G}_R^{(1)}$ and $\tilde{G}_B^{(1)}$, respectively, to form two sets of data points for conducting the guided filtering process, individually. Note that the number of initially-acquired ground-truth values in each input

source of the guided filter are the same, and these ground-truth values are totally non-overlapping with respect to each other². This means that any two input sources used in the guided filter can actually guide each other through the RI process to refine their initially-interpolated values. For that, the image $\tilde{G}_R^{(1)}$ will be guided by $\tilde{R}^{(1)}$ to yield estimations of the corresponding G values; i.e., $\bar{G}_R^{(1)} = \mathfrak{R}(\tilde{G}_R^{(1)}|\tilde{R}^{(1)})$. The roles of the input image p and the guidance image d can be reversed; now, the image $\tilde{G}_R^{(1)}$ guides the image $\tilde{R}^{(1)}$ for generating an estimated image of the R channel; i.e., $\bar{R}^{(1)} = \mathfrak{R}(\tilde{R}^{(1)}|\tilde{G}_R^{(1)})$. Similarly, images $\tilde{G}_B^{(1)}$ and $\tilde{B}^{(1)}$ can also guide each other as described above; that is, $\bar{G}_B^{(1)} = \mathfrak{R}(\tilde{G}_B^{(1)}|\tilde{B}^{(1)})$ and $\bar{B}^{(1)} = \mathfrak{R}(\tilde{B}^{(1)}|\tilde{G}_B^{(1)})$. Finally, the estimated image $\bar{G}^{(1)}$ can be obtained by combining the individually estimated two parts, $\bar{G}_R^{(1)}$ and $\bar{G}_B^{(1)}$, through an union operation (i.e., $\bar{G}^{(1)} = \bar{G}_R^{(1)} \cup \bar{G}_B^{(1)}$) via an interleaving manner. As a final note, all the four guided filtering processes as described above can be realized in parallel on implementation, since the involved computations are completely independent from each other.

As discussed in Section 2.2, the obtained estimated images $\bar{R}^{(1)}$, $\bar{G}^{(1)}$, and $\bar{B}^{(1)}$, as shown in Fig. 2.4 (c), are subtracted from their corresponding mosaiced images in Fig. 2.4 (a) only at those pixels that have ground-truth values (i.e., those *shaded* pixels), respectively, to arrive at the residual fields. That is, $\Delta R^{(1)} = R_m - \bar{R}^{(1)}$, $\Delta G^{(1)} = G_m - \bar{G}^{(1)}$, and $\Delta B^{(1)} = B_m - \bar{B}^{(1)}$, as shown in Fig. 2.4 (d). The computed residuals are used to *predict* the residuals at those *unshaded* pixels, where the ground-truth values are not available. For that, the 1-D linear interpolation is conducted along the horizontal direction to generate the interpolated residual fields $\overline{\Delta R}^{(1)}$, $\overline{\Delta G}^{(1)}$, and $\overline{\Delta B}^{(1)}$, as shown in Fig. 2.4 (e). As simple as it seems, it has been found that the 1-D linear interpolation is actually an effective approach for interpolating the residuals, since

²To see this, one can refer to the initially-acquired mosaiced images in Fig. 2.4 (a) with a splitting of G_m as the way did for $\tilde{G}^{(1)}$ as previously described.

the residual field is usually extraordinarily smooth as mentioned previously. These interpolated residual fields—as the compensations due to estimation errors—are then added back to the respective estimated component images to yield a set of *refined* images $\hat{R}^{(1)}$, $\hat{G}^{(1)}$, and $\hat{B}^{(1)}$, as illustrated in Fig. 2.4 (f).

The above-described RI-based refinement process will be repeated through iterations (i.e., $k = 2, 3, \dots, K$) for further improving the estimation of the missing values. That is, the three refined images in Fig. 2.4 (f) will be further fed back to the starting point of the loop for performing the same process in the next iteration by setting $\tilde{R}^{(k)} = \hat{R}^{(k-1)}$, $\tilde{G}^{(k)} = \hat{G}^{(k-1)}$, and $\tilde{B}^{(k)} = \hat{B}^{(k-1)}$. Note that the *ground-truth* values at those shaded pixels as initially recorded in the R_m , G_m , and B_m will be always preserved in each refined image (as explained in Section 2.2), while the restored values at those unshaded pixels will be progressively improved through each iteration.

2.3.2 Local Window Size Selection for the Guided Filter

As mentioned in Section 2.2, an important parameter of the guided filter is the size or dimension of the local window $\Omega_{i,j}$ (i.e., $M \times N$), which could incur large impact to the demosaicing performance. In our preliminary work in [70], the window size was gradually increased in each iteration. However, it has been observed that the IRI process failed to converge by using this approach. To address this issue, an effective strategy is developed to determine the local window size before the inception of the IRI process, and the determined window size will be constantly used throughout the IRI process. Our extensive simulations have shown that this will guarantee the convergence of the IRI process, which is to be detailed in the next sub-section.

Since it is difficult to mathematically derive the optimal window size for each test image, the proposed strategy attempts to select a suitable window size from a set of candidate windows with four different scales for each IRI process—i.e., $\Psi_h = \{(M_t, N_t) | t =$

$1, 2, 3, 4\}$ for the horizontal IRI process, and $\Psi_v = \{(N_t, M_t) | t = 1, 2, 3, 4\}$ for the vertical IRI process, where $M_t = 2t + 1$ and $N_t = 4t + 1$. In our approach, only one of the four window scales (i.e., $t = 1, 2, 3$, and 4) will be selected and used for conducting the IRI process. The selection is performed based on the three initially interpolated images, $\tilde{R}^{(1)}$, $\tilde{G}^{(1)}$, and $\tilde{B}^{(1)}$, as follows.

It has been observed through extensive simulations that using a smaller window size will lead to smoother residual field in most test cases. This trend can be observed from the *degree of smoothness* measured on the G-channel residual fields as follows. First, for each possible window scale t , the guided filtering process as described in Section 2.3.1 is applied along the horizontal and the vertical direction independently using the corresponding window sizes from Ψ_h and Ψ_v (i.e., (M_t, N_t) and (N_t, M_t)) to generate two estimated images $\bar{G}_h^{(1)}$ and $\bar{G}_v^{(1)}$, respectively. That is, for the horizontal IRI processing: $\bar{G}_h^{(1)} = \bar{G}_{R,h}^{(1)} \cup \bar{G}_{B,h}^{(1)}$, where $\bar{G}_{R,h}^{(1)} = \Re(\tilde{G}_{R,h}^{(1)} | \tilde{R}_h^{(1)})$ and $\bar{G}_{B,h}^{(1)} = \Re(\tilde{G}_{B,h}^{(1)} | \tilde{B}_h^{(1)})$. Likewise, for the vertical IRI processing: $\bar{G}_v^{(1)} = \bar{G}_{R,v}^{(1)} \cup \bar{G}_{B,v}^{(1)}$, where $\bar{G}_{R,v}^{(1)} = \Re(\tilde{G}_{R,v}^{(1)} | \tilde{R}_v^{(1)})$ and $\bar{G}_{B,v}^{(1)} = \Re(\tilde{G}_{B,v}^{(1)} | \tilde{B}_v^{(1)})$. Subsequently, the residual fields $\Delta G_h^{(1)}$ and $\Delta G_v^{(1)}$ can be respectively computed by subtracting $\bar{G}_h^{(1)}$ and $\bar{G}_v^{(1)}$ from the mosaiced image G_m only at those shaded pixel locations, where their ground-truth values are available, as previously explained. Second, the degree of smoothness of each residual field is individually measured in a similar manner as described in Section 2.2.3 except the filtering method exploited. To be more specific, rather than applying the Laplacian filter $\mathcal{L}$ as employed in (2.9), a two-tap 1-D filter is used instead for calculating the residual gradients along each direction— $\partial_h = [-1, 0, 1]$ for horizontal direction and $\partial_v = [-1, 0, 1]^T$ for vertical direction, i.e.,

$$\delta_h = \left| \partial_h * \Delta G_h^{(1)} \right| = \left| [-1, 0, 1] * \Delta G_h^{(1)} \right|, \quad (2.10)$$

$$\delta_v = \left| \partial_v * \Delta G_v^{(1)} \right| = \left| [-1, 0, 1]^T * \Delta G_v^{(1)} \right|. \quad (2.11)$$

The total degree of smoothness of the residuals, denoted as γ_t , is then computed as the

average value over δ_{min} , where $\delta_{min}(i, j) = \min(\delta_h(i, j), \delta_v(i, j))$ at each pixel (i, j) . Note that the smaller the γ_t , the smoother the residual fields.

Generally speaking, using a smaller window size will yield smoother residual fields. This means that $\gamma_t \leq \gamma_{t+1}$, for $t = 1, 2$, and 3 . Therefore, an intuitive strategy would be simply selecting the smallest window scale (i.e., $t = 1$) for all test images. However, it turns out that such a simple strategy might yield inferior demosaicing results with noticeable artifacts for some regions. To tackle this issue, rather than selecting the *smallest* window scale constantly, the *largest* window scale is adaptively chosen as long as its resulted measurement γ_t satisfies the following criterion: $\gamma_t \leq 1.2 \cdot \gamma_1$, where the constant value 1.2 is empirically determined and used in all of our experiments. Extensive simulations have shown that the proposed strategy robustly yields a much improved demosaicing performance compared to that of using a fixed window scale t for all test images. More experimental details will be reported and discussed in Section 2.4.

Lastly, it is important to bear in mind that the estimated images $\bar{G}_h^{(1)}$ and $\bar{G}_v^{(1)}$ with the iteration index (1) in the above discussions are simply generated for the purpose of measuring the total degree of smoothness of the G-channel residual fields under various window sizes. The iteration index (1) is denoted here to be consistent with the established notations. Once the window size is determined as discussed above, the entire IRI process still needs to begin with index (1) based on the selected window size.

2.3.3 Stopping Criterion for the IRI Process

As mentioned in the previous sub-section, the proposed IRI process will converge. In fact, this will happen as long as the window size for the guided filter is *fixed* during the IRI process. This convergence property has been examined by checking the *mean absolute difference* (MAD) yielded between the refined G-component images obtained in

Figure 2.5: The $MAD(k)$ values as the iteration number k increases for the *horizontal* IRI process, where $MAD(k)$ is the difference yielded between the refined G-component images obtained in iteration k and in iteration $k - 1$.

the iterations k and $k - 1$, respectively. The MAD value at iteration k is computed as

$$MAD(k) = \frac{1}{HW} \sum_{i=1}^H \sum_{j=1}^W \left| \hat{G}^{(k)}(i, j) - \hat{G}^{(k-1)}(i, j) \right|, \quad (2.12)$$

where the integers H and W are the height and the width of the image, respectively. Note that $MAD(1)$ is computed by regarding the initially interpolated G-component image as the refined G-component image at ‘iteration 0’, i.e., $\hat{G}^{(0)} \triangleq \tilde{G}^{(1)}$. Extensive simulation results have shown that the sequence of resulted $MAD(k)$ values will be monotonically decreasing towards to zero as the iteration number k increases, which means that the proposed IRI process indeed converges. To justify this aspect, two test color images are randomly picked from the Kodak dataset and from the IMAX dataset, respectively, and their $MAD(k)$ values resulted from the use of the IRI process conducted along the *horizontal* direction are shown in Fig. 2.5. Similar curves have been obtained for the IRI process performed along the *vertical* direction as well (not shown).

Despite the convergence property of the IRI process, it has been observed that the

overall quality of the refined images might, in fact, begin to degrade after several iterations. This is mainly because each IRI process is performed along one direction at a time (i.e., either in the horizontal or in the vertical direction), while in a natural image the edges and textures can be oriented along various directions. Consequently, the IRI process could cause large estimation errors when such a mismatch is incurred, which will further propagate into the subsequent iterations of the IRI process.

Based on this observation, a stopping criterion is developed in our work to yield an early termination of the IRI process, rather than simply relying on the convergence of the process. For that, a threshold $\eta = 0.5$ is empirically determined such that the $MAD(k)$ value computed at each iteration will be checked against η ; i.e., the iterative process will be terminated once $MAD(k) < \eta$ (see Fig. 2.5). According to our extensive simulations, the IRI process will be terminated after 2 to 5 iterations based on this threshold value.

2.3.4 Final Reconstruction of the G Channel

Denote K_h as the last iteration when the *horizontal* IRI process is stopped; that is, the restored G-component image along the horizontal direction is set as $\hat{G}_h = \hat{G}_h^{(K_h)}$. To obtain the restored G-component image along the *vertical* direction, the entire IRI process will be applied to the initially-acquired mosaiced images as depicted in Fig. 2.4, except along the vertical direction. Suppose the vertical IRI process is stopped at the iteration index K_v , then $\hat{G}_v = \hat{G}_v^{(K_v)}$.

In order to generate the final restored G-component image, the two restored G-component images, $\hat{G}_h$ and $\hat{G}_v$, are subject to be linearly combined using a soft-decision scheme. Note that this combination operation is only carried out at those *unshaded* pixels where their corresponding G values are missing, while the initially-acquired ground-truth values are still preserved and presented in the final demosaiced image. For that, the weights of $\hat{G}_h(i, j)$ and $\hat{G}_v(i, j)$, denoted as $w_h(i, j)$ and $w_v(i, j)$, respectively, are com-

puted by simply taking the inverse of the square of the residual gradient. The rationale behind this is that a smaller residual gradient amplitude means a locally smoother residual field, therefore the corresponding restored G-component value, obtained via residual interpolation, should be considered more accurate and assigned with larger weights. This step is similar to that of several existing CDI-based methods (e.g., [13],[16]), except that the residual gradient, rather than the CD gradient, is used to compute these weights, i.e.,

$$w_h(i, j) = \frac{1}{\kappa_h^2(i, j)}, \quad (2.13)$$

$$w_v(i, j) = \frac{1}{\kappa_v^2(i, j)}, \quad (2.14)$$

where $\kappa_h = \mathcal{F} * |\partial_h * \Delta G_h^{(K_h)}|$ and $\kappa_v = \mathcal{F} * |\partial_v * \Delta G_v^{(K_v)}|$. Symbol $*$ denotes the convolution operation, and ∂_h and ∂_v are the 1-D gradient operators as previously used in (2.10) and (2.11). The Gaussian filter $\mathcal{F}$ is further deployed here to take the neighboring pixels into consideration for yielding more reliable measurements on the residual gradients. The 9×9 Gaussian kernel with a standard variation of 2 is empirically determined and set for $\mathcal{F}$. Finally, $\hat{G}_h$ and $\hat{G}_v$ can be linearly combined (for those unshaded pixels only) as

$$\hat{G}(i, j) = \frac{w_h(i, j) \cdot \hat{G}_h(i, j) + w_v(i, j) \cdot \hat{G}_v(i, j)}{w_h(i, j) + w_v(i, j)}. \quad (2.15)$$

2.3.5 Final Reconstructions of the R and B Channels

Under the guidance of the restored G-component image $\hat{G}$ as obtained from (2.15), the R channel and the B channel can be separately reconstructed using the same RI-based reconstruction scheme as employed in [54] (Fig. 2.3).

As highlighted in Section 2.3.2 previously, the quality of the reconstructed G channel is fairly sensitive to the local window size used in the guided filtering. However, it turns out that the reconstruction of the R channel and of the B channel are, in fact, much less

sensitive to the window size exploited. This could be due to the following two reasons: (1) the G channel has been well reconstructed and used for guiding the demosaicing process of the R channel and of the B channel, respectively; and (2) the processing flow of Fig. 3 is *non-iterative*. Consequently, the impact of which window size is used becomes quite limited. With further investigation, the same window size configuration as used in [54], i.e., 11×11 , is adopted, as it yields a slightly better demosaicing performance than that of using other square windows, such as 7×7 , 9×9 , and 13×13 .

2.4 Experimental Results

The proposed demosaicing method has been evaluated using the Kodak dataset (24 images) and the IMAX dataset (18 images), since they have been commonly used for evaluating demosaicing performance. Each full-color test image is first downsampled according to the Bayer pattern [55], followed by conducting the demosaicing process using various methods under comparison. Our proposed method is compared with six state-of-the-art demosaicing methods: the *learned simultaneous sparse coding* (LSSC) [62], the *integrated gradient* (IGD) [9], the *gradient-based threshold free demosaicing* (GBTf) [13], the *local directional interpolation and nonlocal adaptive thresholding* (LDI-NAT) [14], the *residual interpolation* (RI) [54], and the *minimized-Laplacian residual interpolation* (MLRI) [56]. Both the objective performance evaluation and subjective quality assessment will be conducted as follows.

2.4.1 Objective Performance Evaluations

For quantitatively evaluating the objective performance of the demosaicing process, the difference incurred between the original and the demosaiced images are measured based on two commonly-used image fidelity metrics—i.e., the *peak signal-to-noise ratio* (PSNR) and the *structural similarity* (SSIM) [71]. Both have been measured for each color channel

separately as well as for all the three channels jointly. Note that pixels that are less than 10 pixels from the image border are excluded from the calculation of these measurements. For the combined measurement of the PSNR, a widely-used PSNR measurement metric, called the *color peak signal-to-noise ratio* (CPSNR), is adopted, which is defined as

$$\text{CPSNR} = 10 \log \frac{255^2}{\frac{1}{3HW} \sum_i^H \sum_j^W \| I_o(i, j) - I_d(i, j) \|_2^2}, \quad (2.16)$$

where $\| \cdot \|_2$ denotes the l_2 norm of a vector. Integers H and W denote the height and the width of the image, respectively. Note that both $I_o(i, j)$ and $I_d(i, j)$ are vectors, representing the RGB values at pixel (i, j) on the original image and the demosaiced image, respectively. For a joint measurement of the SSIM, we simply compute the *average* SSIM over the three SSIM values individually obtained from the three channels.

The PSNR and CPSNR results are documented in Table 2.2. It can be observed that our method has delivered the highest average PSNR among all these methods over the IMAX dataset. Moreover, one can see that our proposed method constantly outperforms the two RI-based methods (i.e., [54] and [56]) on both datasets. It is important to point out that, although our developed demosaicing approach adopts the same RI-based reconstruction strategy (Fig. 2.3) for the reconstruction of the R channel and of the B channel, better reconstruction results have been achieved for these two channels. This is because our proposed IRI-based approach (Fig. 2.4) is able to restore a much more accurate full-resolution G channel in the first place, which will be served as the guidance image to guide the reconstruction of the R channel and of the B channel, respectively.

It should be noted that several CDI-based methods (i.e., IGD [9] and GBTF [13]) have achieved better PSNR performance than the proposed method on the Kodak dataset. This could be explained by the fact that the spectral correlations existing among the three color channels in most test images from the Kodak dataset are extraordinarily high, as discussed in several previous works (e.g., [68]). This could effectively benefit the

TABLE 2.2: AVERAGE PSNR AND CPSNR RESULTS (IN dB) ON THE KODAK AND THE IMAX DATASETS. THE HIGHEST AVERAGE PSNR RESULT IN EACH COLUMN IS HIGHLIGHTED IN BOLDFACE.

(a) Average PSNR and CPSNR results (in dB) on the Kodak dataset.

Methods	PSNR			CPSNR
	R	G	B	
LSSC [62]	40.53	44.31	40.65	41.44
IGD [9]	39.66	43.39	40.02	40.63
GBTf [13]	39.48	43.12	39.85	40.43
LDI-NAT [14]	37.04	39.45	37.12	37.71
RI [54]	37.90	40.95	37.79	38.58
MLRI [56]	38.83	41.77	38.81	39.53
Proposed	38.82	42.38	39.15	39.77

(b) Average PSNR and CPSNR results (in dB) on the IMAX dataset.

Methods	PSNR			CPSNR
	R	G	B	
LSSC [62]	36.04	38.82	34.72	36.17
IGD [9]	34.33	37.38	33.45	34.69
GBTf [13]	33.54	36.57	32.72	33.92
LDI-NAT [14]	36.29	39.79	34.48	36.25
RI [54]	36.07	39.96	35.35	36.48
MLRI [56]	36.32	39.87	35.34	36.60
Proposed	36.80	40.28	35.72	37.06

(c) Average PSNR and CPSNR results (in dB) on the Kodak and IMAX datasets combined.

Methods	PSNR			CPSNR
	R	G	B	
LSSC [62]	38.60	41.96	38.11	39.18
IGD [9]	37.38	40.82	37.21	38.09
GBTf [13]	36.94	40.31	36.79	37.64
LDI-NAT [14]	36.72	39.59	35.99	37.08
RI [54]	37.12	40.53	36.75	37.68
MLRI [56]	37.75	40.96	37.33	38.28
Proposed	37.96	41.48	37.68	38.61

TABLE 2.3: AVERAGE SSIM RESULTS ON THE KODAK AND THE IMAX DATASETS. THE HIGHEST AVERAGE SSIM RESULT IN EACH COLUMN IS HIGHLIGHTED IN BOLDFACE.

(a) Average SSIM results on the Kodak dataset.

Methods	R	G	B	Average
LSSC [62]	0.9848	0.9925	0.9835	0.9869
IGD [9]	0.9833	0.9913	0.9829	0.9858
GBTF [13]	0.9827	0.9910	0.9824	0.9854
LDI-NAT [14]	0.9727	0.9828	0.9710	0.9755
RI [54]	0.9759	0.9866	0.9730	0.9785
MLRI [56]	0.9803	0.9886	0.9788	0.9826
Proposed	0.9790	0.9892	0.9782	0.9821

(b) Average SSIM results on the IMAX dataset.

Methods	R	G	B	Average
LSSC [62]	0.9550	0.9734	0.9274	0.9519
IGD [9]	0.9403	0.9622	0.9052	0.9359
GBTF [13]	0.9317	0.9558	0.8927	0.9267
LDI-NAT [14]	0.9604	0.9787	0.9266	0.9552
RI [54]	0.9591	0.9795	0.9400	0.9595
MLRI [56]	0.9601	0.9785	0.9384	0.9590
Proposed	0.9633	0.9818	0.9411	0.9621

(c) Average SSIM results on the Kodak and IMAX datasets combined.

Methods	R	G	B	Average
LSSC [62]	0.9720	0.9843	0.9595	0.9719
IGD [9]	0.9649	0.9788	0.9496	0.9644
GBTF [13]	0.9609	0.9759	0.9439	0.9602
LDI-NAT [14]	0.9674	0.9811	0.9520	0.9668
RI [54]	0.9687	0.9836	0.9589	0.9704
MLRI [56]	0.9717	0.9843	0.9615	0.9725
Proposed	0.9723	0.9860	0.9623	0.9735

generated CD fields becoming smoother than the residual fields. As expected, when it comes to color images with lower spectral correlations among the color channels, such as those in the IMAX dataset, these CDI-based methods produce quite poor results. In other words, the CDI-based methods are sensitive to the amount of spectral correlations inherited in the color image. In contrast, our proposed method is much less sensitive to the spectral correlations and thus achieves much more robust performance.

