The Hong Kong Polytechnic University

Department of Electronic and Information Engineering

On the Restoration and Formation of Color-Quantized Images

Fung Yik-Hing

A thesis submitted in partial fulfillment of the requirements for the

Degree of Doctor of Philosophy

November 2005

CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgement has been made in the text.

_____ (Signed)

FUNG YIK HING (Name of student)

To my parents

Abstract

Color quantization is an image processing technique which reduces the number of colors used to represent an image at a minimum quality penalty. It has been widely used in a number of applications such as displaying an image in a low-end display, delivering images over the Internet and producing hardcopies of color images.

To a certain extent, color quantization can be considered as a lossy compression process in which bits per pixel are reduced. A color-quantized image cannot be perfectly reconstructed so as to produce its original version. The restoration of a color-quantized image is hence necessary. Image restoration is a field in which people concern the reconstruction or estimation of an uncorrupted image from its distorted version. Conventional restoration algorithms are dedicated for restoring noisy blurred images. Direct application of existing restoration techniques are generally inadequate to deal with the restoration of compressed images since the degradation models of the two cases are completely different. Though some dedicated algorithms for restoring JPEG-encoded images and halftoned images have been proposed recently, little effort has been seen in the literature for restoring color-quantized images.

This thesis presents four novel algorithms for restoring color-quantized images. When an image is color-quantized, a palette is used to define the available colors that can be produced in the output. The smaller the palette size, the more artefacts are introduced in the output. Halftoning is an image processing technique which reduces these artefacts by making use of the property of our human visual system.

In the four presented restoration algorithms, two of them were proposed for restoring color-quantized images in which no error diffusion are involved and the other two handle the case when error diffusion is involved. These algorithms were developed independently by tackling the technical problems with different techniques including Regularization, Projection onto Convex Sets (POCS) and Simulated Annealing (SA) separately. All these algorithms make a good use of the available color palette to derive useful *a priori* information for restoration.

As mentioned before, color quantization is commonly used in printing applications to produce high quality hardcopies of color images. In this particular application, without lose of generality, a full color image is decomposed into three color planes and each plane is halftoned with binary halftoning algorithm independently. Multiscale error diffusion is a recently proposed halftoning technique. This technique was known to be superior to conventional halftoning techniques such as error diffusion by eliminating directional hysteresis completely. To evaluate its performance and explore its application in producing color hardcopies, a detailed analysis on multiscale error diffusion was carried out. This thesis presents a report of our analysis.

In order to efficiently render halftone images for various printers and displays which support different resolutions, it is desirable that halftoning results of different scales can be produced at a time and all of them can be embedded in a single full-scale halftone image such that a simple down-sampling process can extract images of suitable resolutions from the full-scale halftone image if necessary. This scalable property was addressed in this thesis. Based on the aforementioned analysis on multiscale error diffusion, we extended its idea and proposed a binary scalable

multiresolution halftoning algorithm. This algorithm can produce scalable color prints by handling different color planes separately.

With the rapidly evolving computer and communication technologies, the Internet has become the most popular media for information exchange among remote sites throughout the world. There are many forms of information available over the Internet today and digital color images are very popular among them. Images are compressed before its delivery over the Internet, which results in different output image file formats. GIF format is a popular image format generated with color quantization. Since networks of different channel bandwidth and clients of different capability scattered over the Internet, it is desirable to make the color-quantized materials scalable so as to save the channel bandwidth and other resources.

A multiscale vector error diffusion algorithm for color quantization was proposed in this thesis. Unlike those halftoning algorithms for printing color images, the proposed algorithm does not handle color planes separately and is able to handle color quantization using any arbitrary palettes. This proposed algorithm was used as a framework for generating scalable color-indexed images and a multiscale multiresolution vector error diffusion algorithm was proposed accordingly. Images possessing this scalable property support transmission over the Internet which contains clients with different display resolutions, system with different caching resources and networks with varying bandwidths and QoS capabilities.