Although the sparse-representation-based LSSC [62] method has delivered the best PSNR performance on the Kodak dataset, however it requires to learn an initial dictionary from a set of external images for reconstructing the mosaiced image via sparse representation. Such learning procedure is not only computational expensive but also highly data-dependent. For the latter, note that the LSSC’s PSNR performance drops drastically on the IMAX dataset, while our proposed method is more robust.

The SSIM metric is known to be more consistent with the human visual perceptions on evaluating the image quality. The measured SSIM results are documented in Table 2.3, from which one can see that our proposed approach has achieved the highest average SSIM score on the Kodak and the IMAX datasets combined (denoted as “Kodak+IMAX” in Table 2.3). Specifically, it can be observed that, in terms of SSIM, our method is able to deliver a fairly close performance compared with those of the CDI-based methods and the sparse-representation-based LSSC method on the Kodak dataset, while yielding a much higher performance gain on the IMAX dataset. All the above-mentioned justifications clearly demonstrate the superior performance of the proposed demosaicing algorithm.

2.4.2 Subjective Quality Assessment

Besides the objective evaluations on the demosaiced image quality, our proposed method shows clear advantages on the aspect of subjective quality assessment as well. Four representative test images, out of the total 42 images from the Kodak and the IMAX

Figure 2.6: Four representative test images from the Kodak and the IMAX datasets.

datasets combined, are demonstrated in Fig. 2.6 (a)-(d), respectively. To conduct subjective evaluation through visual comparison, the demosaicing results of four cropped-up sub-images (as indicated by the green-colored frames in each test image) are presented in Figs. 2.7-2.10, respectively.

In Fig. 2.7, it can be seen that the sub-image of the “IMAX 5” image (Fig. 2.7 (a)) contains some areas with highly-saturated colors (i.e., those red and yellow stripes). As previously mentioned, the CD fields tend to be unsmooth around the boundaries of such areas owing to stark color contrast. As a result, more distinct color artifacts have been yielded by those CDI-based demosaicing methods (i.e., IGD [9], GBTF [13], and LDI-NAT [14]); see the black and white dots around the boundaries of yellow and red stripes in Fig. 2.7 (c)-(e). Similar artifacts have also been produced by the sparse-based LSSC [62] method; see Fig. 2.7 (b). On the other hand, the RI-based demosaicing methods (i.e., RI [54] and MLRI [56]) have produced much improved demosaiced images since the residual fields are much smoother than the CD fields in this case, as can be assessed from Fig. 2.7 (f)-(g). However, some color artifacts can still be observed in these two images; for example, many black dots can be observed within the yellow stripe. On the contrary, by properly utilizing the superior smoothness property of the residual fields, the proposed IRI-based approach is able to generate a full-resolution G channel with much improved quality, compared with that of [54] that exploits the GBTF [13] method.

Figure 2.7: Visual comparison of the demosaicing results for a region in the “IMAX 5” image.

Figure 2.8: Visual comparison of the demosaicing results for a region in the “IMAX 17” image.

(a) Original

(b) LSSC [62]

(c) IGD [9]

(d) GBTF [13]

(e) LDI-NAT [14]

(f) RI [54]

(g) MLRI [56]

(h) Proposed

Figure 2.9: Visual comparison of the demosaicing results for a region in the “Kodak 8” image.

Figure 2.10: Visual comparison of the demosaicing results for a region in the “Kodak 19” image.

The restored full-resolution G channel is then exploited to guide the reconstruction of the R channel and the B channel, respectively. All these jointly deliver the best visual quality (see Fig. 2.7 (h)) among all the methods under comparison. The same comparison statements can be drawn for the results on the “IMAX 17” image, which is presented in Fig. 2.8.

Figs. 2.9 and 2.10 display the demosaicing results of a sub-image respectively cropped from the “Kodak 8” and the “Kodak 19” images. Both these two cropped sub-images contain dense edges clustered in a small local region. In Fig. 2.9, it can be easily observed that our method has delivered distinctly superior result in the “roof” area, with the demosaiced image almost identical to the original image, while, on the other hand, severe color leakage can be observed in the results of LDI-NAT [14], RI [54], and MLRI [56], as shown in Fig. 2.9 (e)-(g). The LSSC [62], IGD [9], and GBTF [13] methods have yielded considerably better results than those in Fig. 2.9 (e)-(g), but they still produced noticeable color artifacts, as illustrated in Fig. 2.9 (b)-(d). The demosaiced results of another widely-used test image with dense edges are shown in Fig. 2.10. It can be observed that the LDI-NAT [14] and MLRI [56] methods have produced the most distinct artifacts in this case (i.e., false colors), while the other methods have delivered much better demosaiced image quality and their results are fairly close one another. However, with a closer look, one can see that the proposed method has yielded the least amount of artifacts at the right bottom area of the sub-image, where other methods have all produced visible blueish color leakage.

In addition to the above-described subjective evaluations, a further visual comparison of the results obtained by the LSSC algorithm and the proposed IRI method is conducted on two representative images from the Kodak dataset, as shown in Fig. 2.11. From the PSNR values illustrated in the captions, one can see that the LSSC algorithm has delivered considerably higher PSNRs than ours for both demonstrated image patches. In

Figure 2.11: Visual comparison of the demosaicing results of the LSSC algorithm and the proposed IRI algorithm.

Fig. 2.11 (c) and (d), it can be seen that the LSSC performs clearly better around the eyes of the face logo. This is because its sparse model learned from small training image patches can capture such small structures (given that similar structures can be found among the training images), while the guided filter employed in our algorithm sometimes could treat those structures as “outliers” and fail to fit them accurately. However, it should be pointed out that the LSSC algorithm has also produced more noticeable artifacts on the left-side boundary of the logo. As for the Fig. 2.11 (g) and (h), these two results are actually quite distinguishable, despite the large difference on PSNR values. This again demonstrates that PSNR is not a good fidelity metric that is in line with human visual perceptions. In conclusion, although the LSSC outperforms the proposed IRI algorithm on PSNR, its advantage in terms of perceptual quality is not so significant and sometimes even unnoticeable.

TABLE 2.4: AVERAGE RUNNING TIME (IN SECONDS) OF DIFFERENT METHODS FOR DEMOSAICING ONE IMAGE ON THE KODAK AND THE IMAX DATASETS. THE SPATIAL RESOLUTION OF TEST IMAGES IN EACH DATASET IS INDICATED BELOW THE CORRESPONDING DATASET NAME.

Methods	Kodak (512×768)	IMAX (500×500)
LSSC [62]	565.06	363.50
GBTF [13]	0.11	0.07
LDI-NAT [14]	1430.64	887.14
RI [54]	1.52	0.95
MLRI [56]	2.63	1.64
Proposed	8.78	4.71

2.4.3 Comparison of Computational Complexity

The proposed IRI-based demosaicing algorithm is implemented in Matlab and its computational complexity is compared with other five state-of-the-art demosaicing methods that have available Matlab source codes³. All these methods are tested on the same machine with an Intel Core i5-2400 3.1 GHz CPU and 8 GB RAM. The running times of all methods are documented in Table 2.4.

As expected, the proposed IRI-based method is computationally more expensive than the GBTF [13] method and the two *non-iterative* RI-based methods—i.e., RI [54] and MLRI [56], due to its iterative processing flow. However, it delivers considerably better demosaicing performance than these three methods, as described in Sections 2.4.1 and 2.4.2. On the other hand, our proposed method is significantly more efficient than the LSSC [62] and LDI-NAT [14] methods. Note that although the LSSC method delivers the most comparable (and sometimes even superior) demosaicing performance to that of ours in terms of objective and subjective evaluations, our method is clearly more practical due to its much lower computational complexity.

³Since the IGD [9] method has only made executable code available, thus it is not involved in this comparison.

TABLE 2.5: AVERAGE PSNR AND SSIM RESULTS (FOR THE G CHANNEL) OBTAINED BY USING DIFFERENT WINDOW SIZE CONFIGURATIONS

Window Size	Kodak		IMAX	
	PSNR	SSIM	PSNR	SSIM
$M_1 = 3, N_1 = 5$	39.62	0.9806	40.09	0.9810
$M_2 = 5, N_2 = 9$	41.62	0.9874	39.95	0.9808
$M_3 = 7, N_3 = 13$	42.12	0.9886	38.39	0.9734
$M_4 = 9, N_4 = 17$	42.22	0.9888	38.18	0.9720
Proposed	42.37	0.9892	40.29	0.9818

2.4.4 Effectiveness of Proposed Window Size Selection Strategy

To demonstrate that the local window size of the guided filter does have a significant influence on the performance of the proposed IRI-based G-channel demosaicing approach, and to justify the effectiveness of our proposed window size selection strategy, we have further tested our method using each of the four window size configurations as described in Section 2.3.2. The resulting average PSNR and SSIM measurements using each fixed configuration, as well as that using the adaptive configuration obtained through our proposed window size selection strategy, are documented in Table 2.5.

First of all, it can be observed from Table 2.5 that significantly different results have been yielded when using different window size configurations, which confirms that the choice of the window size does have a large impact to the performance. It is also worth noting that, images in the Kodak dataset generally prefer large local windows (see the “Kodak” column), while most images in the IMAX dataset, in contrast, favor small window size (see the “IMAX ” column). Secondly, it can be observed that through the proposed window size selection strategy, distinctly superior performance has been achieved compared with that of using fixed window sizes for all test images.

2.5 Summary

In this chapter, a new demosaicing method based on the very recently-developed residual interpolation (RI) approach is proposed that further exploits the RI-based demosaicing strategy to reconstruct the most important G channel. The key novelty of our method lies in the newly developed *iterative residual interpolation* (IRI) process which deploys the RI process to all the three channels iteratively. The proposed IRI algorithm is able to restore a much more accurate G channel, compared with the existing start-of-the-art methods, with which the restoration of the R channel and the B channel will be greatly benefited as well. Extensive simulation results on commonly-used test datasets have clearly shown that our proposed method is able to deliver superior quality of the demosaiced image in terms of both objective evaluation and subjective assessment.

Chapter 3

Fast Contrast-guided Interpolation with Convolutional Edge Diffusion

3.1 Introduction

Image interpolation or up-scaling is a fundamental image processing task that aims to generate a higher-resolution (HR) image from a lower-resolution (LR) version. It offers a practical and economical means to increase the resolution of an image when its original resolution is not large enough, which is critical for generating high-resolution imagery for the HD displays. Although image interpolation has been an active area of research for many years, it remains a very challenging problem. Due to the high resolution and large size of HD displays, this task becomes even more difficult because various visually unpleasant artifacts caused by interpolation will be more easily noticeable on the screen.

Conventional interpolation algorithms, such as bilinear [20], bicubic spline [21], and bicubic convolution [22], are computationally efficient and easy to implement, but they often suffer from certain forms of artifacts, such as blurring, aliasing, ringing, jaggy edges, etc. To achieve high image quality, more sophisticated interpolation methods have been proposed; among them, the *edge-guided interpolation* (EGI) methods (e.g., [23]–[31]) have been considered as a fairly attractive approach. The basic idea of this approach is that: if the pixel under consideration is situated on an edge, then the interpolation of its asso-

ciated missing pixels should be conducted along the same direction of the corresponding edge in order to preserve edge sharpness and avoid image artifacts. To accomplish this objective, edge information needs to be extracted from the input image first, followed by utilizing this information to guide the interpolation process. In [26], Li and Orchard exploited the local covariance estimated from the LR image as the edge information to adjust the coefficients of the interpolation filter. In [27], Zhang and Wu applied the *linear minimum mean square error* (LMMSE) method to adaptively fuse the interpolated values that were individually obtained along various directions. In [28], Zhang and Wu exploited a 2-D piecewise autoregressive process to model the image structure for estimating a block of missing pixels simultaneously using a soft-decision scheme. In [30], Giachetti and Asuni developed an *iterative curve based interpolation* (ICBI) method that uses the local second-order image derivative information to adaptively interpolate the missing pixel values.

Besides the EGI methods, the learning-based methods have also been developed, which explore the prior knowledge learned from a set of external images (e.g., [72]–[78]) or from the input image itself (e.g., [79]–[81]) for conducting the interpolation process. Although these methods are sometimes more capable of restoring fine image details, they are highly data-dependent (hence, less robust) and generally more computationally expensive.

Recently, a novel interpolation method, called the *contrast-guided interpolation* (CGI) [32], has been proposed and demonstrated state-of-the-art performance on the interpolated image quality. The CGI is the first algorithm that innovatively incorporates the local *contrast* information into the interpolation process. The novelty to fulfill this goal lies on the re-evaluation and possible re-classification of some edge-nearby non-edge pixels as ‘edge’ ones to conduct directional interpolation for their associated to-be-interpolated pixels, respectively. This re-classification is realized by applying an edge diffusion pro-

cess, called the *iterative edge diffusion* (IED), in the CGI to diffuse the variations on each *directional variation* (DV) field computed from the input image under a specific direction. Each diffused DV field is *continuous*-valued and will be thresholded to generate a *contrast-guided decision map* (CDM), which is *binary*-valued and will be used to conduct the directional interpolation along various directions, respectively. There are four such DV fields that are subject to be diffused and then thresholded in the CGI method [32].

Despite its superior interpolation performance, the CGI's IED process incurs high computational complexity and algorithmic complexity, due to its iterative updating process on generating four diffused DV fields. This inevitably hinders the use of CGI in those applications that require real-time performance and low power consumption. To address this critical issue, a *fast* edge diffusion scheme, called the *convolutional edge diffusion* (CED), is proposed in this chapter, which greatly reduces the computation time while maintaining superior performance on image quality. Our work was motivated by a critical observation as follows. Since each diffused DV field (*continuous*-valued) will be thresholded to generate a *binary* CDM in the subsequent step [32], such binarization operation will inevitably destroy the fidelity that was preserved in the previous stage through the data term of the IED's energy functional. Furthermore, due to the presence of this data term, an iterative optimization process is performed in the IED, which is computationally expensive. Therefore, this data term is lifted in our approach to yield a simplified energy functional to begin with, and the derived diffusion equation bears the well-known heat equation form with a very significant property to be utilized on the development of the proposed CED.

The rest of this chapter is organized as follows. Section 3.2 provides a succinct description of the original CGI algorithm as necessary background for the development of our proposed method. Section 3.3 describes the proposed fast CGI algorithm with the developed CED process. Performance evaluations of the developed fast CGI algorithm

against the original CGI and several other state-of-the-art methods are presented in Section 3.4. A summary of this chapter is provided in Section 3.5.

3.2 Contrast-guided Image Interpolation (CGI)

Given a low-resolution image I_l , four DV fields U_θ need to be generated, one for each direction θ under consideration. In [32], the four considered directions are 45° , 135° , 0° , and 90° . Each U_θ is computed by $U_\theta = |I_l * d_\theta|$, where $*$ denotes the convolution and d_θ is a differentiation filter for extracting the edges of I_l along the direction that is *perpendicular* to θ . The generated four *continuous*-valued DV fields U_θ are then diffused individually by minimizing the following energy functional:

$$E(u_\theta) = \iint \left\{ U_\theta^2 (u_\theta - U_\theta)^2 + \lambda \left[(u_\theta)_x^2 + (u_\theta)_y^2 \right] \right\} dx dy, \quad (3.1)$$

where u_θ denotes the *diffused* continuous-valued DV field. (The argument (x, y) of $U_\theta(x, y)$ and $u_\theta(x, y)$ is omitted for the ease of presentation.) Symbols $(u_\theta)_x$ and $(u_\theta)_y$ represent the partial derivatives of u_θ along the horizontal (i.e., x -axis) and the vertical (i.e., y -axis) directions, respectively.

The first term of the integrand in (3.1) is called the *data term*, which is the fidelity constraint imposed on the diffusion process, reflecting how close of the diffused u_θ when compared it with the undiffused U_θ . Note that besides the commonly-used “squared error” term, $(u_\theta - U_\theta)^2$, a factor U_θ^2 is further incorporated for better preserving those large variations on the initial DV field U_θ . The second term is called the *smoothness term*, which measures the degree of *unsmoothness* (or *variation*) incurred in u_θ . The constant λ provides a trade-off between these two terms. Note that $\lambda = 0.2$ was empirically determined and used in [32].

Based on the calculus of variations [82], the optimal u_θ that minimizes the functional

$E(u_\theta)$ in (3.1) satisfies the following *Euler-Lagrange differential equation*:

$$\frac{\partial E}{\partial u_\theta} - \frac{d}{dx} \frac{\partial E}{\partial (u_\theta)_x} - \frac{d}{dy} \frac{\partial E}{\partial (u_\theta)_y} = 0. \quad (3.2)$$

The above equation can be equivalently written as

$$U_\theta^2(u_\theta - U_\theta) - \lambda \nabla^2 u_\theta = 0, \quad (3.3)$$

where ∇^2 denotes the Laplacian operator. The closed form solution for (3.3), however, is difficult to obtain. Therefore, a numerical approach is resorted to, which converts the problem into seeking the *steady-state solution* of the following diffusion equation [32]:

$$\frac{\partial u_\theta(t)}{\partial t} = \lambda \nabla^2 u_\theta(t) - U_\theta^2(u_\theta(t) - U_\theta), \quad (3.4)$$

where t is a time variable. Based on (3.4), u_θ can be iteratively updated through

$$u_\theta(t + \Delta t) = u_\theta(t) + \frac{\partial u_\theta(t)}{\partial t}, \quad (3.5)$$

starting from $u_\theta(0) = U_\theta$. Note that the second term of (3.5) on the right-hand side can be found in (3.4). The steady-state u_θ is considered reached when there is no further noticeable change incurred between $u_\theta(t + \Delta t)$ and $u_\theta(t)$ in (3.5), or when (3.4) is sufficiently close to zero.

After diffusion, the obtained four *continuous*-valued diffused DV fields will be binarized via a simple thresholding process to generate four *binary*-valued fields, respectively. In [32], these binary fields are called the *contrast-guided decision maps* (CDMs). For $CDM_\theta(i, j) = 1$, this means the pixel $I_l(i, j)$ will be treated as an *edge* pixel situated on an edge with the direction of θ . The four binary CDMs are then used to guide the interpolation process through two stages, as illustrated in Fig. 3.1. Note that, for the 2×2 image up-scaling, each pixel $I_l(i, j)$ registered on the lattice of the HR image $I(i, j)$ has three missing pixels associated—i.e., one *diagonal* pixel $I(2i, 2j)$ plus two *off-diagonal* pixels

Figure 3.1: The two-stage interpolation framework as adopted by the CGI method: (a) interpolation of the *diagonal* pixels $I(2i, 2j)$; and (b) interpolation of the *off-diagonal* pixels $I(2i - 1, 2j)$ and $I(2i, 2j - 1)$. The black and hollow dots denote the available and missing pixels, respectively.

$I(2i - 1, 2j)$ and $I(2i, 2j - 1)$. In Stage 1 as shown in Fig. 3.1 (a), the interpolation of the diagonal pixels will be conducted first under the guidance of CDM_{45° and CDM_{135° , followed by the interpolation of the off-diagonal pixels according to CDM_{0° and CDM_{90° in Stage 2 as illustrated Fig. 3.1 (b). In any stage, for each $CDM_\theta(i, j) = 1$, the 1-D cubic spline filtering will be performed to conduct interpolation along the direction of θ ; otherwise, the 2-D isotropic bicubic filtering will be applied instead. For the detailed interpolation process, refer to the algorithm steps as described in [32].

3.3 Proposed Fast CGI via Convolutional Edge Diffusion (CED)

Despite the superior interpolation performance achieved by the CGI, it has been noticed that its iterative edge diffusion process (IED) is computationally expensive as it requires

an iterative updating process to generate four diffused DV fields. To reduce the computational cost, a *fast* edge diffusion approach, called *convolutional edge diffusion* (CED), is developed in this chapter.

3.3.1 Proposed CED Algorithm

As discussed previously in Section 3.2, the data term of (3.1), $U_\theta^2(u_\theta - U_\theta)^2$, essentially imposes a fidelity constraint for the iterative optimization process to minimize $E(u_\theta)$. This term has the effect of enforcing each diffused DV field u_θ to be as close to its corresponding initial DV field U_θ as possible. As a result, this data term allows the IED scheme to preserve the prominent edge information contained in the original DV field U_θ . However, we have observed that each diffused DV field will be binarized in the subsequent step to generate its corresponding binary CDM in the CGI framework. This could make such carefully designed, and computationally expensive, IED scheme with the above-described fidelity constraint becoming ‘over-sophisticated’, since the binarization operation inherited in the thresholding process is a very crude process. Based on this observation, a new energy functional is proposed in our approach by simply dropping this data term as

$$E_s(u_\theta) = \iint \left[(u_\theta)_x^2 + (u_\theta)_y^2 \right] dx dy. \quad (3.6)$$

By following the same derivations as presented in [32], a new diffusion equation can be easily arrived at

$$\partial u_\theta(t) / \partial t = \nabla^2 u_\theta(t). \quad (3.7)$$

Following the same development and arguments as that of the IED, the steady-state solution of (3.7) will minimize the new energy functional in (3.6), and this can be obtained by iteratively updating $u_\theta(t)$ based on (3.5) until it converges, where the derivative $\partial u_\theta(t) / \partial t$ in (3.5) can be found in (3.7).

Even without performing the above-described iterative optimization, in fact one can easily judge from the objective functional in (3.6) and see that $u_\theta = c$, where c is a constant and corresponds to a completely uniform field, is the optimal solution on minimizing $E_s(u_\theta)$. This is clearly undesirable for our image interpolation task because this means that all the edge information, which is originally presented on U_θ , will be completely ‘flatten’ or destroyed. (This means the generated binary CDM will be all zeros, leading to no guidance to the follow-up interpolation process.) To avoid such issue, a stopping criterion is imposed in our approach to terminate the iteration process before it converges. Obviously, such early termination (before converging) will lead to a non-optimal $u_\theta(t)$ on minimizing (3.6). However, this sacrifice on optimality causes no concern on the generation of CDMs for our image interpolation task because the edge information on the original DV fields, which is crucial for guiding the follow-up interpolation process, is still largely preserved in the diffused DV fields.

Interestingly, it turns out that the derived (3.7) bears the form of the well-known *heat equation*, which was originally developed to model the conduction of heat in an isolated and uniform material. Furthermore, there is a very significant and useful property of the heat equation that can be exploited to greatly benefit our DV-field diffusion task, particularly on the aspect of computational efficiency. That is, given a desired amount of diffusion to yield, it can be realized by simply convolving the *initial* DV field U_θ with a Gaussian kernel *once*, rather than gradually updating the DV field through iterations [83]. To be more specific, the resultant $u_\theta(t)$ computed at each time t of the above-described iterative updating process can be directly obtained by convolving the initial DV field U_θ with a continuous 2-D isotropic Gaussian kernel $G_{(\mu, \sigma^2)}$, where the mean $\mu = 0$ and the standard deviation $\sigma = \sqrt{2t}$. (Refer to the appendix at the end of this chapter for

detailed derivation of this property). That is,

$$u_\theta(t) = G_{(0,2t)} * U_\theta. \quad (3.8)$$

Therefore, one can determine at what time t the diffusion should be stopped by setting the value of the parameter σ of the Gaussian kernel accordingly. Hence, our proposed fast diffusion method is coined as *convolutional edge diffusion* (CED).