Author's Publications

(List of publications of the author on which this thesis is based)

Book Chapter

1. Y. H. Fung, Y. H. Chan, W. C. Siu and S. O. Choy, "Restoring images with regularization in transform domain," *Recent Developments in Theories and Numerics: International Conference on Inverse Problems*, 2003 World Scientific Publishing Co. Pte. Ltd., pp.325-335.

International Journal Papers

1. Y. H. Fung and Y. H. Chan, "A POCS-based Algorithm for Restoring Colour-Quantized Images," IEE Proceeding – Vision, Image and Signal Processing, Vol.151, No.2, Apr 2004, pp.119-127.
2. Y. H. Chan and Y. H. Fung, "A regularized constrained iterative restoration algorithm for restoring color-quantized images," Signal Processing Vol.85, No.7, Jul 2005, pp.1375-1387.
3. Y. H. Fung and Y. H. Chan, "A POCS-based Restoration Algorithm for Restoring Halftoned Color-Quantized Images," accepted, to appear in IEEE Transactions on Image Processing.
4. Y. H. Fung and Y. H. Chan, "Embedding halftones of different resolutions in a full-scale halftone," IEEE Signal Processing Letters, Vol.3, No.3, Mar 2006, pp. 153-156.
5. Y. H. Fung and Y. H. Chan, "A simulated annealing restoration algorithm for restoring halftoned color-quantized images," accepted, to appear in Signal Processing: Image Communication.
6. Y. H. Fung and Y. H. Chan, "A technique for producing scalable color-quantized images with error diffusion," accepted, to appear in IEEE Transactions on Image Processing.
7. Y. H. Fung and Y. H. Chan, "Halftoning with Multiscale Colour Error diffusion for Colour-indexed Displays," submitted for possible publication in IEE Proceedings - Vision, Image and Signal Processing.

International Conference Papers

1. Y. H. Fung and Y. H. Chan, “An improved algorithm for removing impulse noise based on long-range correlation in an image,” Proceedings, IEEE International Conference on Multimedia and Expo (ICME'02), Aug 26-29, 2002, Lausanne, Switzerland, Vol.1, pp.157-160.
2. Y. H. Chan and Y. H. Fung, “Regularized Restoration of Color-Quantized Images,” Proceedings, Eusipco’2002, XI European Signal Processing Conference, Sep 3-6, 2002, Toulouse, France, Vol.3, pp.279-282.
3. Y. H. Fung and Y. H. Chan, “An Iterative Algorithm for Restoring Color-Quantized Images,” Proceedings, IEEE International Conference on Image Processing (ICIP'02), Rochester, New York, Sep 22-25, 2002, Vol.I, pp.313-316.
4. K. H. Chung, Y. H. Fung and Y. H. Chan, “Image enlargement with fractal,” Proceedings, IEEE ICASSP'03, Apr 6-10, 2003, Hong Kong, Vol.VI, pp.273-276.
5. Y. H. Fung and Y. H. Chan, “Restoring halftoned color-quantized images with simulated annealing,” Proceedings, IEEE International Conference on Multimedia and Expo (ICME 2004), Taipei, Taiwan, Jun 27-30, 2004, pp.367-370.
6. K. C. Liu, Y. H. Fung and Y. H. Chan, “Restoring halftoned color-quantized image with genetic algorithms,” Proceedings, 2004 International Symposium on Intelligent Multimedia, Video and Speech Processing, Oct 20-22, 2004 Hong Kong, pp.202-205.
7. Y. H. Fung and Y. H. Chan, “Fingerprint recognition with improved wavelet domain features,” Proceedings, 2004 International Symposium on Intelligent Multimedia, Video and Speech Processing, Oct 20-22, 2004 Hong Kong, pp.33-36.
8. Y. H. Fung and Y. H. Chan, “Restoration of halftoned color-quantized images using projection onto convex sets,” Proceedings, IEEE International Conference on Image Processing (ICIP’ 04), Singapore, Oct 24-27, 2004, pp.325-328.
9. Y. H. Fung, Y. H. Chan and K. C. Lui, “A Multiscale Color Quantization Algorithm for producing scalable media,” accepted to appear in EUSIPCO 2005.
10. Y. H. Fung and Y. H. Chan, “A Multiscale Color Error Diffusion Algorithm for Color Quantization,” accepted, to appear 2005 International Symposium on