Lastly, in order to conduct diffusion on the 2-D image lattice, an $N \times N$ discrete Gaussian kernel is used in our approach to approximate the continuous Gaussian kernel. In this case, the kernel width (i.e., filter length) N also becomes a crucial parameter that needs to be properly determined, besides σ . In Section 3.4, the selection of both N and σ will be discussed in detail.

3.3.2 Comparison of the Proposed CED and the CGI's IED

Our proposed CED scheme is clearly advantageous on both computational efficiency and algorithmic simplicity, while it is able to generate nearly identical CDMs as those yielded by the computationally expensive IED method. To justify this claim, a set of diffused DV fields (*continuous*-valued) and their resulted CDMs (*binary*-valued), experimented on the test image “Lena”, are demonstrated in Fig. 3.2. Note that since our proposed CED scheme does not have any fidelity constraint being imposed on the diffusion process, it is expected that less edge information will be preserved on its diffused DV field. Indeed, this is the case and can be observed in Fig. 3.2 (b). To compare it with the diffused DV field yielded by the IED as shown in Fig. 3.2 (a), their difference image is presented in Fig. 3.2 (c). One can see that there are clear differences at those edges as being circled. As expected, such differences are, greatly diminished in the final *binary* CDMs due to the thresholding process that binarizes each continuous-valued diffused DV field into a binary CDM. This can be observed from Fig. 3.2 (f), which is the difference map yielded between

Figure 3.2: A comparison of the diffused DV fields (*continuous*-valued) and the final CDMs (*binary*-valued), generated by the CGI’s IED and our proposed CED, respectively, using the test image “Lena”: (a) ; (b) u_{0° by CED ; (c) differences between (a) and (b); (d) CDM_{90° by IED; (e) CDM_{90° by CED; and (f) differences between (d) and (e).

the two binary CDMs as shown in Figs. 3.2 (d) and (e). In this difference map, the green pixels are those pixels that were classified as ‘edge’ pixels using our proposed CED but being treated as ‘non-edge’ pixels using the IED; for the red pixels, the interpretation is reversed. One can see that all the green and red pixels together only occupy a very small portion of the total number of pixels, and they are distributed quite randomly. This means that these two CDMs are nearly identical, practically speaking.

Such conclusion can be drawn for all 500 test images from the BSDS500 dataset [84] that we have experimented. The BSDS500 dataset is a widely-used benchmark dataset for image segmentation algorithms and it contains images with a variety of contents. It has been observed from these 500 images that the total discrepancy (i.e., the total proportion of such red and green pixels) between the generated CDMs of our CED and

Figure 3.3: The eight test images used in our experiments.

the IED is only 0.82%. Since the generation of CDMs is the core of the CGI framework as they will be used to guide the follow-up interpolation process, our approach is able to deliver nearly the same interpolation results as that of the original CGI as expected. To be more specific, the average PSNR yielded by the original CGI (with IED) and our fast CGI (with CED) over the above-mentioned 500 test images are 29.49 dB and 29.51 dB, respectively, which are extremely close to each other.

3.4 Experimental Results

Extensive simulation experiments have been conducted to evaluate the interpolation performance of the proposed *fast* CGI algorithm with incorporation of our developed fast edge diffusion scheme, CED. The obtained simulation results are compared with that of the bicubic interpolation and five state-of-the-art methods. Experiments are performed on 8 commonly-used test images with pixel resolutions ranging from 256×256 to 512×768 , as shown in Fig. 3.3.

To determine the kernel width (i.e., filter length) N and the standard deviation σ

Figure 3.4: Some PSNR results of our proposed *fast* CGI algorithm averaged on 8 commonly-used test images using different settings of the filter length N and the standard deviation σ of the Gaussian kernel.

of the Gaussian kernel, the influence of these two parameters on the interpolation performance, measured in peak signal-to-noise ratio (PSNR) in dB, is investigated and presented in Fig. 3.4. One can see that the best performance is achieved when $\sigma = 1.5$, 2.0, and 2.5. Furthermore, when $\sigma < 1.5$, the PSNR drops drastically as σ is further decreased, due to insufficient amount of diffusion. On the other hand, when $\sigma > 2.5$, the performance will gradually degrade as σ increases (especially for large N) because the edge information starts being smoothed out. In this work, $N = 7$ and $\sigma = 2.0$ are empirically determined and used in all our experiments.

For each test image, its ‘LR version’ is first obtained by directly down-sampling the original test image by a factor of 2×2 *without* performing anti-aliasing low-pass filtering. The obtained LR image is then interpolated with an up-scaling factor of 2×2 using the previously-mentioned methods for comparison. The peak signal-to-noise ratio (PSNR) and the structural similarity (SSIM) index [71] are measured against the original image (viewed as the ground-truth HR image) to quantitatively evaluate the quality of the interpolated image. The obtained PSNR and SSIM results are documented in Table 3.1

TABLE 3.1: AVERAGE PSNR (IN dB) RESULTS. THE HIGHEST PSNR FOR EACH TEST IMAGE ARE HIGHLIGHTED IN BOLDFACE.

Images	Bicubic	LMMSE [27]	SAI [28]	ScSR [73]	NARM [81]	CGI [32]	Proposed
<i>Station</i>	24.65	25.07	25.94	25.90	26.14	26.25	26.37
<i>Cameraman</i>	25.26	25.55	25.77	25.92	25.77	25.76	25.88
<i>Wheel</i>	19.59	19.64	21.53	21.48	20.75	22.37	22.37
<i>Plane</i>	32.05	32.48	32.98	33.06	33.21	32.67	32.68
<i>Lena</i>	34.24	34.27	34.91	34.70	35.27	34.67	34.66
<i>PPT</i>	26.77	26.79	26.26	27.46	27.68	26.77	27.05
<i>Butterfly</i>	26.17	26.44	27.40	27.75	27.31	27.65	27.68
<i>Motorbike</i>	27.27	27.38	28.33	28.18	28.39	28.05	28.06
<i>Average</i>	27.00	27.20	27.89	28.06	28.07	28.02	28.09

TABLE 3.2: AVERAGE SSIM RESULTS. THE HIGHEST SSIM FOR EACH TEST IMAGE ARE HIGHLIGHTED IN BOLDFACE.

Images	Bicubic	LMMSE [27]	SAI [28]	ScSR [73]	NARM [81]	CGI [32]	Proposed
<i>Station</i>	0.8928	0.9028	0.9160	0.9160	0.9215	0.9213	0.9218
<i>Cameraman</i>	0.8649	0.8692	0.8732	0.8751	0.8777	0.8720	0.8736
<i>Wheel</i>	0.7723	0.7686	0.8415	0.8396	0.8308	0.8619	0.8601
<i>Plane</i>	0.9218	0.9243	0.9274	0.9288	0.9314	0.9265	0.9268
<i>Lena</i>	0.9020	0.9019	0.9070	0.9065	0.9145	0.9042	0.9037
<i>PPT</i>	0.9499	0.9540	0.9531	0.9600	0.9659	0.9542	0.9559
<i>Butterfly</i>	0.9507	0.9531	0.9621	0.9631	0.9634	0.9635	0.9635
<i>Motorbike</i>	0.8674	0.8660	0.8844	0.8823	0.8865	0.8810	0.8815
<i>Average</i>	0.8902	0.8925	0.9081	0.9089	0.9115	0.9106	0.9109

and Table 3.2, respectively, from which it can be observed that the ScSR, the NARM, the original CGI, and the proposed method achieve significantly better results than the other three methods (i.e., bicubic, LMMSE, and SAI). The overall performance of these four methods are fairly close to one another in terms of the average PSNR and the SSIM measurements.

For conducting subjective performance evaluation, a small portion of the interpolated images obtained from two test images “Wheel” and “Motorbike”, are shown in Fig. 3.5 and Fig. 3.6, respectively.

It can be observed that although the bicubic interpolation is simple and easy to

Figure 3.5: Visual comparison of the image interpolation results obtained from the test image “Wheel”.

Figure 3.6: Visual comparison of the image interpolation results obtained from the test image “Motorbike”.

implement, it generates fairly distinct artifacts. Even for state-of-the-art methods, such as LMMSE, SAI, ScSR, and NARM, they still produce some noticeable artifacts, as can be observed in Figs. 3.5 (c)-(f) and Figs. 3.6 (c), (e), (f). In contrast, the two methods under the CGI framework, i.e., the original CGI [32] and the proposed fast CGI, outperform other methods on the interpolated image quality, and the proposed *fast* CGI method sometimes delivers even slightly better subjective quality than the original CGI, as demonstrated in Fig. 3.6 (g) and (h).

Among the methods under comparison, the NARM algorithm has delivered better PSNRs and SSIMs than our proposed method in some cases. In the NARM method, each up-scaled image patch is generated via a non-local regularized sparse reconstruction process. The performance of such process is highly dependent on whether similar image patches of the processed patch can be found in the input image or among the training images. If yes, the NARM could outperform our proposed method. A closer examination shows that this often happens at those edges oriented in-between the four possible interpolation directions (i.e., 45° , 135° , 0° , and 90°) of our methods. An example is demonstrated in Fig. 3.7, where the edge is oriented between 0° and 45° . In this case, neither the 0° nor 45° 1D cubic spline filter used in our method can deliver satisfying result. On the other hand, the NARM algorithm is not affected by the edge direction since it does not use any directional interpolation filter. Hence, it could perform much better than our proposed method in such cases.

The interpolation framework employed in the original CGI and the proposed fast CGI algorithms only supports a scaling factor of 2, but a larger scaling factor in the power of 2 can be achieved by multiple executions of the 2×2 interpolation sequentially (e.g., twice for achieving a 4×4 interpolation), as mentioned in the original CGI work. In Fig. 3.8, a subjective comparison between the interpolated images with a scaling factor of 4 obtained using different methods are demonstrated. The interpolation is directly conducted on the

Figure 3.7: Visual comparison of the image interpolation results obtained from the test image “Plane”.

Figure 3.8: Visual comparison of the image interpolation results obtained from the test image “Butterfly” using a scaling factor of 4.

input image without down-sampling to avoid significant aliasing artifacts. One can see that the proposed algorithm is still able to keep the sharpness of the edges under such large scaling factor. Other state-of-the-art methods, such as SAI, NARM, and CGI, have delivered very similar results to ours, but with considerably higher computational cost as to be discussed later.

Besides the superior performance, the most appealing merits of the proposed method are its high computational efficiency and algorithm simplicity. The proposed method was

TABLE 3.3: TOTAL RUN TIME (IN SECONDS) FOR INTERPOLATING 8 TEST IMAGES

Bicubic	LMMSE [27]	ScSR [73]	NARM [81]	CGI [32]	Proposed
0.25	118.40	2727.04	4317.99	3.10	0.87

implemented in Matlab and compared with other five state-of-the-art methods that have available Matlab source codes. These codes were run on the same machine with a 2.50 GHz CPU and 16 GB memory. The total run time for interpolating all 8 test images by each method is documented in Table 3.3. One can see that our proposed fast CGI only takes 0.87 seconds, which is drastically faster than the other methods. Comparing with the original CGI method, the total run time of our proposed fast CGI is only about 1/4 of the original CGI's on average. By considering the total run time of the edge diffusion process alone, our proposed CED only takes 0.23 seconds for processing these 8 test images, which is about 1/10 of the time consumed by the IED (in 2.48 seconds).

3.5 Summary

In this chapter, a much more efficient *convolutional edge diffusion* (CED) algorithm is developed for replacing the CGI's *iterative edge diffusion* (IED) so that a fast version of the CGI algorithm can be realized. Compared with the IED, the CED only requires about 1/10 of its computation time, since only *one* convolution operation is needed to conduct diffusion in our proposed method. The resulted *fast* CGI algorithm is able to deliver almost the same interpolated image quality as that of the CGI, while only taking about 1/4 of the CGI's total running time. This makes our *fast* CGI algorithm highly attractive in those applications that demand real-time performance and low power consumption.

Appendix

In this appendix, the detailed derivation of the solution of the heat equation is provided. The heat equation is a parabolic partial differential equation that describes the distribution of heat (or variation in temperature) in an isolated and uniform material over time [83]. For simplicity, we only consider the 1-D case of the heat equation and the derivation can be easily generalized to the 2-D case.

Let $u(x, t)$ be the temperature in a uniform and isolated rod at position x and time t , and $u(x, 0)$ be the initial temperature distribution. Apparently, heat will flow from hotter regions to colder regions as time goes by, and such process can be modeled by the *heat equation* as below:

$$\frac{\partial u}{\partial t} = \nabla^2 u, \quad (3.9)$$

where $\nabla^2 u = \partial^2 u / \partial x^2$ is the Laplacian of u . At each time t , we can consider the $u(x, t)$ as a function of x only, and its Fourier representation is given by

$$u(x, t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} e^{ikx} \hat{u}(k, t) dk, \quad (3.10)$$

where $\hat{u}(k, t)$ is the Fourier transform of $u(x, t)$, which is defined as

$$\hat{u}(k, t) = \int_{-\infty}^{\infty} e^{-ikx} u(x, t) dx. \quad (3.11)$$

Based on (3.10), the $\partial u / \partial t$ and $\nabla^2 u$ in (3.9) can be respectively written as

$$\frac{\partial u}{\partial t} = \frac{1}{2\pi} \int_{-\infty}^{\infty} e^{ikx} \frac{\partial \hat{u}(k, t)}{\partial t} dk, \quad (3.12)$$

$$\nabla^2 u = \frac{\partial^2 u}{\partial x^2} = \frac{1}{2\pi} \int_{-\infty}^{\infty} \frac{\partial^2 e^{ikx}}{\partial x^2} \hat{u}(k, t) dk = \frac{1}{2\pi} \int_{-\infty}^{\infty} -k^2 e^{ikx} \hat{u}(k, t) dk. \quad (3.13)$$

Substitute (3.12) and (3.13) into (3.9), we can get that

$$\frac{\partial \hat{u}(k, t)}{\partial t} = -k^2 \hat{u}(k, t). \quad (3.14)$$

This is actually an ordinary differential equation of $\hat{u}(k, t)$ with respect to t . It bears one of the most basic forms of the ordinary differential equations. After incorporating the boundary condition $\hat{u}(k, 0)$, its solution is simply given by

$$\hat{u}(k, t) = e^{-k^2 t} \hat{u}(k, 0). \quad (3.15)$$

By substituting (3.15) into (3.10), we can arrive at

$$u(x, t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} e^{ikx - k^2 t} \hat{u}(k, 0) dk. \quad (3.16)$$

Since $\hat{u}(k, 0) = \int_{-\infty}^{\infty} e^{-iks} u(s, 0) ds$ according to (3.11)¹, we can further get

$$\begin{aligned} u(x, t) &= \frac{1}{2\pi} \int_{-\infty}^{\infty} e^{ikx - k^2 t} \int_{-\infty}^{\infty} e^{-iks} u(s, 0) ds dk \\ &= \frac{1}{2\pi} \int_{-\infty}^{\infty} u(s, 0) \int_{-\infty}^{\infty} e^{ik(x-s) - k^2 t} dk ds \\ &= \frac{1}{2\pi} \int_{-\infty}^{\infty} u(s, 0) e^{-\frac{(x-s)^2}{4t}} \int_{-\infty}^{\infty} e^{-\left(k\sqrt{t} - \frac{i(x-s)}{2\sqrt{t}}\right)^2} dk ds \\ &= \frac{1}{2\pi} \int_{-\infty}^{\infty} u(s, 0) e^{-\frac{(x-s)^2}{4t}} \frac{\sqrt{\pi}}{\sqrt{t}} ds \quad (\text{arrived based on } \int_{-\infty}^{\infty} e^{-k^2} dk = \sqrt{\pi}) \\ &= \frac{1}{\sqrt{4\pi t}} \int_{-\infty}^{\infty} u(s, 0) e^{-\frac{(x-s)^2}{4t}} ds \end{aligned} \quad (3.17)$$

One can easily see that the obtained solution in (3.17) is essentially a convolution between $u(x, 0)$ (i.e., the initial temperature distribution) and a Gaussian function $\frac{1}{\sqrt{4\pi t}} e^{-\frac{x^2}{4t}}$ with mean $\mu = 0$ and standard deviation $\sigma = \sqrt{2t}$.

¹Note that here the s is a dummy variable for the integral, not to be confused with the x in (3.16).

Chapter 4

Image Sharpness Enhancement via Blurriness-guided Unsharp Masking

4.1 Introduction

Unsharp masking is one of the most widely-used image enhancement techniques for improving the sharpness of image. It has become an indispensable image processing tool in digital photo editing and is often a built-in processing module in digital cameras, printers, and displays [85]. The UM enhancement framework can be summarized as a two-stage process. In the first stage, the input image is decomposed into two layers by applying a linear shift-invariant *low-pass* filter (e.g., Gaussian filter) to the input image under enhancement. The resultant filtered image is called the *base layer*, which contains the main structure of image, while the difference between the base layer and the original image is called the *detail layer*, which contains the fine details of image. Another common implementation of the conventional UM method is to apply a linear shift-invariant *high-pass* filter (e.g., Laplacian filter) to extract the detail layer first, followed by subtracting it from the original image to generate the base layer. In the second stage, the detail layer obtained in the first stage is amplified by multiplying a scaling factor and then added back to the base layer to generate the enhanced image.

In the conventional UM, a *fixed* scaling factor is employed for amplifying the detail

layer. That is, all the pixels on the detail layer are treated *equally* and thus receive the same amount of enhancement. One can intuitively sense that such fixed scaling practice is not always appropriate, because the contents of image are quite dynamic and oftentimes even carry high-level semantic meaning that would not be captured by low-level pixel-based processing. This motivates us to develop an *adaptive* scaling method so that each pixel will have a different scaling factor for receiving various amount of enhancement, including no enhancement at all.

To compute the scaling factors, image characteristics or attributes extracted from the input image can be utilized as the guidance information. Extensive studies have been conducted along this direction in the past, and most of the existing methods utilize the local *contrast* as the attribute to adapt the enhancement strength for each pixel. However, our studies have shown that these methods could lead to under-enhancement (i.e., insufficient amount of enhancement) and over-enhancement across various regions on the enhanced image. The former is often incurred for low-contrast details, while the latter could lead to unwanted amplification of the ‘background’ noise in the blurred regions and overshoot artifacts incurred in the sharp regions with distinct edges. (More details will be discussed in Section 4.5.) In this chapter, the information regarding image’s *blurriness* is measured pixel-wise and used to guide the computation of the scaling factor for each pixel. The consideration of using blurriness is motivated by the following key observations.

First, we have observed that enhancing a highly-blurred image region often leads to unsatisfactory enhancement results with unpleasant artifacts on the enhanced image; for example, the ‘background’ noise tends to be unwelcomely amplified and could even appear more distinct than the image details of the main object as demonstrated in Fig. 4.1. Second, even if the given highly-blurred image region is noise-free, the degradation of its image details could be too severe to be enhanced by applying any image enhancement

Figure 4.1: Enhancement results of image regions at different blur levels by using the conventional UM approach. One can see that enhancing the highly-blurred patch (blue-framed) on the background leads to distinct amplified granular noise, while enhancing the less-blurred patch (red-framed) gives more visually desirable result. However, these two patches have extremely similar local contrast measurement (i.e., the average gradient magnitudes of both patches are approximately 5 intensity levels); this means that it is difficult to distinguish them by the local contrast information.

technique. In this case, a proper image *restoration* (rather than *enhancement*) method should be deployed instead. Lastly, the blurred regions could be purposely created by the photographer in the first place in order to highlight the main object and create aesthetic effect, such as the background region as shown in Fig. 4.1. In this case, it becomes totally undesirable to enhance those blurred regions.

Based on these observations, it is highly convincing to us that the image’s *blurriness* information should be utilized to guide the image enhancement process such that those highly-blurred regions are less enhanced, or even receive no enhancement at all. To accomplish this objective, it is critical to first conduct pixel-wise blurriness estimation for the image under enhancement, and then utilize this information to adapt enhancement strength for each pixel. However, existing blurriness estimation approaches are unable to deliver such pixel-wise estimation with high accuracy as required by our enhancement task. To tackle this problem, an effective blurriness estimation approach is proposed

to generate the blurriness map via two stages as follows. In the first stage, an existing blurriness estimation method, called the *just noticeable blur estimation* (JNBE) [86], is exploited to generate the *initial* blurriness estimation. In the second stage, the *weighted least square* (WLS) [87] estimation is exploited to refine the initial estimation and generate the final blurriness map. Each pixel’s blurriness measurement on the map is then translated into a scaling factor via a mapping process based on the *generalized Gaussian function*. Extensive simulation results obtained using various test images have clearly demonstrated that our proposed blurriness-guided UM method is able to deliver superior perceptual quality of the enhanced image compared with that by using the conventional UM and other existing adaptive UM methods. Specifically, the proposed method works best for images with in-focus and out-of-focus regions adjacent to each other, which are very common among the images taken by people.

The rest of this chapter is organized as follows. Section 4.2 provides a succinct review of existing unsharp masking algorithms and blurriness estimation approaches. Section 4.3 describes the proposed pixel-wise blurriness estimation approach. The proposed blurriness-guided UM framework is presented in Section 4.4. Performance evaluations are provided in Section 4.5. Conclusions are drawn in Section 4.6.

4.2 Literature Review

Existing Unsharp Masking Methods Extensive investigations have been conducted in the past for improving the conventional UM’s performance (e.g., [34]–[43], [45], [46])—mainly addressing the issues of under-enhancement and over-enhancement (accompanying with artifacts). Generally speaking, the improvements proposed in these advanced UM techniques could come from two aspects: 1) more sophisticated layer-decomposition filter for generating the base layer and the detail layer; and 2) adaptive scaling factor for

amplifying the extracted details. These attempts are highlighted in the following.

For the layer-decomposition filtering, various *edge-preserving* filters (e.g., [40]–[43], [45], [46]) have been developed to replace the conventional UM’s linear shift-invariant low-pass filter, and they have been found to be fairly effective on avoiding the overshoot artifacts caused by the *over*-enhancement on those distinct edges of the input image. When such filters are applied to the input image, those distinct edges will be largely preserved in the resultant filtered image (i.e., the base layer). This means that the extracted details (i.e., the differences between the base layer and the original image) at the corresponding pixel locations on the detail layer will be quite small. As a result, less enhancement will be imposed at these pixels when a fixed scaling factor is employed for amplifying the detail layer. For another implementation of the conventional UM method using a high-pass filter to extract the detail layer as mentioned in Section 4.1, several *nonlinear* polynomial filters (e.g., [35]–[37]) have been proposed to replace the conventional linear shift-invariant high-pass filter. These polynomial filters can be viewed as a multiplication of a linear Laplacian filter and a weighting term computed from the input image, which could be the local intensity mean [35], the square of local gradients [37], or the product of both [36]. With the weighting term, these filters will extract less details at those dark regions (with low intensity mean) and/or those flat regions (with low gradient magnitude), where the noise tends to occur. Consequently, they are able to alleviate the noise amplification issue in many cases.

For the design of adaptive scaling factor, several pixel-classification-based methods (e.g., [34], [38]) have been proposed, in which the pixels on the input image are classified into different classes and each class is assigned with a fixed scaling factor. For example, the method presented in [34] utilizes the *local variance* computed from the input image to classify the pixels into three classes—i.e., low-, medium-, or high-contrast pixels. For the assignment of a fixed scaling factor for each class, the low- and high-contrast classes

are assigned with smaller scaling factors than that of the medium-contrast class, since it has been observed that enhancing the pixels from these two classes often leads to noise amplification (for low-contrast case) and overshoot artifacts (for high-contrast case), respectively. In [38], a similar *contrast*-based pixel classification scheme is also developed, which is based on quantizing the filtered result of the input image using the *Laplacian of Gaussian* (LoG) filter. Unlike most existing UM methods that require all scaling factors to be greater than or equal to one, this method allows the scaling factor to be less than one for those pixels in flat regions such that the image details presented by these pixels, which are noise in many cases, will be suppressed, rather than being enhanced.

In addition to the above-mentioned methods, other advanced UM techniques have attempted to improve image enhancement performance through both layer-decomposition stage and detail-layer's scaling stage. For example, in [88], an adaptive bilateral filter is developed for conducting the layer decomposition, together with an adaptive scaling scheme similar to the one as proposed in [38]. More recently, a generalized UM algorithm is proposed in [39] that exploits an edge-preserving iterative median filter in the layer-decomposition stage. Besides, the scaling factor at each pixel is computed based on the amplitude of the extracted detail, such that the larger the detail amplitude, the smaller the scaling factor to be imposed. These improvements make this method able to alleviate the aforementioned overshoot artifacts.

We have observed that almost all of the above-mentioned UM methods exploit the *local contrast* information to guide the enhancement process, despite that various ways have been attempted to measure the local contrast information, such as gradients, local variance, response of LoG filter as aforementioned. However, it has been found that most of these methods are unable to completely avoid the issues that the conventional UM suffers from, and for those can, they often cause the under-enhancement issue for the low-contrast details. In other words, the local contrast, in fact, is not an effective guidance

information for conducting image enhancement. In this chapter, a more effective image attribute—*blurriness*, is exploited for conducting image enhancement instead.

Existing Blurriness Estimation Methods In order to utilize the blurriness information to guide the enhancement process, the input image is subjected to conduct a blurriness estimation process for each pixel on it. There are two types of blurs existing in natural images—i.e., *defocus blur* and *motion blur*. Further consider that the objects being captured by a photographer’s camera are normally static in most cases. Hence, we limit our concern to defocus blur in this chapter, without considering the motion blur. Among the existing defocus blur estimation methods (e.g., [86], [89]–[92]), a blurred image is often mathematically modeled as the convolution of the ideal all-in-focus image with a blur kernel (i.e., a point spread function), which is typically assumed to be a spatially-varying disc function [92] or the Gaussian function [86], [89]–[91]. In this case, the pixel-wise blurriness estimation can be boiled down to estimating the parameters of the blur kernel (e.g., the radius of disc function or the standard deviation of the Gaussian function).

In [89], the first- and second-order image derivatives are utilized to estimate the edge location and the amount of blur. The method proposed in [90] exploits the so-called *local contrast priors* to infer the blurriness at each pixel, followed by applying a Markov random field based propagation process to refine the obtained blurriness map. In [91], the amount of blur is estimated from the gradient ratio computed between the input image and its Gaussian blurred version. More recently, a sparse-representation-based blurriness estimation approach, called the *just noticeable blur estimation* (JNBE) [86], is developed that exploits the sparseness measured from an image patch’s sparse representation to infer its blurriness. This method has shown state-of-the-art blurriness estimation performance among all existing blurriness estimation algorithms. However,

all the aforementioned blurriness estimation methods are unable to deliver the estimation accuracy that is required by our blurriness-guided image enhancement task. Therefore, a new blurriness estimation method is proposed and described in the following section.

4.3 Proposed Pixel-wise Blurriness Estimation

For our image enhancement task, the computed pixel-wise blurriness map, denoted by M , is required to possess two essential properties as follows. First, for each region under a fixed focus level as presented on the input image I , the estimated blurriness for all pixels within this region should be a constant (or, at least, very close to each other numerically). In other words, this region should be perceived as a (nearly) uniform or flat region on M . Second, a sharp and accurate boundary should be yielded between two adjacent uniform regions (on M) that have sufficiently different focus levels (on I). Multiple blurriness estimation methods have been experimented (i.e., [86], [91]–[93]), but all of them failed to produce a blurriness map that could possess these two properties. To address this problem, a novel pixel-wise blurriness estimation algorithm is proposed in this chapter, which consists of the following two sequential stages: 1) an initial blurriness map $\widehat{M}$ is first generated using the JNBE algorithm [86], and 2) the *weighted least square* (WLS) estimation method is then used to refine $\widehat{M}$ to get the final accurate blurriness map M . These two stages will be described in detail in the following sub-sections, respectively.

4.3.1 Stage 1: Initial Blurriness Estimation

In our approach, the JNBE algorithm [86] is employed for generating the initial blurriness map $\widehat{M}$, simply because of its state-of-the-art performance. For a color input image I , the initial blurriness map $\widehat{M}$ will be estimated based on its luminance channel recorded in the CIELab color space. The JNBE algorithm assumes that the blur kernel is a spatially-varying Gaussian kernel and its standard variation can be used as the

estimated blurriness. This sparse-representation-based algorithm needs to establish an over-complete dictionary in prior using a large set of Gaussian-blurred training image patches. With such obtained dictionary, the blurriness estimation is conducted based on the key observation that the sparse representation of a clear image patch tends to be less sparse than that of a blurred one. The rationale behind this is that each ‘atom’ (i.e., basis vector) of the dictionary contains only smooth structures as the dictionary is learned from Gaussian-blurred patches. Consequently, a clear patch (often containing sharp transitions) will generally require more dictionary atoms to represent it; thus, yielding a less-sparse representation.

Based on this observation, the blurriness of an image patch $y \in \mathbb{R}^{n \times 1}$ with n pixels is inferred from the sparseness of its sparse representation coefficient $x \in \mathbb{R}^{k \times 1}$, which is obtained by solving

$$x = \arg \min \|x\|_0, \quad \text{subject to } \|y - Dx\|_2 \leq \epsilon, \quad (4.1)$$

where $D \in \mathbb{R}^{n \times k}$ is the aforementioned dictionary established from a set of Gaussian-blurred patches, which has k atoms (basis vectors). Symbols $\|\cdot\|_0$ and $\|\cdot\|_2$ denote the l_0 and l_2 norms of a vector, respectively. The constant ϵ forces Dx to be close to y ($\epsilon = 0.07$ was used in [86]). After x is obtained, its sparseness is measured by $\|x\|_0$ (i.e., the number of non-zero elements in x). On the other hand, the blurriness of y is measured by the standard deviation σ of its underlying Gaussian blur kernel as mentioned previously. According to the above-described observation, the larger the $\|x\|_0$, the smaller the σ , roughly speaking. More precisely, based on a large set of training image patches with ground-truth σ , it has been found in [86] that the relationship between $\|x\|_0$ and σ approximately fits a logistic regression model, which is expressed as: $\|x\|_0 = f(\sigma)$. Therefore, given the sparse representation coefficient x of an image patch y , its estimated blurriness $\bar{\sigma}$ can be simply computed as $\bar{\sigma} = f^{-1}(\|x\|_0)$. For each pixel

p on the input image, the above-mentioned blurriness estimation needs be conducted on a 8×8 patch surrounding p , yielding its corresponding blurriness estimation $\bar{\sigma}_p$. Consequently, a blurriness map $\widehat{M}$ of the same size of the input image I , where $\widehat{M}_p = \bar{\sigma}_p$, will be generated.

However, the generated $\widehat{M}$ is very coarse (as demonstrated in Fig. 4.2 (b)) and far from satisfactory for being used to guide our image enhancement task. Specifically, it has two major problems as follows. First, since the JNBE estimation is essentially conducted in a patch-wise manner, it fails to yield accurate boundaries with pixel-wise accuracy among adjacent objects at different focus levels, as can be observed from Fig. 4.2 (b). As a result, some background pixels will be enhanced in the enhancement stage, and this could easily introduce artifacts surrounding the object's boundary on the enhanced image. Second, when a clear patch is highly smooth, the JNBE method will mistakenly yield a high blurriness value for it, leading to 'white' regions within a clear object (which is supposed to appear as a whole dark region) in the generated $\widehat{M}$, as illustrated in Fig. 4.2 (b). This could cause under-enhancement issues for the low-contrast image details within such 'white' regions. To address these issues, a refinement step of $\widehat{M}$ is clearly needed.

4.3.2 Stage 2: Blurriness Map Refinement

In this work, the *weighted least square* (WLS) estimation method [87] is adopted to refine the coarse $\widehat{M}$ for generating a much improved blurriness map, denoted by M . Specifically, M is obtained by minimizing a cost functional $\mathcal{F}$ that is defined as [87]

$$\mathcal{F}(M) = \sum_p \left\{ w_p \left(M_p - \widehat{M}_p \right)^2 + \lambda \left[h_p (\partial_x M)_p^2 + v_p (\partial_y M)_p^2 \right] \right\}, \quad (4.2)$$

where the subscript p denotes the pixel location. The partial differentiation operators ∂_x and ∂_y compute the derivatives along the horizontal (i.e., x -axis) and the vertical (i.e., y -

Figure 4.2: Results of various blurriness map refinement schemes: (a) original image; (b) the initial blurriness map $\widehat{M}$ generated by using the JNBE algorithm [86]; (c) the confidence weightage map computed according to (4.3); (d) the refined map M by using the bilateral filter [41] with $\sigma_s = 5$ and $\sigma_r = 0.1$; (e) the refined map M obtained by using the WLS-based approach without the confidence weights; (f) the refined map M obtained by using the proposed WLS-based approach with the adaptive confidence weightage map as shown in (c).

axis) directions on M , respectively. The first term of (4.2) is the *data* term, which is used to prevent M from being deviated away from the initial map $\widehat{M}$ drastically. Meanwhile, the second term of (4.2) is the *smoothness* term, which forces M to be as smooth as possible. The constant λ provides a trade-off between these two terms, and $\lambda = 0.2$ is empirically set in this approach.

The *confidence weight* w_p in (4.2) indicates how confident of the initial blurriness estimation $\widehat{M}_p$ obtained at the pixel p . The main objective of introducing this weighting factor is to discard those unconfident (thus, unreliable) initial blurriness estimations by setting their corresponding weighting factors to zero. By doing so, the data term as in (4.2) will take no effect (i.e., no fidelity constraint) at the corresponding pixels, thus their

new (refined) estimated blurriness values will be highly close to those of their neighboring pixels due to the effect of the smoothness term. Similar weighting strategy has been practiced in some depth map upsampling and enhancement works (e.g., [53], [94]). Now, the issue is boiled down to how to identify these pixel locations. As previously mentioned, the JNBE algorithm often mistakenly yields such ‘white’ pixels within clear objects on the $\widehat{M}$. Fortunately, the number of such ‘white’ pixels are usually quite small according to our observations. To identify these locations, the mean shift [95] segmentation algorithm is adopted for this task and conducted on $\widehat{M}$. Based on the segmentation results, w_p is determined as follows:

$$w_p = \begin{cases} 0, & \widehat{M}_p > \sigma_t \text{ and } size(R_p) < N_t; \\ 1, & \text{otherwise,} \end{cases} \quad (4.3)$$

where R_p denotes the segmented region containing the pixel p . Thresholds σ_t and N_t are empirically determined and used throughout all our experiments as follows: $\sigma_t = 1.8$ and $N_t = 2.5\% \cdot N$, where N is the total number of pixels of the input image I . Based on (4.3), $w_p = 0$ will be assigned to the pixel p , if its corresponding initial blurriness estimation $\widehat{M}_p$ is greater than σ_t and meanwhile it belongs to a small segmented region containing less than N_t pixels. An example of the confidence weightage map computed via (4.3) is demonstrated in Fig. 4.2 (c), from which one can see that the ‘white’ regions in Fig. 4.2 (b) have been successfully detected (via mean shift) and assigned with zero weights.

Furthermore, the *smoothness weights* h_p and v_p in (4.2) are imposed to adapt the amount of smoothing at each pixel location. They are determined by the input image I based on its gradients along the horizontal direction (i.e., x -axis) and vertical direction (i.e., y -axis); i.e., $\partial_x I$ and $\partial_y I$, respectively:

$$h_p = (|(\partial_x I)_p|^\gamma + \kappa)^{-1} \text{ and } v_p = (|(\partial_y I)_p|^\gamma + \kappa)^{-1}, \quad (4.4)$$

where κ is a small constant number imposed for avoiding the divided-by-zero incidence,

and the exponent γ controls the sensitivity of h_p and v_p to the above-mentioned gradients $(\partial_x I)_p$ and $(\partial_y I)_p$, respectively. In our approach, $\kappa = 10^{-5}$ and $\gamma = 2$ are used. According to (4.4), the smoothness weights will be small for those pixel locations that incur large gradients on I , thus imposing less amount of smoothing on $\widehat{M}$. This has the effect of preserving those edges on $\widehat{M}$ that correspond to sharp object boundaries on I .

By using the matrix notation, the objective functional as defined in (4.2) can be re-written as follows:

$$\mathcal{F}(M) = \mathcal{J}(V_M) = (V_M - V_{\widehat{M}})^T W (V_M - V_{\widehat{M}}) + \lambda (V_M^T D_x^T A_h D_x V_M + V_M^T D_y^T A_v D_y V_M), \quad (4.5)$$

where V_M and $V_{\widehat{M}}$ are the vector form of M and $\widehat{M}$, respectively. Matrices D_x and D_y are the discrete (forward) differentiation operators ∂_x and ∂_y in the Toeplitz matrix form, respectively. Matrix W is a diagonal matrix containing the confidence weight w_p . Similarly, matrices A_h and A_v are also diagonal matrices, which contain the smoothness weights h_p and v_p , respectively. The vector V_M that minimizes the above objective functional $\mathcal{J}(V_M)$ can be derived by letting $\partial \mathcal{J} / \partial V_M = 0$, as follows:

$$\frac{\partial \mathcal{J}}{\partial V_M} = W (V_M - V_{\widehat{M}}) + \lambda (D_x^T A_h D_x + D_y^T A_v D_y) V_M = 0 \quad (4.6)$$

$$\Rightarrow (W + \lambda D_x^T A_h D_x + \lambda D_y^T A_v D_y) V_M = W V_{\widehat{M}} \quad (4.7)$$

$$\Rightarrow V_M = (W + \lambda D_x^T A_h D_x + \lambda D_y^T A_v D_y)^{-1} W V_{\widehat{M}}. \quad (4.8)$$

The equation in (4.8) describes a sparse linear system, which can be solved using the *preconditioned conjugate gradients* approach [96]. Several more efficient solvers are also available (e.g., [47], [97]). Finally, the obtained optimal vector V_M is converted back to the matrix form to yield the final refined blurriness map M .

It is worthwhile to mention that the bilateral filter [41] was originally suggested in the JNBE [86] for refining the initially-estimated blurriness map $\widehat{M}$. However, one can

Figure 4.3: The processing pipeline of the proposed blurriness-guided adaptive unsharp masking, conducted on the luminance component L , recorded in the CIELab color space.

see from Fig. 4.2 (d) that the bilateral filter delivered a fairly poor refinement result M —that is, it failed to filter out the noisy structures and to yield sharp region boundaries. Other local smoothing filters, such as the guided filter [42] and the domain transform filter [43], have also been investigated, and they all produce similar results (not presented here) to that of the bilateral filter according to our simulations. On the other hand, the WLS-based approach is able to deliver a much better refined map M , even without the use of the confidence weighting (see Fig. 4.2 (e)). With further incorporating confidence weights, the final refined blurriness map is fairly smooth and meets our initially imposed goals about what the final blurriness estimation map should look like in order to facilitate the follow-up UM process.

4.4 Proposed Blurriness-guided Unsharp Masking

The framework of the proposed blurriness-guided unsharp masking method is illustrated in Fig. 4.3. If the input image I is a color image, the enhancement will be conducted only on its luminance component L , recorded in the CIELab color space. The enhanced luminance component will then be combined with the original two chrominance components, followed by converting them back to the RGB color space to generate the final enhanced color image for presentation.

In Fig. 4.3, the refined blurriness map M , which is obtained through the blurriness estimation process as described in the previous section, will be utilized as a guidance information to determine the scaling factor (i.e., enhancement strength) for each pixel. The entire enhancement process can be mathematically described as

$$\tilde{L} = B + \Lambda \otimes T, \quad (4.9)$$

where $\tilde{L}$ is the enhanced luminance component, B and T are the *base layer* and *detail layer* decomposed from the input luminance component L , respectively, that is, $L = B + T$. Unlike many existing UM algorithms that employ a *fixed* scaling factor for all pixels on the detail layer T , an *adaptive* scaling factor matrix Λ is used in our approach to scale the detail layer T via pixel-wise multiplication (denoted by $\otimes$).

Since the pixel-wise scaling factor matrix Λ is to be imposed on the detail layer T for enhancement, the layer-decomposition filter conducted in the first stage of the UM process is intimately related to how effective of the matrix Λ to be generated in the second stage. For that, we classify the layer-decomposition filters into two categories: *edge-preserving* filter (e.g., [40]–[43], [45], [46]) and *non-edge-preserving* filter (e.g., Gaussian filter, average filter). The rationale behind such classification is due to the fact that if an edge-preserving filter were used, more edge information will be kept on the base layer B , and therefore the details presented on the detail layer T will have smaller amplitudes. To

yield effective enhancement, larger scaling factors tend to be applied. However, the same scaling factors might be too large to be used for a detail layer that was generated by a non-edge-preserving filter, because more details with large amplitudes tend to appear on the detail layer. Hence, over-enhancement will be incurred that could result in image artifacts.

Since the scaling factor matrix Λ is obtained through a mapping process, two mapping algorithms are needed, one for each aforementioned layer-decomposition filter case. In our approach, if an edge-preserving filter were used in the layer decomposition stage, only the *blurriness* information M , computed from the input image I , will be used as the input of the mapping process (see Fig. 4.3). On the other hand, if a non-edge-preserving filter were used instead, the *contrast* information (computed from the detail layer T) will be incorporated as an extra guidance information on the blurriness information to avoid or alleviate the overshoot artifacts due to the less sophisticated filtering stage. Note that this part is not shown in Fig. 4.3 for simplifying the block-diagram drawing, and it can be easily added into Fig. 4.3 according to our more detailed descriptions in what follows.

4.4.1 Case 1: Computation of Λ for Edge-preserving Filter

As highlighted previously, if an edge-preserving filter (e.g., [40]–[43], [45], [46]) were used in the layer-decomposition stage, only the blurriness information is exploited for the computation of the scaling factor matrix Λ . As explained in Section 4.1, enhancing a highly-blurred image region is undesirable, because it is quite unlikely to yield an appreciable quality improvement, while often getting unwanted artifacts in return. Therefore, it is more appropriate to assign smaller scaling factors to the pixels in those highly-blurred regions to impose less amount of enhancement, or even no enhancement at all. To accomplish this objective, a mapping process based on the *generalized Gaussian function* is proposed to translate the blurriness estimation M_p into a scaling factor Λ_p for each

pixel location p . Our proposed mapping function is defined as

$$\Lambda_p = \rho \cdot \mathcal{G}_b(M_p) + \eta, \quad (4.10)$$

where

$$\mathcal{G}_b(M_p) = e^{-\left(\frac{M_p}{\alpha}\right)^\beta}. \quad (4.11)$$

The parameters α and β determine the *scale* and the *shape* of the function $\mathcal{G}_b(M_p)$, respectively, as shown in Fig. 4.4 (a) under different settings of these parameters. One can see that the scale parameter α determines how wide the range of blurriness will be assigned with large scaling factors, while the shape parameter β controls how fast of the transition changing from higher values of scaling factors to lower ones. On the other hand, since $\mathcal{G}_b(M_p) \in (0, 1]$, the obtained Λ_p will fall into the range of $(\eta, \rho + \eta]$. This means that ρ and η decide the maximum and minimum scaling factors (i.e., the dynamic range) that could be imposed by (4.10). Setting large values to these two parameters will clearly yield more perceivable enhancement; however, this might also cause undesirable over-enhancement artifacts more easily in return.

In practice, there is no optimal setting for the above-mentioned parameters (i.e., α , β , ρ , and η), and their values should be determined based on the requirements of each specific application or up to the user's preference. For our enhancement task conducted on common natural images, it has been found through extensive simulations that using $0.8 \leq \alpha \leq 1.4$ and $4 \leq \beta \leq 8$ generally yields visually pleasant enhancement results. As for ρ , we have found that letting $1 \leq \rho \leq 3$ is able to deliver fairly perceivable improvement on image sharpness, while not yielding the overshoot artifacts in most cases. Finally, η is typically set as $\eta = 1$, in which case no enhancement will be imposed at all if the estimated blurriness M_p is very large, since $\Lambda_p \rightarrow \eta = 1$ when $M_p \rightarrow \infty$. It is worthwhile to point out here that it is also possible to set $\eta < 1$ in (4.11) as practiced in [38] and [88]. This will even suppress some unwanted details (such as noise) presented

Figure 4.4: An illustration of the the proposed mapping functions for generating the adaptive scaling factor matrix Λ : (a) the blurriness-guided mapping function as defined in (4.10) for Case 1, when an edge-preserving filter is used; (b) the blurriness- and contrast-guided mapping function as defined in (4.12) for Case 2, when non-edge-preserving filter is used. Note that $\rho = 3$, $\eta = 1$ are used for generating the above plots, thus $\Lambda_p \in (1, 4]$ in both cases.

in the blurred background. This will be demonstrated in Section 4.5.

An example of the scaling factor matrix Λ generated by the above-described mapping process can be found in Fig. 4.3, from which one can see that relatively small scaling factors have been allocated to those *highly-blurred* regions, as expected. Consequently, such regions will receive less amount of enhancement, or even no enhancement at all. Our extensive simulation results have shown that this effectively avoids the unwanted amplified noise that often appears in those highly-blurred regions.

4.4.2 Case 2: Computation of Λ for Non-edge-preserving Filter

If a *non-edge-preserving* filter (such as the Gaussian filter, the average filter, to name a few) is used for conducting layer decomposition, both *blurriness* and *contrast* information will be jointly exploited for the computation of the scaling factor matrix Λ . The rationale behind the use of contrast information as an additional guidance, besides the blurriness information, is justified with detailed explanations as follows.