Intelligent Signal Processing and Communication Systems (ISPAC2005), December 13-16, 2005, Hong Kong.

11. K. H. Chung, Y. H. Fung, K. C. Lui and Y. H. Chan, “A low-complexity Multiscale Error Diffusion Algorithm for Digital Halftoning,” accepted, to appear 2005 International Symposium on Intelligent Signal Processing and Communication Systems (ISPAC2005), December 13-16, 2005, Hong Kong.
12. Neil B. K. W. Yu, Y. H. Fung and Y. H. Chan, "A study on the performance of multiscale error diffusion algorithms," 2005 Asia-Pacific Workshop on Visual Information Processing (VIP2005), 11-13 Dec 2005, Hong Kong

Acknowledgements

I would like to take this opportunity to express my sincere gratitude to my supervisor, Dr. Chris Yuk-Hee Chan, for his patient guidance, professional advice, continuous support and encouragement, concerning to my study and the attitude of my life. It is a rare privilege and a great honour for me to work with him. I benefited greatly from the countless discussions with him on every stage of the study.

I would also like to thank my friends and colleagues for their encouragement during the course of my study. The countless discussions with them have been proved to be fruitful and inspiring.

Thanks are also to all members of staff of the Department of Electronic and Information Engineering. Financial support provided by the Research Office of The Hong Kong Polytechnic University of the Research Grants Council of the Hong Kong Special Administrative Region throughout the entire period of candidature is greatly appreciated, which made possible this research.

Last, but not least, I must express my heartfelt gratitude to my parents, my sisters and Miss. P. Y. Cheung for their understanding, endless support and encouragement.

Table of Contents

CERTIFICATE OF ORIGINALITY ii

Abstract…….. iv

Author's Publications…… vii

Acknowledgements…… x

Table of Contents…….. xi

List of Figures…… xvii

List of Tables…….. xxii

List of Abbreviations…… xxvi

Statement of Originality…… xxvii

Chapter 1. Introduction……. **1**

1.1. Background.…..…… 1

1.2. Problems Addressed……..…….…….. 3

1.2.1. Restoration of Color-quantized Images…… 4

1.2.2. Formation of Scalable Color Prints……...……..….. 5

1.2.3. Formulation of Halftoned Color-quantized Images…….. 7

1.3. Organization of the Thesis…… 8

Chapter 2. A Comprehensive Literature Review…… **10**

2.1. Introduction……... 10

2.2. Image Restoration……. 10

2.2.1 Image Restoration Techniques…….. 13

2.2.1.1 Miller Regularization…….. 13

2.2.1.2 Projections onto Convex Sets (POCS)…… 14

2.2.1.3 Simulated Annealing (SA)…….. 16

2.3. Digital Halftoning and Its Analysis..…… 16

2.3.1. Analytical Tools……..…….. 17

2.3.2. Digital Halftoning Methods…….. 19

2.3.2.1. Standard Error Diffusion Algorithm..…….…… 19

2.3.2.2. Adaptive Error Diffusion Algorithm…….. 21

2.3.2.3. Multiscale Error Diffusion Algorithm…… 21

2.4 Color Quantization…… 23

2.4.1. Color Quantization Methods……. 25

2.4.1.1. Color Quantization of Images……. 25

2.4.1.1.1 Color Quantization Without Error Diffusion……. 26

2.4.1.1.2 Color Quantization With Error Diffusion……...…….. 26

2.4.1.2. Adaptive Method for Dithering Color Images……..... 28

2.4.1.3. Fuzzy Error Diffusion……. 29

Chapter 3. A Constrained Least Square Restoration Algorithm for Restoring Color-quantized Images…… 30