The overshoot artifacts are typically introduced due to the enhancement imposed on those strong details incurred on the detail layer T [39]. This could happen, especially when a non-edge-preserving type of low-pass filter is deployed in the layer-decomposition stage, which means that more edge information could be incurred on the detailed layer, since the strong edge information failed to be preserved on the base layer B as much as possible.

As non-edge-preserving filters are preferable in many applications due to their algorithmic simplicity and computational efficiency, the above-mentioned overshoot artifacts must be addressed and prevented. Obviously, working on the detail layer T is the most direct and effective way to pursue. For that, an addition multiplier is proposed and inserted into (4.10) to adaptively prevent overshoot artifacts. That is,

$$\Lambda_p = \rho \cdot \mathcal{G}_b(M_p) \cdot \mathcal{G}_c(|T_p|) + \eta. \quad (4.12)$$

where

$$\mathcal{G}_c(|T_p|) = e^{-\left(\frac{|T_p|}{\mu}\right)^\nu}. \quad (4.13)$$

That is, $\mathcal{G}_c(|T_p|)$ is another generalized Gaussian function similar to $\mathcal{G}_b(|M_p|)$, except that it works in the contrast domain rather than in the blurriness domain. According to (4.13), the larger the value of $|T_p|$, the smaller the $\mathcal{G}_c(|T_p|)$ value to be further multiplied in (4.12). By incorporating this extra term, the scaling factors computed via (4.12) will be smaller compared with the case if (4.10) were used (as $\mathcal{G}_c(|T_p|) \in (0, 1]$), especially at those pixels where the detail amplitude $|T_p|$ are large. This effectively helps to avoid or alleviate the overshoot artifacts.

Similar to the α and β in (4.11), the μ and ν in (4.13) control the shape of $\mathcal{G}_c(|T_p|)$. In this work, μ and ν are empirically determined as $\mu = 1/2(\rho + \eta)$ and $\nu = 3$, respectively. Note that, according to (4.10), Λ_p reaches to its maximum value $\rho + \eta$ as the largest scaling factor, when $\mathcal{G}_b(M_p) = 1$. Hence, μ is set to be inversely proportional to $\rho + \eta$

as a general trend, since the larger the $\rho + \eta$ is, the more likely the overshoot artifacts might be incurred. The extra factor $1/2$ in μ is empirically determined for a more robust enhancement.

Note that a similar scaling factor adaptation scheme can be found in [39], which also utilizes the information presented on $|T|$ and exploits the same computation formula as $\Lambda_p = \rho \cdot \mathcal{G}_c(|T_p|) + \eta$, where $\mathcal{G}_c(|T_p|)$ is defined in (4.13). However, the parameter μ in (4.13) is fixed and set to 1 in [39], while ours is adaptive as the values of ρ and η are adjustable. Furthermore, the scaling scheme in [39] does not involve the blurriness-guided term $\mathcal{G}_b(M_p)$ as ours in (4.12). Consequently, the scaling scheme in [39] is less effective than ours on preventing both noise amplification and overshoot artifacts. All these will be demonstrated in Section 4.5.

Fig. 4.5 demonstrates a set of enhanced images for Case 1 and Case 2, respectively. The representative layer-decomposition filters employed are the edge-preserving bilateral filter for Case 1 and the non-edge-preserving Gaussian filter for Case 2, respectively. Fig. 4.5 conveys some fundamental insights as follows. First, comparing Figs. 4.5 (b) and (c), it can be observed that their amplitude map $|T|$ are quite different, as expected. This clearly justifies why the layer-decomposition filter deployed in the first stage in Fig. 4.3 needs to be considered in the design of mapping algorithm for the computation of the scaling factor matrix. Second, it can be observed from Figs. 4.5 (d) and (e) that the scaling factors computed in Case 2 at those pixels with large detail amplitudes have been effectively reduced compared to their counterparts in Case 1, by incorporating the $\mathcal{G}_c(|T_p|)$ into the mapping function (4.12). Consequently, the enhanced images yielded in both cases, as shown in Fig. 4.5 (g) and (h), have demonstrated proper enhancement without creating artifacts, compared to the original image patch cropped from the original image as shown in Fig. 4.5 (f). In contrast, if the mapping function (4.10) instead of (4.12) is used for Case 2, distinct overshoot artifacts will be yielded as shown in

Figure 4.5: Demonstrations of adaptive scaling factor matrix computation for Case 1 and Case 2: (a) the input image; (b)-(c) the $|T|$ obtained via the Bilateral filter (Case 1) and the Gaussian filter (Case 2), respectively; (d)-(e) the Λ computed for Case 1 and Case 2, respectively; (f) an image patch cropped from (a); (g)-(h) enhanced patches in Case 1 and Case 2, respectively; (i) enhanced patch in Case 2 using the mapping function (4.10) (i.e., without incorporating $|T|$).

Fig. 4.5 (i). All these results have clearly demonstrated the effectiveness and consistence of our adaptive scaling factor design; that is, it is able to deliver satisfactory enhancement results regardless of which type of layer-decomposition filter is deployed in the first stage of the enhancement process.

4.5 Experimental Results

Extensive simulations have been conducted on a variety of test images to evaluate the enhancement performance of the proposed blurriness-guided unsharp masking (BUM) approach. As previously mentioned, our proposed BUM is a general UM framework that can be deployed with various layer-decomposition filters. In this work, we have tested two most representative ones, i.e., the non-edge-preserving Gaussian filter and the edge-preserving bilateral filter [41]. The enhancement results of our approach have been compared to those of the conventional UM [33] and three other adaptive UM methods—

i.e., the *generalized unsharp masking* (GUM)¹, the *adaptive unsharp masking* (AUM) [34], and the *cubic unsharp masking* (CUM) [37]. To the best of our knowledge, there is currently no widely-accepted quantitative performance evaluation metrics for sharpness enhancement methods. Therefore, in this work, we choose to use visual (subjective) comparisons to justify the superiority of our proposed approach.

4.5.1 Parameter Settings

Compared with the conventional UM process, our proposed BUM approach is highly adaptive, not only for each pixel but also for the layer-decomposition filter used. The adaptation is realized by the use of the generalized Gaussian functions whose inputs are the estimated *blurriness* guidance information (Case 1) and additional *contrast* guidance information (Case 2). For the blurriness estimation stage, the default parameter settings previously mentioned in Section 4.3 are used; that is, $\lambda = 0.2$ as in (4.2); $\sigma_t = 1.8$ and $N_t = 2.5\% \cdot N$ (N is the total number of pixels of the input image) as in (4.3); $\kappa = 10^{-5}$ and $\gamma = 2$ as in (4.4).

For the scaling factor computation stage using the generalized Gaussian functions, the common parameters for both Case 1 and Case 2 are set as follows: $\rho = 2.5$, $\eta = 1$ or 0 , $\alpha = 1$, and $\beta = 6$. For Case 2, two extra parameters are required, i.e., μ and ν as in (4.12). As mentioned in Section 4.4.2, they are set as $\mu = 1/2(\rho + \eta)$ and $\nu = 3$, respectively. Note that two settings for η are investigated; i.e., $\eta = 1$ and $\eta = 0$. The former is practiced in most existing adaptive UM methods that ensures the original details on the input image will at least remain unchanged, if not enhanced, while the latter allows the proposed enhancement process to even *smooth* (an opposite to the purpose of *enhancement*) image details, as this might be desirable for suppressing the

¹The original GUM algorithm has an extra contrast enhancement step conducted on the base layer via the histogram equalization (HE) algorithm. However, for a fair comparison, this step has been removed in our simulations.

noise often presented in those highly-blurred background regions.

Throughout our simulations, all the parameters of our proposed algorithm are fixed to their default values as previously described (i.e., not fine-tuned for each test image). However, since image enhancement is a highly subjective task, sometimes fine-tuning of parameters is indispensable in some real applications in order to achieve the desired enhancement effect. Among all the parameters in our algorithm, it has been found that the α and ρ have the most significant impact on the enhanced image. Specifically, α is essentially a threshold value on the blurriness measurements and determines pixels with which range of blurriness will be enhanced. The choice of α is critical especially for those slightly-blurred regions that have middle-valued blurriness measurement. The influence of the enhancement factor ρ is more straightforward; the larger the value of ρ , the more significant the enhancement effect, but meanwhile the more likely the over-enhancement artifacts will occur. In real applications, we believe the above-mentioned key parameters should be open to the users for fine-tuning, rather than being fixed to certain values. In fact, this is what commonly practiced in existing enhancement softwares.

To conduct a fair comparison with the conventional UM and other adaptive UM methods (i.e., the GUM [39], AUM [34], and CUM [37]), the enhancement strength of these methods have been adjusted to match that of ours as controlled by the parameters ρ and η . For example, the fixed scaling factor in the conventional UM method is set as $\rho + \eta$ to make sure its imposed enhancement strength is close to ours, perceptually speaking. Other than that, all the involved parameters in these methods under comparison are set to their originally proposed default values, respectively. In what follows, the performance evaluations of our proposed BUM against these methods in the $\eta = 1$ and the $\eta = 0$ cases will be presented and discussed, respectively.

Figure 4.6: Visual comparison of the enhancement results on the image "old man"; (a) original image; (b) UM with Gaussian filter; (c) GUM with Gaussian filter [39]; (d) AUM with Gaussian filter [34]; (e) CUM [37]; (f) Proposed BUM with Gaussian filter ($\eta = 1$).

Figure 4.7: Visual comparison of the enhancement results on the image “parrot”; (a) original image; (b) UM with Gaussian filter; (c) GUM with Gaussian filter [39]; (d) AUM with Gaussian filter [34]; (e) CUM [37]; (f) proposed BUM with Gaussian filter ($\eta = 1$).

4.5.2 Subjective Evaluations

Fig. 4.6 shows the enhancement results on the test image “old man” produced by the proposed BUM with the Gaussian filter and by other four UM methods under comparison. It can be clearly observed from Fig. 4.6 (b) that the conventional UM method has yielded amplified noise on the highly-blurred region (e.g., a red-color framed background region), as well as overshoot artifacts on the foreground region that has clear details (e.g., a green-color framed). Three *contrast*-based UM methods, GUM, AUM, and CUM, have delivered slightly improved results compared with that of the UM, but they have either failed to avoid all artifacts or caused insufficient amount of enhancement (or under-enhancement) for those image content with low-contrast details. Especially, the GUM algorithm is unable to avoid the unwanted noise amplification often incurred in blurred background regions (see the first close-up in Fig. 4.6 (c)), while the AUM and CUM methods can perform insufficient amount of enhancement for those less-blurred regions with low-contrast details (see the second-row close-ups in Fig. 4.6 (d) and (e), respectively). Clearly, our proposed approach has constantly delivered the best enhancement results among all these methods as shown in Fig. 4.6 (f) in terms of perceptual quality. That is, it is able to prevent both the amplified noise and overshoot artifacts, without causing under-enhancement for those low-contrast image details.

Another illustration of the enhancement results yielded by these methods is shown in Fig. 4.7, which is experimented on the test image “parrot.” The same comparison statements as drawn from Fig. 4.6 can be also obtained from Fig. 4.7, especially from the blue-color framed patch located at the boundary between the blurred and clear regions. Again, one can see that our proposed BUM is the only method that is able to avoid both the noise amplification on the blurred background region while without causing under-enhancement for those low-contrast details within the foreground. It also significantly

alleviates the overshoot artifacts at the foreground-background boundary.

In Fig. 4.8, a comparison of the enhancement results yielded on the image “burger” by the proposed BUM with the bilateral filter and by other two comparable methods—i.e., the UM and the modified AUM², are demonstrated. As expected, the UM with use of the bilateral filter as the layer-decomposition filter has produced more distinct noise in the highly-blurred regions (see the first close-up in Fig. 4.8 (b)). Moreover, the so-called “gradient reversal” artifacts [42] have also been introduced at the edge as shown in the second close-up in Fig. 4.8 (b). This is a typical phenomenon that happens when a blurred edge gets enhanced with the bilateral filter being used for layer decomposition, since the bilateral filter tends to somewhat sharpen those blurred edges [42]. The GUM with bilateral filter has yielded similar noise amplification issue, as shown in Fig. 4.8 (c). On the other hand, the AUM with use of the bilateral filter has alleviated the noise amplification by reducing the scaling factor for those low-contrast regions. However, this is accomplished at the expense of imposing under-enhancement on the regions with low-contrast image details, as can be observed from the third close-up in Fig. 4.8 (d). The CUM has also largely avoided the noise amplification at the first patch, but it yielded visually unnatural results in the second and third patches, as illustrated in Fig. 4.8 (e). Lastly, one can see from Fig. 4.8 (f) that the proposed BUM with bilateral filter has completely avoided all these issues by only enhancing those less-blurred regions that are supposed to be enhanced.

Fig. 4.9 shows the results obtained by our proposed BUM approach with $\eta = 0$ using the bilateral filter. In this case, the scaling factors allocated to the highly blurred regions will be close to 0. According to (4.9), this means that the enhanced image will be close

²The original pixel-classification-based AUM method cannot work with the nonlinear bilateral filter as it involves a scaling factor adjusting step that requires the adopted filter to be linear. To address this issue, such step is discarded and the scaling factor for each classified class is directly allocated to the pixels belong to it; to be specific, $\Lambda_p = 1, 3.5$, and 2 are allocated to the classified low-, medium-, and high-contrast pixels, respectively.

Figure 4.8: Visual comparison of the enhancement results on the test image "burger"; (a) original image; (b) UM with bilateral filter; (c) GUM with bilateral filter [39]; (d) modified AUM with bilateral filter [34]; (e) CUM [37]; (f) proposed BUM with bilateral filter ($\eta = 1$).

Figure 4.9: Visual comparison of the enhancement results on the test image “bear”; (a) original image; (b) UM with bilateral filter; (c) GUM with bilateral filter [39]; (d) modified AUM with bilateral filter [34]; (e) CUM [37]; (f) proposed BUM with bilateral filter ($\eta = 1$).

to the detail-removed base layer in these regions. One can see that our propose BUM is the only method that is able to simultaneously suppress the noise in the highly-blurred background and enhance the details in the less-blurred foreground, while all the other methods under comparison failed to accomplish both at the same time.

However, it should be pointed out that our proposed BUM method using $\eta = 0$ does not work well on very noisy images. Although the BUM could suppress the noise in the blurred regions to some extent, but it tends to produce more distinct amplified noise on the less-blurred regions. In fact, conducting sharpness enhancement is completely undesirable for these images due to the existence of strong noise. Lastly, using $\eta = 0$ sometimes could smooth those desired details if some of the foreground regions are mistakenly assigned high blurriness values during the blurriness estimation process, which typically happens when the details presented in the foreground region are fairly weak. Therefore, $\eta = 1$ is recommended in practice as it is able to deliver more robust enhancement results.

4.5.3 Comparison of Computational Complexity

This sections provides a comparison of the computational complexity of the proposed BUM method and other state-of-the-art UM methods. All these methods are tested using their MATLAB implementations on the same machine with an Intel Core i7-6700 3.4 GHz CPU and 8 GB RAM. The Gaussian filter is employed to conduct the layer decomposition in all methods except the CUM method, which employs a polynomial filter instead. The running times (average of 10 runs) of all methods for enhancing an 800×533 image are documented in Table 4.1.

It can be seen that the proposed BUM method is computationally much more expensive than other UM methods under comparison. A closer examination shows that over 99.8% of the running time is spent on the generation of the blurriness map; to be

TABLE 4.1: AVERAGE RUNNING TIME (IN SECONDS) OF DIFFERENT METHODS FOR ENHANCING AN 800×533 IMAGE

UM	GUM [39]	CUM [37]	AUM [34]	Proposed
0.24	0.27	0.26	0.51	5.27

specific, the generation of the initial blurriness map costs about 3.14 seconds, and the refinement process of the blurriness map costs about 2.12 seconds. Hence, in order to make our proposed algorithm more practical for real-time enhancement applications, it is definitely necessary to investigate more efficient blurriness estimation methods.

4.6 Summary

In this chapter, a novel blurriness-guided adaptive unsharp masking method is proposed that incorporates the blurriness information into the enhancement process. The proposed method is motivated by several key observations which suggest that enhancing those highly blurred image regions is undesirable. Based on such observations, a two-stage blurriness estimation method is developed to generate an accurate pixel-wise blurriness map of each input image, followed by utilizing the obtained blurriness map to impose adaptive enhancement strength at different pixels, such that those highly-blurred regions are less enhanced or not enhanced at all. Extensive simulations on various test images have demonstrated that the proposed method is able to deliver superior perceptual quality of the enhanced image compared with that by using the conventional UM and other existing adaptive UM methods.

Chapter 5

Semantic Image Content Filtering via Edge-preserving Scale-aware Filter

5.1 Introduction

Digital image filtering is an important building block of the digital image processing field, and it has been widely used for many fundamental image processing tasks such as noise reduction, sharpness enhancement, edge detection, to name a few. Conventional *linear shift-invariant* (LSI) filters have been widely exploited to process images, mostly from a *frequency* point of view. For example, a low-pass filter, such as average filter and Gaussian filter, can be applied to preserve the lower frequencies of an image while suppressing the higher ones. Likewise, the opposite can be achieved by using a high-pass or band-pass filter, such as Laplacian filter, Laplacian of Gaussian filter, Sobel filter, and so on. However, such frequency-based filters are unable to deliver a semantic-level filtering of the image contents; that is, to keep only the semantically meaningful contents of an image while removing the subjectively unimportant ones. This is because the semantic importance of image contents cannot be simply determined based on the frequencies. A demonstration of this insight is shown in Fig. 5.1. One can see from Fig. 5.1 (b) that the extracted important structures of the input image contain significant frequency

Figure 5.1: Top row (from left to right): the input image, the filtered image (or extracted important image structures) by using the proposed SICF algorithm (to be detailed in Section 5.3), and the filtered image by using a low-pass Gaussian filter. Bottom row: the amplitude spectrum of the Fourier transform of the respective images on the top row. Note that the zero-frequency component is situated at the center of the spectrum.

components of both low and high frequencies. This is also the case for those removed unimportant details, whose Fourier spectrum is the difference between the spectrums as shown in Fig. 5.1 (a) and (b). As a result, it is impossible to separate the main image structures from the unwanted details using any frequency-based filters; for example, when a low-pass filter is applied, not only will the main structures be blurred, but also the unwanted details cannot be completely removed, as shown in Fig. 5.1 (c). Therefore, to accomplish the aforementioned semantic-level filtering task, which we coin as *semantic image content filtering* (SICF) in this thesis, more sophisticated image filters are needed. For that, a new SICF algorithm is developed in this chapter, followed by demonstrating several application examples to show the effectiveness and potential of the SICF.

Over the last decades, the so-called *edge-aware* or *edge-preserving* filters (e.g., [40]–[47], [98]–[106]) have emerged and been widely attempted to conduct SICF. As the name

suggests, they are intended to preserve the distinct edges presented on the image, which are considered to be visually important, while smoothing other insignificant details. This makes them clearly advantageous over the conventional low-pass LSI filters in applications like noise removal and compression artifacts suppression [107]. Moreover, due to such attractive “edge-preserving” capability, these filters can also be applied as a pre-processing step to improve the performance of many other image processing techniques, such as edge detection [46], semantic segmentation [108], tone mapping [109], and so on.

Most existing edge-aware filters can be categorized into two classes—i.e., *local-average-based* (e.g., [40]–[43], [98]–[103]) and *global-optimization-based* (e.g., [44]–[47]) filters. In the former, the filtered value at each pixel is explicitly or implicitly computed as a weighted average of its neighboring pixel values on the input image. The representative works in this class include the well-known *anisotropic diffusion filter* [40], *bilateral filter* [41], *guided filter* [42], and *domain transform filter* [43]. This class of filters focuses on determining the filter kernel coefficients at each pixel (i.e., the weights of that pixel’s neighboring pixels values) for computing its filtered value. Typically, the larger the spatial distance or the color/intensity difference between a neighboring pixel and the to-be-filtered pixel, the smaller the weight. Among this class of filters, the most widely-used one is probably the bilateral filter [41], in which the filter kernel is defined as a multiplication of two Gaussian kernels respectively defined in the spatial domain and range (intensity) domain. On the other hand, in the latter (i.e., the global-optimization-based edge-aware filters), the filtered image is obtained by seeking the optimum of a defined objective functional, which usually contains a data fidelity term and a smoothness (or regularization) term. One early work of this class of filters is the *total variation filter* [44], which employs the total variation (i.e., the l_1 norm of the gradient magnitude) as the regularization term. More recently, the *weighted-least-square filter* [45] and the *l_0 -smoothing filter* [46] have been developed that exploit the l_2 and l_0 norms of the image gradients

to define the regularization term, respectively. Compared with the local-average-based filters, it has been observed that the global-optimization-based filters are generally able to preserve edges more effectively. Nevertheless, they tend to incur more computational cost due to the involved optimization process.

In general, edge-aware filters are conducted involving only a single input image. However, if an additional image were used as another input image, the information extracted from this extra image can be exploited as the guidance information to adjust the coefficients of the filter kernel for local-average-based filters or define the objective functional for global-optimization-based filters. In this case, the extra image is known as the *guidance image*, and the filter becomes the so-called *joint filter* or *cross filter* [110]. It has been observed that by making use of the structural information from the guidance image, the joint filters can deliver superior filtering results to that of the ordinary edge-aware filters in various applications, such as depth map enhancement [107], image denoising [110], image colorization [111], image matting [112], and so on.

Despite the great popularity of the aforementioned edge-aware filters, they are often unable to effectively conduct SICF on the aspects of removing unwanted image details in *high* contrast and preserving important image structures in *low* contrast. This is due to the fact that they determine the semantic importance of image contents based on the color or intensity *contrast* [48]. Recently, it has been pointed out in [48] that the semantically unimportant image details should be identified according to their *scale* rather than the *contrast*. Inspired by this insight, several *scale-aware* filters (e.g., [48]–[53],[113],[114]) have been proposed that consider the *scale* of image contents in the design of filters. In [48], Subr *et al.* proposed to average two manifolds respectively constructed from the local maximums and minimums to smooth out those high-contrast fine-scale details. In [50], Karacan *et al.* utilized the measured *region covariance* to discriminate unimportant textures from major structures based on the repetition property of texture

patterns. In [51], a new distance measure that is able to implicitly capture the scale of structures, called the *tree distance*, is defined based on a *minimum spanning tree* (MST) constructed over the input image. By incorporating this new distance into the bilateral filter to replace the originally employed Euclidean distance, the resultant filter is able to effectively remove the unimportant small-scale details. In [49], Xu *et al.* developed a global-optimization-based scale-aware filter that employs a novel regularization term, called the *relative total variation* (RTV), to distinguish small-scale variations from large-scale ones, so that a larger amount of smoothing can be imposed on those unwanted small-scale variations to remove them from the input image.

More recently, a simple and effective scale-aware filter, called the *rolling guidance filter* (RGF) [52], has been developed and shown fairly impressive performance on conducting SICF. The RGF employs a “smooth-and-recover” strategy to achieve scale-aware filtering. It starts with a Gaussian smoothing step to remove the small-scale details of the input image. Obviously, this will inevitably blur the large-scale structures as well. To recover those major structures, a joint filtering process is then iteratively applied to the input image, where the blurred image obtained from the Gaussian smoothing step is employed as the initial guidance image. In essence, the “rolling guidance” means that the guidance image evolves from iteration to iteration, and the filtered image obtained from each iteration is served as the new guidance image for the next iteration. It has been proven in [52] that the filtered image obtained at each iteration is guaranteed to be sharper than that at the previous iteration, and this iterative process will eventually converge to an image where the original large-scale structures on the input image are largely restored, while the removed small-scale textures remain unrecovered. A demonstration of the filtered images of the RGF and several other filters is shown in Fig. 5.2. It can be observed that the RGF delivers much superior results compared to the Gaussian filter and the bilateral filter [41] on preserving large-scale edges while removing small-

Figure 5.2: Filtering results on (a) the “batman” image, obtained by using (b) the Gaussian filter, (c) the bilateral filter [41], (d) the rolling guidance filter [52], and (e) the proposed edge-preserving scale-aware filter, respectively.

scale textures. However, it can be also observed from Fig 5.2 (d) that the RGF are not completely edge-preserving as fairly-noticeable distortions have been introduced to the structural edges, i.e., many sharp corners on the edge have been heavily rounded. This is because the Gaussian filter, as applied for generating the initial detail-removed guidance image of the RGF, has caused severe degradations to those main structures, such that they cannot be satisfactorily recovered by iteratively applying the joint filtering process.

In this chapter, a new filtering algorithm, called the *edge-preserving scale-aware filter* (ESF), is developed to pursue and demonstrate the usefulness of the SICF. The proposed ESF is an extension of the scale-aware RGF with a significant improvement on the edge-preserving capability. The key novelty of the proposed ESF lies in the deploy-

ment of a more effective detail removal scheme that is able generate a much improved *initial* guidance image for the follow-up structure recovery process to start with. In the proposed detail removal scheme, a newly-developed *adaptively weighted relative to-total variation* (AWRTV) filter is first applied to suppress the small-scale details while effectively preserving the large-scale structures, followed by applying a fixed Gaussian filter to further remove possible residual details. Compared with original RGF’s detail removal scheme using only a Gaussian filter, the proposed scheme is able to smooth out more small-scale details while imposing much less degradations to the important large-scale structures. As a result, those important structural edges can be more effectively recovered in our proposed ESF through the same structure recovery process as employed in the original RGF, as demonstrated in Fig. 5.2 (e). Extensive simulations conducted on a variety of test images have demonstrated that the proposed ESF is able to deliver superior SICF results on the aspects of small-scale detail removal and large-scale edge preservation, compared to other state-of-the-art edge-aware or scale-aware filters for conducting the SICF.

The rest of this chapter is organized as follows. Section 5.2 provides a succinct review of the RGF method as necessary background for the development of our proposed ESF algorithm. Section 5.3 describes the proposed ESF method in detail. Section 5.4 conducts performance evaluations of our proposed ESF against several comparable filters, followed by demonstrating several practical applications that can be greatly benefited by our ESF. Finally, a summary of this chapter is provided in Section 5.5.

5.2 Rolling Guidance Filter

The flowchart of the RGF [52] is demonstrated in Fig. 5.3. In essence, it is an iteratively conducted joint filtering process with two input images; one is the original input image

Figure 5.3: The flowchart of the *rolling guidance filter* (RGF) [52]. Note that R^0 is a Gaussian-blurred version of the input image I , where the superscript denotes the iteration index.

I , and the other is a detail-removed version of I , denoted as R^k . In each iteration, the filtered image R^k is an enhanced version of R^{k-1} , in the sense that the main structures on R^k are getting closer to those of the original image I . Note that the input image I is always presented in each iteration k and used to compute the R^k . In [52], the initial detail-removed image R^0 is obtained by applying a Gaussian filter to the input image I . That is,

$$R_p^0 = \frac{1}{Z_p} \sum_{q \in \Omega_p} \exp\left(-\frac{\|p - q\|^2}{2\sigma^2}\right) I_q, \quad (5.1)$$

where $Z_p = \sum_{q \in \Omega_p} \exp\left(-\frac{\|p - q\|^2}{2\sigma^2}\right)$ is a normalization factor. Here, Ω_p denotes a $(2r + 1) \times (2r + 1)$ local region centered at pixel p ; symbols r and σ are the radius and standard deviation of the Gaussian kernel, respectively. Finally, $\|p - q\|$ is the Euclidean distance between the pixel p and its neighboring pixel $q \in \Omega_p$. In our implementation, the Gaussian kernel radius r is empirically set to 3σ . The objective of applying the Gaussian filter is to remove those small-scale image details from the image I . However, the main structures of the image will be inevitably blurred as well, as demonstrated in Fig. 5.3.

At each follow-up iteration k ($k > 0$), a joint filtering process will be applied to the input image I , using the filtered image R^{k-1} from the previous iteration as the guidance. The obtained filtered image R^k will then be used as the guidance image for the next iteration. The objective of such iteratively conducted joint filtering process is to progressively enhance the blurred main structures on R^k , until the original main structures of I are satisfactorily recovered. In [52], the joint bilateral filter [110] is employed by default to conduct the joint filtering at each iteration¹, in which case the filtered value R_p^k at each pixel p is given by

$$R_p^k = \frac{1}{J_p} \sum_{q \in \Omega_p} \exp\left(-\frac{\|p - q\|^2}{2\sigma_s^2}\right) \exp\left(-\frac{\|R_p^{k-1} - R_q^{k-1}\|^2}{2\sigma_r^2}\right) I_q, \quad (5.2)$$

where

$$J_p = \sum_{q \in \Omega_p} \exp\left(-\frac{\|p - q\|^2}{2\sigma_s^2}\right) \exp\left(-\frac{\|R_p^{k-1} - R_q^{k-1}\|^2}{2\sigma_r^2}\right) \quad (5.3)$$

is a normalization factor. One can see that the joint bilateral filter kernel is a product of two Gaussian kernels; i.e., a *spatial* kernel with standard deviation σ_s and a *range* kernel with standard deviation σ_r . In the RGF [52], σ_s and σ (the standard deviation of the Gaussian kernel in (5.1)) are set to the same value, i.e., $\sigma_s = \sigma$. The joint bilateral filter has two inputs, i.e., the *input* image I and the *guidance* image R^{k-1} , and one output image, i.e., R^k , as can be seen from (5.2). The R^{k-1} is called the guidance image in [52] because the range kernel $\exp\left(-\frac{\|R_p^{k-1} - R_q^{k-1}\|^2}{2\sigma_r^2}\right)$ as in (5.2) is computed based on the intensity differences on R^{k-1} , rather than on the input image I itself as practiced in the standard bilateral filter. Given $\sigma_s = \sigma$, it can be observed from (5.1) with (5.2) that R^k will be equivalent to R^0 when the range kernel is a constant. This property allows the joint bilateral filter to impose two different effects on the small-scale details and large-scale structures simultaneously, as follows:

¹It has been reported in [52] that besides the joint bilateral filter, some other local-average-based filters can also be deployed to achieve similar results, such as the guided filter [42] and the domain transform filter [43].

- For small-scale details, since they have been completely removed by the Gaussian filter, R^0 will be *nearly flat* in the corresponding local regions. In this case, the range kernel of the joint bilateral filter in (5.2) will be almost a constant since $\|R_p^0 - R_q^0\|$ are close to 0. As mentioned, this will make $R_p^1 \approx R_p^0$ in those regions according to (5.1) and (5.2). Similarly, it can be further deduced that $R_p^k \approx R_p^{k-1} \approx \dots \approx R_p^0$ for the same regions. In other words, for those removed small-scale details, the joint bilateral filter will impose almost no enhancement or recovery.
- For large-scale structures, their sharp boundaries tend to be blurred but not completely removed by the Gaussian filter. Therefore, the corresponding regions on R^0 are not completely flat (i.e., intensity variations still exist). In this case, the range kernel at iteration $k = 1$, i.e., $\exp\left(-\frac{\|R_p^0 - R_q^0\|^2}{2\sigma_r^2}\right)$, will take effect and make the joint bilateral kernel in (5.2) ‘narrower’ than the Gaussian filter kernel in (5.1). As a result, the filtered value R_p^1 obtained via the joint bilateral filter will be closer to the original value I_p , compared to the R_p^0 as obtained via the Gaussian filter. In other words, R^1 will be sharper than R^0 on those regions corresponding to large-scale structures of I . Furthermore, with those large-scale structures becoming sharper (i.e., having larger intensity differences) on R^1 , the range kernel tends to be even narrower at iteration $k = 2$. This makes the joint bilateral kernel also become narrower and thus the obtained R^2 will be even sharper than R^1 . Therefore, as the iteration goes, the blurred large-scale structures will be gradually enhanced.

It has been reported in [52] that the above-mentioned iterative process will finally converge and the initial main structures on the input image I will be largely recovered in the final output image R^K , as shown in Fig. 5.3. However, it has been observed that the RGF often leads to severe distortions on some important structural edges; that is, it is not an effective edge-preserving filter. For example, one can observe from Fig. 5.3 that

the dog's tail are not well preserved in the final output image R^K . Besides, the sharp corners on the input image I tend to be heavily rounded, as previously demonstrated in Fig. 5.2. This limitation of the RGF has been well addressed in our proposed method, which is to be detailed in the following section.

5.3 Proposed Edge-preserving Scale-aware Filter

In this chapter, a novel extension of the original RGF [52], called the *edge-preserving scale-aware filter* (ESF), is developed. We have observed that the edge distortions encountered by the original RGF are mainly caused by the severe blurring imposed by the employed Gaussian filter in the detail removal stage. Motivated by such observation, a more effective detail removal scheme is developed in this work to generate a much improved initial guidance image R^0 for the follow-up structure recovery process to start with. For that, an improved version of the *relative total variation* filter [49], called the *adaptively weighted relative total variation* (AWRTV) filter, is developed and employed to largely remove the small-scale details on the input image while effectively preserving the large-scale structures. After applying the AWRTV filter, a Gaussian filter with a fixed and relatively small variance value is further employed on the obtained image to remove those residual (or unremoved) details. Compared with the original RGF's detail removal scheme using only a Gaussian filter, the proposed detail removal scheme is able to smooth out more small-scale details on the input image while imposing much less degradations on the large-scale main structures. Consequently, the main structural edges will be recovered more effectively in our proposed ESF using the same structure recovery process of the original RGF (as depicted in Fig. 5.3). In summary, our proposed ESF also has two sequential stages: 1) small-scale detail removal (i.e., generating the initial detail-removed guidance image R^0); and 2) large-scale structure recovery (i.e., producing

the structure-recovered output image R^K). In what follows, these two stages will be described in detail.

5.3.1 Stage 1: Detail Removal

In this stage, a global-optimization-based filter, called the *adaptively weighted relative total variation* (AWRTV) filter, is developed and first applied to the input image in order to remove those small-scale details while preserving those large-scale structures. The developed AWRTV is inspired by the RTV filter as developed in [49], but with several critical modifications as to be detailed in the following. It conducts small-scale detail removal by optimizing the following objective functional:

$$\mathcal{F}(S) = \sum_p \left\{ (S_p - I_p)^2 + \lambda \left(\frac{|(\partial_x S)_p|}{\mathcal{L}_{x,p} + \epsilon} + \frac{|(\partial_y S)_p|}{\mathcal{L}_{y,p} + \epsilon} \right) \right\}, \quad (5.4)$$

where I and S denote the input image and the desired detail-removed image, respectively.² The first term in (5.4) is a quadratic data fidelity term, while the second term is a regularization term. The constant λ balances the impacts of these two terms. In the regularization term, the partial differentiation operators ∂_x and ∂_y compute the derivatives of S along the horizontal (i.e., x -axis) and the vertical (i.e., y -axis) directions, respectively, and the small constant $\epsilon = 10^{-3}$ is employed to avoid any numerical instability. The $\mathcal{L}_{x,p}$ and $\mathcal{L}_{y,p}$ in the denominators are termed as the *windowed inherent variation* in [49], which are defined as follows:

$$\mathcal{L}_{x,p} = \left| \sum_{q \in \Omega_p} w_{p,q} \cdot (\partial_x S)_q \right|, \quad \mathcal{L}_{y,p} = \left| \sum_{q \in \Omega_p} w_{p,q} \cdot (\partial_y S)_q \right|, \quad (5.5)$$

where $w_{p,q}$ is a Gaussian weighting factor for each $(\partial_x S)_q$ and $(\partial_y S)_q$. The $\mathcal{L}_{x,p}$ and $\mathcal{L}_{y,p}$ play an important role on differentiating large-scale structures and small-scale details;

²For the sake of discussion, the images I and S are assumed to be grayscale images. In case I is a color image, the described smoothing process will be conducted to its three channels individually.

Figure 5.4: Illustration of the gradient directions in different types of patches: (a) input image; (b)-(c) gradient direction maps of a patch containing large-scale structure (red-framed) and another containing only small-scale textures (green-framed), respectively.

specifically, they tend to be smaller if the surrounding region of pixel p contains *only* small-scale textures, because the gradients in such regions are with more *inconsistent* or *scattered* directions, as demonstrated in Fig. 5.4, and thus they will counteract each other when being aggregated. Hence, by incorporating the $\mathcal{L}_{x,p}$ and $\mathcal{L}_{y,p}$ in the denominators, the regularization term is able to impose *larger* penalties to those detail-only regions than to those regions containing large-scale structures.

The most important difference between our objective functional as defined in (5.4) and that in the original RTV method [49] lies in the employed weighting scheme for computing $\mathcal{L}_{x,p}$ and $\mathcal{L}_{y,p}$ in the regularization term. In [49] the weighting factor $w_{p,q}$ as in (5.5) is computed from an *isotropic* Gaussian kernel as shown in Fig. 5.5 (a), while in our approach it is adaptively computed from either *isotropic* or *anisotropic* Gaussian kernels, depending on whether the pixel p is near a structural edge or not, and if yes, what the edge's direction is. Therefore, our approach is coined as *adaptively weighted relative total variation* (AWRTV) filter.

Ideally, if p is situated near an edge along θ direction, the employed Gaussian kernel should be along the θ direction as well, such that less texture pixels will be involved in the computation of $\mathcal{L}_{x,p}$ and $\mathcal{L}_{y,p}$. For simplicity, only four directions are considered in

Figure 5.5: Different Gaussian weighting kernels as employed in our approach: (a) the isotropic Gaussian kernel with standard deviation σ_v ; (b)-(e) the four anisotropic Gaussian kernels along the 0° , 90° , 45° , and 135° directions, respectively. For each anisotropic kernel, the standard deviations along the major and minor axes are σ_v and $\sigma_v/2$, respectively.

this work, i.e., 0° , 90° , 45° , and 135° . Accordingly, four anisotropic Gaussian kernels are used in addition to the original isotropic Gaussian kernel, as illustrated in Fig. 5.5. For each pixel p , the $w_{p,q}$ will be computed from one of these five Gaussian kernels, i.e., $w_{p,q} \propto G^*(x_p - x_q, y_p - y_q)$, where $G^* \in \{G_{iso}, G_{0^\circ}, G_{90^\circ}, G_{45^\circ}, G_{135^\circ}\}$. Here x_p and y_p are the x -axis and y -axis coordinates of pixel p , respectively; likewise, for x_q and y_q . A straightforward strategy is employed to determine the G^* , based on the observation that at an edge that is approximately along the θ direction, the G_θ will yields the largest value of $\mathcal{L}_{x,p} + \mathcal{L}_{y,p}$ (as its involved pixels are mostly on the edge and having similar gradient directions). For that, denote the value of $\mathcal{L}_{x,p} + \mathcal{L}_{y,p}$ yielded by using each of these five Gaussian kernels as ϕ_{iso} , ϕ_{0° , ϕ_{90° , ϕ_{45° , and ϕ_{135° , respectively, then the G^* is determined as follows:

$$G^* = \begin{cases} G_{\theta^*}, & \phi_{\theta^*} > \eta \cdot \phi_{iso}; \\ G_{iso}, & \text{otherwise,} \end{cases} \quad (5.6)$$

where $\theta^* = \arg \max \phi_\theta$; that is G_{θ^*} is the anisotropic Gaussian kernel that yields the largest ϕ_θ . The condition $\phi_{\theta^*} > \eta \cdot \phi_{iso}$ ensures p is indeed an ‘edge’ pixel, because otherwise ϕ_{θ^*} would be quite close to ϕ_{iso} and the latter should be used for better robustness. In this work, the threshold η is empirically set to 1.2 and used throughout all our experiments. It has been observed from extensive experimental results exploiting

the anisotropic Gaussian weighting kernels for computing the $\mathcal{L}_{x,p}$ and $\mathcal{L}_{y,p}$ allows our developed AWRTV filter to preserve those thin structures much better than the original RTV filter, as to be demonstrated later.

Another difference between our objective functional and the original RTV's is that the numerators in the regularization term were $\sum_{q \in \Omega_p} w_{p,q} \cdot |(\partial_x S)_q|$ and $\sum_{q \in \Omega_p} w_{p,q} \cdot |(\partial_y S)_q|$ in the original RTV [49], while in our approach they are simply $|(\partial_x S)_p|$ and $|(\partial_y S)_p|$ (i.e., without being locally weighted), respectively. This considerably reduces the algorithmic complexity of the follow-up optimization process, while leading to no noticeable degradation on the filtering results according to our extensive experimental results.

The objective functional as defined in (5.4) is non-convex. Therefore, an iterative numerical approach is resort to in [49] to optimize it, which is similar to the iterative reweighted least square algorithm as proposed in [115]. The key idea of this iterative numerical approach is to approximate the two non-convex terms of the regularization term in each iteration n by two quadratic terms, respectively, as follows:

$$\frac{|(\partial_x S)_p|}{\mathcal{L}_{x,p} + \epsilon} \approx \frac{1}{\mathcal{L}_{x,p}^{n-1} + \epsilon} \frac{(\partial_x S)_p^2}{|(\partial_x S^{n-1})_p| + \epsilon_s}, \quad (5.7)$$

$$\frac{|(\partial_y S)_p|}{\mathcal{L}_{y,p} + \epsilon} \approx \frac{1}{\mathcal{L}_{y,p}^{n-1} + \epsilon} \frac{(\partial_y S)_p^2}{|(\partial_y S^{n-1})_p| + \epsilon_s}, \quad (5.8)$$

where S^{n-1} is the intermediate smoothed image obtained from iteration $n - 1$, and the whole iterative process starts with the initial smoothed image $S^0 = I$. Similar to ϵ , the small constant $\epsilon_s = 0.02$ is used to avoid the division-by-zero incidence. The $\mathcal{L}_{x,p}^{n-1}$ and $\mathcal{L}_{y,p}^{n-1}$ are the windowed inherent variation computed on S^{n-1} via (5.5). Let $u_{x,p}^{n-1} = \frac{1}{\mathcal{L}_{x,p}^{n-1} + \epsilon}$, $u_{y,p}^{n-1} = \frac{1}{\mathcal{L}_{y,p}^{n-1} + \epsilon}$, $v_{x,p}^{n-1} = \frac{(\partial_x S)_p^2}{|(\partial_x S^{n-1})_p| + \epsilon_s}$, and $v_{y,p}^{n-1} = \frac{(\partial_y S)_p^2}{|(\partial_y S^{n-1})_p| + \epsilon_s}$, the intermediate smoothed image S^n ($n \geq 1$) is obtained by optimizing the following functional, i.e.,

$$S^n = \arg \min_p \left\{ (S_p - I_p)^2 + \lambda \left[u_{x,p}^{n-1} v_{x,p}^{n-1} (\partial_x S)_p^2 + u_{y,p}^{n-1} v_{y,p}^{n-1} (\partial_y S)_p^2 \right] \right\}. \quad (5.9)$$

Figure 5.6: Filtering results of different detail-smoothing filters: (a) input image; (b)-(d) smoothed images by using the (b) Gaussian filter, (c) the original RTV filter [49], and (d) the proposed AWRTV filter.

Since $u_{x,p}^{n-1}$, $u_{y,p}^{n-1}$, $v_{x,p}^{n-1}$, and $v_{y,p}^{n-1}$ are all constants that can be computed from the S^{n-1} , this objective functional is essentially in the same form of the objective functional of the well-known *weighted-least-square* (WLS) filter [45], which has been exploited to conduct the blurriness map refinement process as introduced in Chapter 4. Since (5.9) is regularized by a convex quadratic term, a global optimum exists and it can be obtained by solving a derived linear system, as practiced in [87] and [45]. Such optimization process will be repeated until it converges (typically within 5 iterations as reported in [49]). Suppose the iteration is terminated at iteration N , then the final output image of the proposed AWRTV filter will be $S = S^N$.

In Fig. 5.6, the detail removal results of different filters are demonstrated. One can see that the proposed AWRTV clearly outperforms the Gaussian filter as it smooths out the small-scale details more thoroughly while preserving the major structures much better. Compared with the original RTV filter, the proposed AWRTV is able to better preserve the structures that are thin or not very large, as respectively indicated by the red and blue arrows in both Fig. 5.6 (c) and (d). This is mainly due to the developed adaptive weighting scheme for computing the $\mathcal{L}_{x,p}$ and $\mathcal{L}_{y,p}$ in the regularization term.

Despite the promising filtering results yielded by the developed AWRTV, its pro-

duced filtered images still suffer from two major issues: 1) considerable contrast loss could be imposed to the main structures; and 2) some small-scale details near those large-scale edges might have not been completely removed. The former is also known as the “intensity shift” issue, which is commonly encountered by global-optimization-based filters [42]. It is introduced mainly because the smoothness term in the objective functional will globally suppress the contrast of the image. On the other hand, the latter is introduced because the $\mathcal{L}_{x,p}$ and $\mathcal{L}_{y,p}$ in the regularization term are computed over a local window. This means when the details are near an edge, they will have similar measured $\mathcal{L}_{x,p}$ and $\mathcal{L}_{y,p}$ values to those of the edge pixels. Consequently, they will be treated as edge pixels and thus largely preserved.

In order to address the first issue, the structure recovery process as employed in the original RGF (as depicted in Fig. 5.3) can be applied directly using the filtered image of the AWRTV (i.e., S) as the initial guidance image, i.e., $R^0 = S$. However, this approach cannot remove the residual details left on S . To tackle this problem, a Gaussian filter G is further applied to S to suppress those residual details, and the obtained blurred image is then fed to the structure recovery process to progressively recover the major structures, i.e., $R^0 = G * S$, where $*$ denotes the convolution operation. The standard deviation of the employed G is empirically set to 2, which is able to sufficiently suppress the residual details in most cases, while not yielding severe blur to the main structures.