3.1. Introduction…….…….. 30

3.2. Formulation of *A Priori* Information…… 31

3.3. Formulation of Constraints for Restoration…….. 34

3.4. Formulation of Restoration Algorithm……. 36

3.5. Performance Evaluation…… 39

3.5.1 Realization Details of the Proposed Algorithm……. 39

3.5.2 Algorithms for Comparative Study…… 40

3.5.3 Simulation Results……. 42

3.6. Summary…...…… 51

Chapter 4. A POCS-based Algorithm for Restoring Color-quantized Images 53

4.1. Introduction……... 53

4.2. Formulation of A *Priori* Information…….…….. 53

4.3. Formulation of Constraints for Restoration..…… 54

4.4. Formulation of the POCS Algorithm……...…. 56

4.5. Simulation and Comparative Study…….. 57

4.5.1 Realization Details of the Proposed Algorithm……. 58

4.5.2 Algorithms for Comparative Study..…….. 64

4.5.3 Simulation Results and Implications……. 64

4.6. Summary…..……. 66

Chapter 5. A POCS-based Restoration Algorithm for Restoring Halftoned Color-quantized Images……. 67

5.1. Introduction……... 67

5.2. Formulation of A Priori Information……...……. 70

5.3. Formulation of POCS Algorithm…….……..…… 75

5.4. Simulation and Comparative Study……..…… 77

5.4.1 Realization Details of the Proposed Algorithm……. 77

5.4.2 Algorithms for Comparative Study....…… 78

5.4.3 Simulation Results....……. 78

5.4.4 Robustness Study……...…..…… 83

5.5. Summary..……. 90

Chapter 6. A Simulated Annealing Restoration Algorithm for Restoring Halftoned Color-quantized Images……... 91

6.1. Introduction……... 91

6.2. Proposed Simulated Annealing Restoration Algorithm……...……. 92

6.3. Performance Evaluation and Comparative Study……. 94

6.3.1 Realization Details of the Proposed Algorithm……. 94

6.3.2 Simulation Results……. 98

6.4. Summary…...……. 106

Chapter 7. On the Production of Scalable Color-quantized Images for Printing Purpose……....…. 108

7.1. Introduction……... 108

7.2. Proposed Framework for Generating Scalable Binary Halftones with MED Algorithms...…… 109

7.3. Picking an Appropriate MED……... 112

7.3.1 Performance Analysis on MED Algorithms..……..... 112

7.3.2 Performance Analysis on Constrained MED Algorithms…..… 119

7.4. Comparative Study on the Performance of different Constrainted Halftoning Algorithms.……...…….. 124

7.5. Summary...…… 129

Chapter 8. On the Production of General Scalable Halftoned Color-quantized Images.…… 130

8.1. Introduction……... 130

8.2. Multiscale Color Error Diffusion Algorithm…..…….. 132

8.2.1. Constructing Energy Pyramid **E** …….. 133

8.2.2. Searching the Pixel for Color Quantization…….. 134

8.2.3. Color Quantization and Error Diffusion……... 134

8.3. Extension to Produce Scalable Halftoned Color-quantized Images…. 135

8.4. Performance on Multiscale Color Error Diffusion……... 137

8.4.1 Details of the Simulation……... 137

8.4.2 Simulation Results…… 138

8.5. Performance on Generating Scalable Halftoned Color-quantized 146

Images…..……..……

8.5.1 Details of the Simulation…….. 146

8.5.2 Simulation Results…… 147

8.6. Summary……..……..……. 157

Chapter 9. Conclusions……. 158

9.1 The Work Done……. 158

9.2 Future Works……. 161

Appendix A Testing Images……. 162

Bibliography …….. 165

List of Figures

Figure 2.1. Linear degradation model……....……. 11

Figure 2.2. Projections among different closed convex constraint sets…….….. 15

Figure 2.3. The system of error diffusion……..……..… 19

Figure 2.4. Color quantization without error diffusion.……..…...…….. 26

Figure 2.5. Color quantization with error diffusion.……..…...……... 27

Figure 3.1. The local coordinate system associated with a Voronoi region R_k .. 31