5.3.2 Stage 2: Structure Recovery

In this stage, the same structure recovery process as depicted in Fig. 5.3 will be conducted, except using $R^0 = G * S$, rather than a Gaussian smoothed version of the input image. In summary, the processing steps of our proposed ESF can be described based on Fig. 5.7 as follows. Given the input image I (Fig. 5.7 (a)), the developed AWRTV filter is first applied to generate a filtered image S (Fig. 5.7 (b)), where the small-scale details are

Figure 5.7: A demonstration of the input and intermediate images involved in the proposed ESF: (a) the input image I ; (b) the filtered image S obtained by using the proposed AWRTV; (c) the further smoothed version of S , i.e., $G * S$; (d)-(f) filtered images after the 1st, 2nd, and 10th iteration of the structure recovery process. One can see that the main image structure becomes well-recovered after only 2 iterations.

largely removed. A Gaussian filter G with the standard deviation constantly set to be 2 is further exploited to remove the residual details on S , and the obtained image (Fig. 5.7 (c)) is then served as the initial guidance image for the structure recovery process as employed in the original RGF, i.e., $R^0 = G * S$, to generate the final output image R^K . It is worth noting that, since the initial guidance image R^0 in our approach is much less degraded than that produced by the Gaussian filter (as shown in Fig. 5.6 (b)), the follow-up structure recovery process (i.e., iteratively conducted joint bilateral filtering process) will require fewer iterations to converge, as demonstrated in Fig. 5.8. It has been observed from extensive simulations that terminating the iterative process after 2 iterations (i.e., using R^2 as the final output image) is sufficient for our approach to produce satisfactory

Figure 5.8: The mean absolute difference between the filtered images R^k and R^{k-1} (denoted by $MAD(k)$) as measured on the “dog” image in Fig. 5.7 (a) by applying the proposed ESF and the RGF, respectively.

results in most cases. As for the original RGF, its number of iterations is set to be 5 as suggested in [52], and such setting is used throughout all our experiments.

In Fig. 5.9, a comparison of the structure recovery results using different R^0 is demonstrated. The parameters used in the structure recovery process are the same for all the compared cases. One can see that the initial guidance image R^0 indeed has a critical impact on the quality of the final output image R^K . Specifically, using only a Gaussian filter to generate the R^0 as practiced in the original RGF has resulted in a poor-quality R^K where the main structural edges are highly distorted. On other hand, using only the developed AWRTV could lead to unremoved details. Lastly, by further exploiting a Gaussian filter (with a small variance $\sigma = 2$) after the AWRTV filtering to generate R^0 , the proposed ESF is able produced a R^K where the details are thoroughly removed while the main edges are well preserved.

Figure 5.9: A comparison of the structure recovery results R^K using different initial guidance image R^0 . Top row: the input image I . Middle row: the R^0 obtained by applying (b) the Gaussian filter (with $\sigma = 6$) as practiced in the RGF [52], (c) the developed AWRTV filter only, (d) the developed AWRTV followed by a Gaussian filter (with $\sigma = 2$). Bottom row: the R^K obtained by using the respective R^0 from the above row as the initial guidance image. The parameters for the structure recovery process are set to $\sigma_r = 0.05$ and $\sigma_s = 6$ for all three cases.

5.4 Experimental Results

5.4.1 Subjective Evaluations

Extensive simulations have been conducted to evaluate the performance of our proposed ESF using a variety of test images. The filtered results of the proposed ESF are compared to several state-of-the-art smoothing filters, including two representative edge-aware filters (i.e., the *bilateral filter* (BF) [41] and the *weighted-least-square* (WLS) filter [45]) and four recently-developed scale-aware filters (i.e., the *tree filter* (TF) [51], the *static and dynamic guidance* (SD) filter [53], the original *relative total variation* (RTV) filter [49], and the original *rolling guidance filter* (RGF) [52]).

There are four parameters for our proposed ESF—two in the AWRTV-based detail smoothing stage (i.e., the constant λ as in (5.4) and the standard deviation σ_v of the Gaussian kernels for computing the $\mathcal{L}_{x,p}$ and $\mathcal{L}_{y,p}$ as defined in (5.5)), and two in the follow-up RGF stage (i.e., the spatial standard deviation σ_s and the range standard deviation σ_r of the joint bilateral filter as in (5.2)). Among these parameters, λ and σ_v control the detail smoothing ability of the proposed AWRTV filter; that is, the larger the λ and σ_v , the more details will be removed. In contrast, σ_s and σ_r control the structure recovering ability of the follow-up joint filtering stage. In order to avoid the removed details to be recovered, the σ_s should be set to a sufficiently large value; specifically, it has been observed that $\sigma_v \leq \sigma_s \leq 2\sigma_v$ is a good choice in most cases. As for σ_r , it mainly controls how well the main structures can be recovered; i.e., a larger value of the σ_r will lead more blurred main structures. It should be noted that there are no optimal settings for these parameters, and they should be determined for each input image based on the desired smoothing effect for the specific application.

Fig. 5.10 demonstrates the filtered images obtained by using different filters for high-contrast texture removal. It can be observed that the two representative edge-aware filters,

Figure 5.10: Detail removal results of different detail-smoothing filters: (a) original image; (b) BF ($\sigma_s = 4$, $\sigma_r = 0.5$); (c) WLS ($\lambda = 4$, $\alpha = 2$); (d) TF ($\sigma = 0.04$, $\sigma_s = 4$, $\sigma_r = 0.1$, with 2 iterations); (e) SD ($\lambda = 500$, $\mu = 50$, $\nu = 300$, with 5 iterations); (f) RTV ($\lambda = 0.01$, $\sigma = 4$, with 4 iterations); (g) RGF ($\sigma_s = 4$, $\sigma_r = 0.1$); (h) proposed ESF ($\lambda = 0.02$, $\sigma_v = 4$, $\sigma_s = 8$, $\sigma_r = 0.05$).

i.e., BF and WLS, all failed to smooth out those high-contrast textures as shown in Fig. 5.10 (b) and (c). The four state-of-art scale-aware filters delivered much improved results, but they also have imposed different issues, such as residual textures and degradation on the main structures, as can be observed from Fig. 5.10 (d)-(h). In particular, one can see from Fig. 5.10 (g) that the original RGF using Gaussian filter for generating the initial guidance image significantly rounded the corners, as expected. The proposed, in contrast, has successfully avoided all these issues and delivered the best visual quality of the filtered image. Another demonstration of the filtering results yielded by these methods is shown in Fig. 5.11, from which the same comparison statements as drawn

from Fig. 5.10 can also be obtained.

5.4.2 Applications

Due to the attractive edge-preserving scale-aware filtering capacity, the proposed ESF can benefit a variety of practical image processing applications, such as image enhancement, texture replacement, image de-halftoning, edge detection, image denoising, image segmentation, to name a few. In what follows, the results of applying our proposed ESF in several common applications are demonstrated.

Image Enhancement As introduced in Chapter 4, a prevalent strategy for conducting image enhancement is to first decompose the input image into two layers—i.e., a base layer containing major structures and a detail layer containing those fine details. Based on these two layers, the enhancement is realized by amplifying the detail layer and then adding it back to the base layer. The conventional Gaussian filter and the edge-aware filters have been widely-used to accomplish the layer-decomposition step (e.g., [33], [41]–[43], [45], [46]). Naturally, the proposed ESF can also be deployed for this purpose. Fig. 5.12 demonstrates the enhancement results obtained by using different layer-decomposition filters. Note that in this simulation the enhancement is only conducted for the luminance component of the input color image, and the scaling factor applied to the detail layer is fixed to 3. One can see from the obtained base layers (top row) that the proposed ESF is clearly more effective than the BF and the RGF on smoothing out the small-scale details (i.e., extracting them into the detail layer), while preserving the large-scale edges on the base layer. This results in much more significant enhancement of these details without yielding any noticeable artifacts.

Texture Replacement In addition to image enhancement, texture replacement is another practical application that can be realized by exploiting the above-mentioned

decomposed base layer and detail layer. To be more specific, instead of amplifying the detail layer, one can simply replace the detail layer with another texture image, followed by adding it back to the base layer. By doing so, the texture pattern in the original input image will be replaced by the new texture pattern. An example is illustrated in Fig. 5.13.

Image De-halftoning *Halftoning* is the process of approximating continuous color or grayscale tones using various dot patterns [116]. It is universally used by printing devices to render the images to be printed. Owing to the increasing popularity of digital displays and storage devices, there is a trend of digitalizing existing printed materials, such as books, magazines, newspapers, and so on, into digital form for more convenient retrieval and persistent archiving. For that, the *de-halftoning* or *inverse halftoning* process is needed to reconstruct a continuous digital image from the scanned halftone images. While the proposed ESF is not tailored for this specific processing task, it is able deliver quite good de-halftoning results in terms of dot pattern removal and main structure preservation, as demonstrated in Fig. 5.14.

Edge Detection The existence of strong textures often degrade the results of various image processing tasks such as edge detection and semantic segmentation. For such applications, our proposed ESF can be utilized as a preprocessing step to remove those detrimental textures, thus improving the accuracy or effectiveness of the follow-up processing steps. In Fig. 5.15, the edge detection results obtained by applying a canny edge detector [117] on an image and its pre-processed versions via different filters are demonstrated. The Matlab’s built-in `edge` function is employed to conduct the canny edge detection, where the gradient threshold parameter is set to 0.4. It can be observed from Fig. 5.15 (a) that directly applying the edge detection on the input image yields

TABLE 5.1: AVERAGE RUNNING TIME (IN SECONDS) OF DIFFERENT METHODS FOR FILTERING A 500×375 IMAGE

BF [41]	WLS [45]	TF [51]	SD [53]	RTV [49]	RGF [52]	Proposed
0.22	1.90	0.45	5.21	1.78	0.68	2.12

a number of false edges. This issue can be alleviated to some extent by conducting a bilateral filtering process as the pre-processing step, but there are still a number of false edges, as shown in Fig. 5.15 (b). On the other hand, the RGF and the proposed ESF are able to completely avoid the false edge issue. Moreover, the resultant edge map of our ESF is more accurate than that yielded by the RGF, as indicated by the red and blue arrows in Fig. 5.15 (c) and (d).

5.4.3 Comparison of Computational Complexity

In this subsection, a comparison of the computational complexity of the proposed ESF and other state-of-the-art filtering techniques for conducting SICF is presented. All these methods are tested using their MATLAB implementations on the same machine with an Intel Core i7-6700 3.4 GHz CPU and 8 GB RAM. The running times (average of 10 runs) of all methods for filtering a 500×375 image are documented in Table 5.1. One can see that the proposed ESF algorithm has a moderate computational cost (consuming 2.12 seconds on average) compared with other methods. Specifically, the detail-removing stage using the developed AWRTV filter costs around 1.72 seconds, while the structure recovering process based on iterative bilateral filtering cost about 0.40 seconds (with the maximum number of iterations set to be 2). Clearly, the AWRTV filter is the bottleneck of the efficiency of the proposed ESF. In future, it would worthwhile to investigate more efficient method, preferably one without the optimization process as currently employed in the AWRTV, for generating the initial guidance image of the proposed ESF.

5.5 Summary

In this chapter, a novel filtering algorithm, called the *edge-preserving scale-aware filter* (ESF), is developed for conducting semantic image content filtering (SICF). The proposed ESF is inspired by the recently-appeared scale-aware RGF algorithm, but with significantly improved edge-preserving capability. The key novelty our approach lies in the deployed new detail removal scheme with incorporation of the developed *adaptively weighted relative total variation* (AWRTV) filter. Compared with the original RGF's detail removal scheme using only a Gaussian filter, the proposed detail removal scheme is able to smooth out more small-scale details on the input image while imposing much less degradations on the large-scale main structures. Starting from such improved initial guidance image, our proposed ESF is able to recover those main structural edges much more effectively than the original RGF, using the same structure recovery process. Extensive simulations conducted on a variety of test images have demonstrated that the proposed ESF is able to deliver superior SICF results compared to the original RGF and other state-of-the-art edge-aware or scale-aware filters. This makes our proposed ESF highly desirable in many practical image processing applications that can be greatly benefited by an effective SICF process.

Figure 5.11: Detail removal results of different detail-smoothing filters: (a) original image; (b) BF ($\sigma_s = 4$, $\sigma_r = 0.5$); (c) WLS ($\lambda = 4$, $\alpha = 2$); (d) TF ($\sigma = 0.04$, $\sigma_s = 4$, $\sigma_r = 0.1$, with 2 iterations); (e) SD ($\lambda = 500$, $\mu = 200$, $\nu = 200$, with 5 iterations); (f) RTV ($\lambda = 0.02$, $\sigma = 4$, with 4 iterations); (g) RGF ($\sigma_s = 4$, $\sigma_r = 0.1$); (h) proposed ESF ($\lambda = 0.02$, $\sigma_v = 4$, $\sigma_s = 8$, $\sigma_r = 0.05$).

Figure 5.12: Image enhancement results using different layer-decomposition filters: (a) the input color image (bottom) and its luminance component (top) in the CIELab color space; (b)-(d) the extracted base layer images (top) and the resultant enhanced images (bottom) obtained by using (b) the BF ($\sigma_s = 5$, $\sigma_r = 0.2$), (c) the RGF ($\sigma_s = 5$, $\sigma_r = 0.1$), and (d) the proposed ESF ($\lambda = 0.04$, $\sigma_v = 8$, $\sigma_s = 5$, $\sigma_r = 0.1$), respectively.

Figure 5.13: Texture replacement results by using the proposed ESF: (a) the input image; (b) the extracted base layer by applying the proposed ESF ($\lambda = 0.03$, $\sigma_v = 4$, $\sigma_s = 4$, $\sigma_r = 0.08$); (c) the synthetic image by adding another texture image to (b).

Figure 5.14: Image de-halftoning results using different filters: (a) the input color image; (b)-(d) the reconstructed continuous-tone image obtained by using (b) the BF ($\sigma_s = 4$, $\sigma_r = 0.2$), (c) the RGF ($\sigma_s = 4$, $\sigma_r = 0.08$), and (d) the proposed ESF ($\lambda = 0.03$, $\sigma_v = 4$, $\sigma_s = 4$, $\sigma_r = 0.08$), respectively.

Figure 5.15: Edge detection results using a canny edge detector. Top row: the input image and its smoothed versions by using the BF ($\sigma_s = 10$, $\sigma_r = 0.2$), the RGF ($\sigma_s = 10$, $\sigma_r = 0.1$), and the proposed ESF ($\lambda = 0.06$, $\sigma_v = 5$, $\sigma_s = 10$, $\sigma_r = 0.1$), respectively. Bottom row: the detected edge maps for the respective images in the top row.

Chapter 6

Conclusions and Future Works

6.1 Conclusions

Owing to the recent emergence of high-definition (HD) displays and the resultant higher requirements on image resolution and quality, many of the existing image processing techniques need to be further advanced to go along with such trend of demand. Motivated by this vision, this thesis investigates four fundamental image processing techniques that are highly instrumental for generating high-resolution and high-quality imagery for the HD displays; i.e., *image demosaicing*, *image interpolation*, *image enhancement*, and *semantic image content filtering*. Four novel processing algorithms in the respective fields are developed, as summarized as follows.

- Proposed Image Demosaicing Algorithm Using Iterative Residual Interpolation (IRI)

The proposed IRI algorithm is a novel extension of the recently-developed residual interpolation (RI) based demosaicing strategy that has demonstrated superior performance over the conventional color-component difference interpolation (CDI) approach. Unlike the existing RI-based demosaicing methods [54] [56], which have only exploited the RI-based strategy for restoring the R and B channels, the proposed IRI algorithm further applies the RI-based strategy to the most important G

channel via the developed IRI process. The key novelty of the proposed IRI process lies in that the RI process is iteratively applied to all the three color channels by letting them mutually guide each other, such that a progressive refinement of their estimated missing pixel values is achieved. The proposed IRI algorithm is able to restore a much more accurate G channel, compared with the existing start-of-the-art methods. Extensive simulations conducted on two commonly-used test datasets for demosaicing algorithms have demonstrated that our algorithm has achieved the best performance in most cases, compared to the existing state-of-the-art demosaicing methods on both objective and subjective performance evaluations.

- Proposed Fast Contrast-guided Image Interpolation (CGI) Algorithm via Convolutional Edge Diffusion (CED)

In the proposed *fast* CGI algorithm, a *convolutional edge diffusion* (CED) scheme is developed to replace the *iterative edge diffusion* (IED) as employed in the original CGI method. The proposed CED is motivated by a key observation that the thresholding operation applied in the CGI framework for generating the *binary-valued* contrast-guided decision maps (CDMs) will inevitably destroy the high fidelity of the *continuous-valued* diffused directional variation (DV) field achieved through the data term in the IED's objective functional. Therefore, the data term is lifted in our approach to yield a new energy functional. It turns out that the diffusion equation derived from this simplified functional is, in fact, the well-known *heat equation*, from which has a highly attractive property that can be exploited for conducting diffusion. That is, given a desired amount of diffusion to be yielded, it can be realized by simply convolving the DV field with a Gaussian kernel *once*, rather than gradually updating the DV field through iterations. As a result, the total computation time is significantly reduced. Extensive simulation results have

shown that the proposed CED can generate nearly identical CDMs as those produced by the CGI's IED, while only requiring about 1/10 of its computation time. By replacing the IED with the proposed CED in the CGI framework, the resultant fast CGI algorithm requires only 1/4 of the total run time as consumed by the original CGI, while delivering almost the same interpolation performance. This makes our developed algorithm highly attractive for those applications that require real-time performance and low power consumption.

- Proposed Blurriness-guided Unsharp Masking (BUM) Algorithm for Adaptive Image Sharpness Enhancement

The proposed BUM algorithm is motivated by the key observation that enhancing those highly-blurred image regions is undesirable because it is unlikely that any perceptual improvements can be achieved. In fact, this severely-blurred situation should be handled by a restoration technique rather than an enhancement method. Furthermore, applying an enhancement process on such highly-blurred regions could often cause unwanted artifacts in return (e.g., amplified noise). Based on such observation, a two-stage pixel-wise blurriness estimation approach is developed, in which an initial blurriness map of the input image is first estimated via the sparse-representation-based *just noticeable blurriness estimation* (JNBE) [86] algorithm, followed by refining the obtained blurriness map through the developed *weighted-least-square* (WLS) based filtering process. After the refined blurriness map is obtained, an adaptive scaling factor for boosting the extracted image details is then imposed at each pixel location via a mapping process using the generalized Gaussian functions, such that those highly-blurred regions are less enhanced or not enhanced at all. Extensive simulations conducted on various test images have demonstrated that the proposed method is able to deliver superior perceptual

quality of the enhanced image compared with that by using the conventional UM and other existing adaptive UM methods.

- Proposed Edge-preserving Scale-aware Filter (ESF) for Semantic Image Content Filtering (SICF)

The proposed ESF is inspired by the recently-developed scale-aware *rolling guidance filter* (RGF) [52], but with a significant improvement on the aspect of edge preserving. The key novelty of our approach lies in the proposed detail removal scheme for generating a much improved initial guidance image for the follow-up structure recovery stage to start with. Compared with the original RGF's detail removal scheme using only a Gaussian filter, the proposed scheme is able to smooth out more small-scale details on the input image while causing much less degradations to the large-scale main structures. Consequently, those main structures can be recovered more effectively in our proposed ESF via the same structure recovery process as employed in the RGF. Extensive simulations conducted on a variety of test images have clearly demonstrated that the proposed ESF outperforms other state-of-the-art filters for conducting the SICF, on the aspects of small-scale detail removal and large-scale structure preservation. This makes our approach highly desirable in many practical image processing tasks, such as image enhancement, texture replacement, image de-halftoning, edge detection, and so on.

6.2 Future Works

In the following, several possible future research directions are discussed:

- High-dynamic-range (HDR) Image Generation for HD displays

Many major TV manufacturers (such as LG and Samsung) have recently released their Ultra-HD TVs with support of the *high-dynamic-range* (HDR) images, which

can represent a much wider range of contrast and colors than the typical *standard-dynamic-range* (SDR) images with only a 8-bits depth for each color channel. Clearly, this will bring a huge demand of the HDR images in future. In the past, many HDR image generation techniques have been developed, and their designs are mostly based on combining multiple SDR images taken at the same scene under various exposure levels (e.g., [118]–[120]). However, some challenging issues remain largely unsolved for this task; for instance, the so-called “ghost” artifacts are often incurred in the synthesized HDR image due to the camera or object motion during the capture of those SDR images. Therefore, how to effectively generate high-quality HDR images is still a critical problem that needs to be further investigated.

- Fundamental Video Processing Techniques for HD Displays

The developed processing algorithms in this thesis are mostly designed for digital still images. Although they can be applied for videos without any modifications by treating each video frame as a separate image, this is not an effective approach since the inherent temporal correlations existing across the video frames are not utilized. Some research works have already been carried out that exploit the temporal information to conduct video demosaicing [121], video super-resolution [122], and video enhancement [123]. However, due to the higher requirements on video resolution and quality as brought by the HD displays, these techniques clearly need to be further improved as well. Therefore, developing more advanced fundamental video processing techniques for the HD displays would be a worthwhile endeavor in the future works.

Publications

1. Wei Ye and Kai-Kuang Ma, “Image demosaicing by using iterative residual interpolation,” in *Proceedings of IEEE International Conference on Image Processing*, 2014, pp. 1862–1866.
2. Wei Ye and Kai-Kuang Ma, “Color image demosaicing using iterative residual interpolation,” in *IEEE Transactions on Image Processing*, vol. 24, no. 12, pp. 5879-5891, Dec. 2015.
3. Wei Ye and Kai-Kuang Ma, “Convolutional edge diffusion for fast contrast-guided image interpolation,” in *IEEE Signal Processing Letters*, vol. 23, no. 9, pp. 1260-1264, Sep. 2016.
4. Wei Ye and Kai-Kuang Ma, “Blurriness-guided unsharp masking,” to be submitted to *IEEE Transactions on Image Processing*.
5. Wei Ye and Kai-Kuang Ma, “Semantic image content filtering via edge-preserving scale-aware filter,” to be submitted to *IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2017.
6. Wei Ye and Kai-Kuang Ma, “Methods and systems for demosaicing an image,” PCT Patent, WO 2015119574 A1, 2015.
7. Wei Ye and Kai-Kuang Ma, “Fast contrast-guided image interpolation with convolutional edge diffusion,” SG Provisional Patent, Application No. 10201605141U,

filed by Nanyang Technological University, 22 Jun. 2016.

Bibliography

- [1] M. Sugawara, S. Y. Choi, and D. Wood, “Ultra-high-definition television (rec. itu-r bt.2020): A generational leap in the evolution of television [standards in a nutshell],” *IEEE Signal Processing Magazine*, vol. 31, no. 3, pp. 170–174, 2014.
- [2] R. Ramanath, W. E. Snyder, Y. Yoo, and M. S. Drew, “Color image processing pipeline,” *IEEE Signal Processing Magazine*, vol. 22, no. 1, pp. 34–43, 2005.
- [3] X. Li, B. Gunturk, and L. Zhang, “Image demosaicing: A systematic survey,” in *Proceedings of SPIE*, 2008, pp. 68 221J–68 221J–15.
- [4] J. E. Adams Jr and J. F. Hamilton Jr, “Adaptive color plane interpolation in single sensor color electronic camera,” U.S. Patent 5629734, 1997.
- [5] X. Wu and N. Zhang, “Primary-consistent soft-decision color demosaicking for digital cameras (patent pending),” *IEEE Transactions on Image Processing*, vol. 13, no. 9, pp. 1263–1274, 2004.
- [6] K. Hirakawa and T. W. Parks, “Adaptive homogeneity-directed demosaicing algorithm,” *IEEE Transactions on Image Processing*, vol. 14, no. 3, pp. 360–369, 2005.
- [7] K.-H. Chung and Y.-H. Chan, “Color demosaicing using variance of color differences,” *IEEE Transactions on Image Processing*, vol. 15, no. 10, pp. 2944–2955, 2006.

- [8] D. Menon, S. Andriani, and G. Calvagno, “Demosaicing with directional filtering and a posteriori decision,” *IEEE Transactions on Image Processing*, vol. 16, no. 1, pp. 132–141, 2007.
- [9] K.-H. Chung and Y.-H. Chan, “Low-complexity color demosaicing algorithm based on integrated gradients,” *Journal of Electronic Imaging*, vol. 19, no. 2, p. 021104, 2010.
- [10] J. S. J. Li and S. Randhawa, “Color filter array demosaicking using high-order interpolation techniques with a weighted median filter for sharp color edge preservation,” *IEEE Transactions on Image Processing*, vol. 18, no. 9, pp. 1946–1957, 2009.
- [11] L. Zhang and X. Wu, “Color demosaicking via directional linear minimum mean square-error estimation,” *IEEE Transactions on Image Processing*, vol. 14, no. 12, pp. 2167–2178, 2005.
- [12] D. Paliy, V. Katkovnik, R. Bilcu, S. Alenius, and K. Egiazarian, “Spatially adaptive color filter array interpolation for noiseless and noisy data,” *International Journal of Imaging Systems and Technology*, vol. 17, no. 3, pp. 105–122, 2007.
- [13] I. Pekkucuksen and Y. Altunbasak, “Gradient based threshold free color filter array interpolation,” in *Proceedings of the IEEE International Conference on Image Processing*, 2010, pp. 137–140.
- [14] L. Zhang, X. Wu, A. Buades, and X. Li, “Color demosaicking by local directional interpolation and nonlocal adaptive thresholding,” *Journal of Electronic Imaging*, vol. 20, no. 2, p. 023016, 2011.

- [15] I. Pekkucuksen and Y. Altunbasak, “Edge strength filter based color filter array interpolation,” *IEEE Transactions on Image Processing*, vol. 21, no. 1, pp. 393–397, 2012.
- [16] —, “Multiscale gradients based color filter array interpolation,” *IEEE Transactions on Image Processing*, vol. 22, no. 1, pp. 157–165, 2013.
- [17] W. Lu and Y.-P. Tan, “Color filter array demosaicking: new method and performance measures,” *IEEE Transactions on Image Processing*, vol. 12, no. 10, pp. 1194–1210, 2003.
- [18] X. Li, “Demosaicing by successive approximation,” *IEEE Transactions on Image Processing*, vol. 14, no. 3, pp. 370–379, 2005.
- [19] C.-Y. Su, “Highly effective iterative demosaicing using weighted-edge and color-difference interpolations,” *IEEE Transactions on Consumer Electronics*, vol. 52, no. 2, pp. 639–645, 2006.
- [20] A. K. Jain, *Fundamentals of digital image processing*. Englewood Cliffs: Prentice-Hall, 1989.
- [21] H. S. Hou and H. Andrews, “Cubic splines for image interpolation and digital filtering,” *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 26, no. 6, pp. 508–517, 1978.
- [22] R. G. Keys, “Cubic convolution interpolation for digital image processing,” *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 29, no. 6, pp. 1153–1160, 1981.

- [23] K. Jensen and D. Anastassiou, “Subpixel edge localization and the interpolation of still images,” *IEEE Transactions on Image Processing*, vol. 4, no. 3, pp. 285–295, 1995.
- [24] J. Allebach and P. W. Wong, “Edge-directed interpolation,” in *Proceedings of the IEEE International Conference on Image Processing*, vol. 3, 1996, pp. 707–710.
- [25] Q. Wang and R. K. Ward, “A new orientation-adaptive interpolation method,” *IEEE Transactions on Image Processing*, vol. 16, no. 4, pp. 889–900, 2007.
- [26] X. Li and M. T. Orchard, “New edge-directed interpolation,” *IEEE Transactions on Image Processing*, vol. 10, no. 10, pp. 1521–1527, 2001.
- [27] L. Zhang and X. Wu, “An edge-guided image interpolation algorithm via directional filtering and data fusion,” *IEEE Transactions on Image Processing*, vol. 15, no. 8, pp. 2226–2238, 2006.
- [28] X. Zhang and X. Wu, “Image interpolation by adaptive 2-d autoregressive modeling and soft-decision estimation,” *IEEE Transactions on Image Processing*, vol. 17, no. 6, pp. 887–896, 2008.
- [29] X. Liu, D. Zhao, R. Xiong, S. Ma, W. Gao, and H. Sun, “Image interpolation via regularized local linear regression,” *IEEE Transactions on Image Processing*, vol. 20, no. 12, pp. 3455–3469, 2011.
- [30] A. Giachetti and N. Asuni, “Real-time artifact-free image upscaling,” *IEEE Transactions on Image Processing*, vol. 20, no. 10, pp. 2760–2768, 2011.
- [31] K.-W. Hung and W.-C. Siu, “Robust soft-decision interpolation using weighted least squares,” *IEEE Transactions on Image Processing*, vol. 21, no. 3, pp. 1061–1069, 2012.

- [32] Z. Wei and K.-K. Ma, “Contrast-guided image interpolation,” *IEEE Transactions on Image Processing*, vol. 22, no. 11, pp. 4271–4285, 2013.
- [33] A. K. Jain, *Fundamentals of Digital Image Processing*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1989.
- [34] A. Polesel, G. Ramponi, and V. J. Mathews, “Image enhancement via adaptive unsharp masking,” *IEEE Transactions on Image Processing*, vol. 9, no. 3, pp. 505–510, 2000.
- [35] S. K. Mitra, H. Li, I.-S. Lin, and T.-H. Yu, “A new class of nonlinear filters for image enhancement,” in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 1991, pp. 2525–2528.
- [36] G. Ramponi, N. K. Strobel, S. K. Mitra, and T.-H. Yu, “Nonlinear unsharp masking methods for image contrast enhancement,” *Journal of Electronic Imaging*, vol. 5, no. 3, pp. 353–366, 1996.
- [37] G. Ramponi, “A cubic unsharp masking technique for contrast enhancement,” *Signal Processing*, vol. 67, no. 2, pp. 211–222, 1998.
- [38] S. H. Kim and J. P. Allebach, “Optimal unsharp mask for image sharpening and noise removal,” *J. Electron. Imag.*, vol. 14, no. 2, pp. 023 005–023 005–13, 2005.
- [39] G. Deng, “A generalized unsharp masking algorithm,” *IEEE Transactions on Image Processing*, vol. 20, no. 5, pp. 1249–1261, 2011.
- [40] P. Perona and J. Malik, “Scale-space and edge detection using anisotropic diffusion,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 7, pp. 629–639, 1990.

- [41] C. Tomasi and R. Manduchi, “Bilateral filtering for gray and color images,” in *Proceedings of the International Conference on Computer Vision*, 1998, pp. 839–846.
- [42] K. He, J. Sun, and X. Tang, “Guided image filtering,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 6, pp. 1397–1409, 2013.
- [43] E. S. L. Gastal and M. M. Oliveira, “Domain transform for edge-aware image and video processing,” *ACM Transactions on Graphics*, vol. 30, no. 4, pp. 69:1–69:12, 2011.
- [44] L. I. Rudin, S. Osher, and E. Fatemi, “Nonlinear total variation based noise removal algorithms,” *Physica D: Nonlinear Phenomena*, vol. 60, no. 1, pp. 259–268, 1992.
- [45] Z. Farbman, R. Fattal, D. Lischinski, and R. Szeliski, “Edge-preserving decompositions for multi-scale tone and detail manipulation,” in *ACM Transactions on Graphics*, vol. 27, no. 3, 2008, pp. 67:1–67:10.
- [46] L. Xu, C. Lu, Y. Xu, and J. Jia, “Image smoothing via l0 gradient minimization,” in *ACM Transactions on Graphics*, vol. 30, no. 6, 2011, pp. 174:1–174:12.
- [47] D. Min, S. Choi, J. Lu, B. Ham, K. Sohn, and M. N. Do, “Fast global image smoothing based on weighted least squares,” *IEEE Transactions on Image Processing*, vol. 23, no. 12, pp. 5638–5653, 2014.
- [48] K. Subr, C. Soler, and F. Durand, “Edge-preserving multiscale image decomposition based on local extrema,” *ACM Transactions on Graphics*, vol. 28, no. 5, pp. 147:1–147:9, 2009.

- [49] L. Xu, Q. Yan, Y. Xia, and J. Jia, “Structure extraction from texture via relative total variation,” *ACM Transactions on Graphics*, vol. 31, no. 6, pp. 139:1–139:10, 2012.
- [50] L. Karacan, E. Erdem, and A. Erdem, “Structure-preserving image smoothing via region covariances,” *ACM Transactions on Graphics*, vol. 32, no. 6, pp. 176:1–176:11, 2013.
- [51] L. Bao, Y. Song, Q. Yang, H. Yuan, and G. Wang, “Tree filtering: Efficient structure-preserving smoothing with a minimum spanning tree,” *IEEE Transactions on Image Processing*, vol. 23, no. 2, pp. 555–569, 2014.
- [52] Q. Zhang, X. Shen, L. Xu, and J. Jia, “Rolling guidance filter,” in *Proceedings of the European Conference on Computer Vision*, 2014, pp. 815–830.
- [53] B. Ham, M. Cho, and J. Ponce, “Robust image filtering using joint static and dynamic guidance,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 4823–4831.
- [54] D. Kiku, Y. Monno, M. Tanaka, and M. Okutomi, “Residual interpolation for color image demosaicking,” in *Proceedings of the IEEE International Conference on Image Processing*, 2013, pp. 2304–2308.
- [55] B. E. Bayer, “Color imaging array,” U.S. Patent 3971065, 1976.
- [56] D. Kiku, Y. Monno, M. Tanaka, and M. Okutomi, “Minimized-laplacian residual interpolation for color image demosaicking,” in *IS&T/SPIE Electronic Imaging*. International Society for Optics and Photonics, 2014, pp. 90 230L–90 230L.

- [57] B. Gunturk, Y. Altunbasak, and R. Mersereau, “Color plane interpolation using alternating projections,” *IEEE Transactions on Image Processing*, vol. 11, no. 9, pp. 997–1013, 2002.
- [58] E. Dubois, “Frequency-domain methods for demosaicking of bayer-sampled color images,” *IEEE Signal Processing Letters*, vol. 12, no. 12, pp. 847–850, 2005.
- [59] ———, “Filter design for adaptive frequency-domain bayer demosaicking,” in *Proceedings of the IEEE International Conference on Image Processing*, 2006, pp. 2705–2708.
- [60] B. Leung, G. Jeon, and E. Dubois, “Least-squares luma-chroma demultiplexing algorithm for bayer demosaicking,” *IEEE Transactions on Image Processing*, vol. 20, no. 7, pp. 1885–1894, 2011.
- [61] J. Mairal, M. Elad, and G. Sapiro, “Sparse representation for color image restoration,” *IEEE Transactions on Image Processing*, vol. 17, no. 1, pp. 53–69, 2008.
- [62] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman, “Non-local sparse models for image restoration,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2009, pp. 2272–2279.
- [63] A. A. Moghadam, M. Aghagolzadeh, M. Kumar, and H. Radha, “Compressive framework for demosaicing of natural images,” *IEEE Transactions on Image Processing*, vol. 22, no. 6, pp. 2356–2371, 2013.
- [64] B. Gunturk, J. Glotzbach, Y. Altunbasak, R. Schafer, and R. Mersereau, “Demosaicking: color filter array interpolation,” *IEEE Signal Processing Magazine*, vol. 22, no. 1, pp. 44–54, 2005.

- [65] A. Buades, B. Coll, and J. M. Morel, “A non-local algorithm for image denoising,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, 2005, pp. 60–65.
- [66] A. Buades, B. Coll, J. M. Morel, and C. Sbert, “Self-similarity driven color demosaicking,” *IEEE Transactions on Image Processing*, vol. 18, no. 6, pp. 1192–1202, 2009.
- [67] J. Duran and A. Buades, “Self-similarity and spectral correlation adaptive algorithm for color demosaicking,” *IEEE Transactions on Image Processing*, vol. 23, no. 9, pp. 4031–4040, 2014.
- [68] F. Zhang, X. Wu, X. Yang, W. Zhang, and D. Zhang, “Robust color demosaicking with adaptation to varying spectral correlations,” *IEEE Transactions on Image Processing*, vol. 18, no. 12, pp. 2706–2717, 2009.
- [69] S. Weisberg, *Applied linear regression*. John Wiley & Sons, 2005, vol. 528.
- [70] W. Ye and K.-K. Ma, “Image demosaicing by using iterative residual interpolation,” in *Proceedings of the IEEE International Conference on Image Processing*, 2014, pp. 1862–1866.
- [71] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image quality assessment: from error visibility to structural similarity,” *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.
- [72] H. Chang, D.-Y. Yeung, and Y. Xiong, “Super-resolution through neighbor embedding,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, 2004, pp. 275–282.

- [73] J. Yang, J. Wright, T. S. Huang, and Y. Ma, “Image super-resolution via sparse representation,” *IEEE Transactions on Image Processing*, vol. 19, no. 11, pp. 2861–2873, 2010.
- [74] W. Dong, D. Zhang, G. Shi, and X. Wu, “Image deblurring and super-resolution by adaptive sparse domain selection and adaptive regularization,” *IEEE Transactions on Image Processing*, vol. 20, no. 7, pp. 1838–1857, 2011.
- [75] J. Sun, J. Sun, Z. Xu, and H.-Y. Shum, “Gradient profile prior and its applications in image super-resolution and enhancement,” *IEEE Transactions on Image Processing*, vol. 20, no. 6, pp. 1529–1542, 2011.
- [76] R. Zeyde, M. Elad, and M. Protter, “On single image scale-up using sparse-representations,” in *Curves and Surfaces*. Springer, 2012, pp. 711–730.
- [77] C. Dong, C. C. Loy, K. He, and X. Tang, “Learning a deep convolutional network for image super-resolution,” in *Proceedings of the European Conference on Computer Vision*. Springer, 2014, pp. 184–199.
- [78] J.-J. Huang, W.-C. Siu, and T.-R. Liu, “Fast image interpolation via random forests,” *IEEE Transactions on Image Processing*, vol. 24, no. 10, pp. 3232–3245, 2015.
- [79] D. Glasner, S. Bagon, and M. Irani, “Super-resolution from a single image,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2009, pp. 349–356.
- [80] M. Protter, M. Elad, H. Takeda, and P. Milanfar, “Generalizing the nonlocal-means to super-resolution reconstruction,” *IEEE Transactions on Image Processing*, vol. 18, no. 1, pp. 36–51, 2009.

- [81] W. Dong, L. Zhang, R. Lukac, and G. Shi, “Sparse representation based image interpolation with nonlocal autoregressive modeling,” *IEEE Transactions on Image Processing*, vol. 22, no. 4, pp. 1382–1394, 2013.
- [82] R. Courant and D. Hilbert, *Methods of Mathematical Physics*. CUP Archive, 1966.
- [83] J. Weickert, *Anisotropic diffusion in image processing*. B. G. Teubner, Ed. Stuttgart, Germany, 1998.
- [84] D. Martin, C. Fowlkes, D. Tal, and J. Malik, “A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics,” in *Proceedings of the International Conference on Computer Vision*, vol. 2, 2001, pp. 416–423.
- [85] A. C. Bovik, *Handbook of image and video processing*. Academic press, 2010.
- [86] J. Shi, L. Xu, and J. Jia, “Just noticeable defocus blur detection and estimation,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 657–665.
- [87] D. Lischinski, Z. Farbman, M. Uyttendaele, and R. Szeliski, “Interactive local adjustment of tonal values,” *ACM Transactions on Graphics*, vol. 25, no. 3, pp. 646–653, 2006.
- [88] B. Zhang and J. P. Allebach, “Adaptive bilateral filter for sharpness enhancement and noise removal,” *IEEE Transactions on Image Processing*, vol. 17, no. 5, pp. 664–678, 2008.
- [89] J. H. Elder and S. W. Zucker, “Local scale control for edge detection and blur estimation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 7, pp. 699–716, 1998.

- [90] Y.-W. Tai and M. S. Brown, “Single image defocus map estimation using local contrast prior,” in *Proceedings of the IEEE International Conference on Image Processing*, 2009, pp. 1797–1800.
- [91] S. Zhuo and T. Sim, “Defocus map estimation from a single image,” *Pattern Recognition*, vol. 44, no. 9, pp. 1852–1858, 2011.
- [92] X. Zhu, S. Cohen, S. Schiller, and P. Milanfar, “Estimating spatially varying defocus blur from a single image,” *IEEE Transactions on Image Processing*, vol. 22, no. 12, pp. 4879–4891, 2013.
- [93] J. Shi, L. Xu, and J. Jia, “Discriminative blur detection features,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 2965–2972.
- [94] D. Ferstl, C. Reinbacher, R. Ranftl, M. Ruether, and H. Bischof, “Image guided depth upsampling using anisotropic total generalized variation,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 993–1000.
- [95] D. Comaniciu and P. Meer, “Mean shift: a robust approach toward feature space analysis,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, pp. 603–619, 2002.
- [96] Y. Saad, *Iterative methods for sparse linear systems*. Siam, 2003.
- [97] R. Szeliski, “Locally adapted hierarchical basis preconditioning,” vol. 25, no. 3, pp. 1135–1143, 2006.
- [98] J. Lu, K. Shi, D. Min, L. Lin, and M. N. Do, “Cross-based local multipoint filtering,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 430–437.

- [99] F. Durand and J. Dorsey, “Fast bilateral filtering for the display of high-dynamic-range images,” in *ACM Transactions on Graphics*, vol. 21, no. 3, 2002, pp. 257–266.
- [100] S. Paris and F. Durand, “A fast approximation of the bilateral filter using a signal processing approach,” in *Proceedings of the European Conference on Computer Vision*, 2006, pp. 568–580.
- [101] B. Weiss, “Fast median and bilateral filtering,” *ACM Transactions on Graphics*, vol. 25, no. 3, pp. 519–526, 2006.
- [102] F. Porikli, “Constant time $\mathcal{O}(1)$ bilateral filtering,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2008, pp. 1–8.
- [103] Q. Yang, K.-H. Tan, and N. Ahuja, “Real-time $\mathcal{O}(1)$ bilateral filtering,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 557–564.
- [104] J. Van de Weijer and R. Van den Boomgaard, “Local mode filtering,” in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, 2001, pp. II–428.
- [105] R. Fattal, “Edge-avoiding wavelets and their applications,” *ACM Transactions on Graphics*, vol. 28, no. 3, pp. 22:1–22:10, 2009.
- [106] Z. Ma, K. He, Y. Wei, J. Sun, and E. Wu, “Constant time weighted median filtering for stereo matching and beyond,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 49–56.
- [107] S. Paris, P. Kornprobst, J. Tumblin, and F. Durand, “A gentle introduction to bilateral filtering and its applications,” in *ACM SIGGRAPH Courses*, 2007.

- [108] S. Paris, “Edge-preserving smoothing and mean-shift segmentation of video streams,” in *Proceedings of the European Conference on Computer Vision*, 2008, pp. 460–473.
- [109] S. Bae, S. Paris, and F. Durand, “Two-scale tone management for photographic look,” in *ACM Transactions on Graphics*, vol. 25, no. 3, 2006, pp. 637–645.
- [110] G. Petschnigg, R. Szeliski, M. Agrawala, M. Cohen, H. Hoppe, and K. Toyama, “Digital photography with flash and no-flash image pairs,” *ACM Transactions on Graphics*, vol. 23, no. 3, pp. 664–672, 2004.
- [111] A. Levin, D. Lischinski, and Y. Weiss, “Colorization using optimization,” in *ACM Transactions on Graphics*, vol. 23, no. 3, 2004, pp. 689–694.
- [112] ———, “A closed-form solution to natural image matting,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 2, pp. 228–242, 2008.
- [113] H. Cho, H. Lee, H. Kang, and S. Lee, “Bilateral texture filtering,” *ACM Transactions on Graphics*, vol. 33, no. 4, pp. 128:1–128:8, 2014.
- [114] F. Zhang, L. Dai, S. Xiang, and X. Zhang, “Segment graph based image filtering: Fast structure-preserving smoothing,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 361–369.
- [115] R. Chartrand and W. Yin, “Iteratively reweighted algorithms for compressive sensing,” in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2008, pp. 3869–3872.
- [116] J. Kopf and D. Lischinski, “Digital reconstruction of halftoned color comics,” *ACM Transactions on Graphics*, vol. 31, no. 6, pp. 140:1–140:10, 2012.

- [117] J. Canny, “A computational approach to edge detection,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, no. 6, pp. 679–698, 1986.
- [118] P. E. Debevec and J. Malik, “Recovering high dynamic range radiance maps from photographs,” in *Proceedings of the Annual Conference on Computer Graphics and Interactive Techniques*, 1997, pp. 369–378.
- [119] S. B. Kang, M. Uyttendaele, S. Winder, and R. Szeliski, “High dynamic range video,” *ACM Transactions on Graphics*, vol. 22, no. 3, pp. 319–325, 2003.
- [120] K. Jacobs, C. Loscos, and G. Ward, “Automatic high-dynamic range image generation for dynamic scenes,” *IEEE Computer Graphics and Applications*, vol. 28, no. 2, pp. 84–93, 2008.
- [121] X. Wu and L. Zhang, “Improvement of color video demosaicking in temporal domain,” *IEEE Transactions on Image Processing*, vol. 15, no. 10, pp. 3138–3151, 2006.
- [122] S. Farsiu, M. D. Robinson, M. Elad, and P. Milanfar, “Fast and robust multiframe super resolution,” *IEEE Transactions on Image Processing*, vol. 13, no. 10, pp. 1327–1344, 2004.
- [123] E. P. Bennett and L. McMillan, “Video enhancement using per-pixel virtual exposures,” *ACM Transactions on Graphics*, vol. 24, no. 3, pp. 845–852, 2005.