Figure 3.2. Zoomed outputs of various approaches in restoring color-quantized *Cycles*……..…..…… 49

Figure 4.1. Zoomed outputs of various approaches in restoring color-quantized Fruits……...……...…… 62

Figure 5.1. How a cross-section of a color space is partitioned by a color palette: (a) in a color halftone generated for printing applications; (b) in general color quantization……...……. 69

Figure 5.2. Projection of $\vec{\mathbf{U}}'_{(m,n)}$ onto R_k' ……..…… 72

Figure 5.3. Examples of how pixels of an estimate of the original image are updated……... 74

Figure 5.4. Restoration results of color-quantized “Caps” (palette is generated by median-cut algorithm [Heckbert 82] with size 64)……..…..…… 85

Figure 5.5. Restoration results of color-quantized “Parrots” (palette is generated by Octree algorithm [Gervautz 90] with size 128).……... 87

Figure 6.1. SNR Improvements achieved with various combinations of γ and M in restoring halftoned color-quantized (a) “Lenna “, (b) “Peppers” and (c) “Couple”…… ……..…. 97

Figure 6.2. Restored halftoned color-quantized “Fruits” of various approaches (palette is generated by median-cut algorithm [Heckbert 82] with size 128)…….…… 102

Figure 7.1. Halftoning results of (a) PED, (b) MED_k, (c) MED_c98 and (d) MED_c04 for constant gray-level input (13/255) of size 128x128……..…… 113

Figure 7.2. Halftoning results of (a) PED, (b) MED_k, (c) MED_c98 and (d) MED_c04 for constant gray-level input (95/255) of size 128x128…….. 113

Figure 7.3. Performance in terms of Number of dots…….…….……. 114

Figure 7.4. Corresponding directional distribution functions of Figure 7.1…..… 114

Figure 7.5. Corresponding directional distribution functions of Figure 7.2…… 115

Figure 7.6. Performance in terms of directional distribution of dots…..…...….. 116

Figure 7.7. Performance in terms of anisotropy (a) PED, (b) MED_k and (c) MED_c98 and (d) MED_c04……..……. 117

Figure 7.8. Performance in terms of RAPSD (a) PED, (b) MED_k, (c) MED_c98 and (d) MED_c04……..…… 118

Figure 7.9. Halftoning results of (a) CH-SED, (b) CH-AED, (c) CH_p-MED and (d) the proposed algorithm for constant gray-level input (31/255) of size 128x128: (i) full-scale outputs, (ii) down-sampling results of (i) and (iii) directional distributions…….……. 120

Figure 7.10. Halftoning results of (a) CH-SED, (b) CH-AED, (c) CH_p-MED and (d) the proposed algorithm for constant gray-level input (88/255) of size 128x128: (i) full-scale outputs, (ii) down-sampling results of (i) and (iii) directional distributions…….……. 121

Figure 7.11. (a) Original ramp image and halftone results of (b) [Floyd 76], (c) CH-SED, (d) CH-AED, (e) CH_p-MED and (f) the proposed algorithm…..…….. 124

Figure 7.12. Down-sampling results of Figures 8.2a-8.2f…….…..……... 124

Figure 7.13. Down-sampling results of Figures 8.2a-8.2f…….…..……... 125

Figure 7.14. Performance in terms of directional distribution of dots of full-scale halftones…….…..…….……. 125

Figure 7.15. Performance in terms of directional distribution of dots of down-sampled halftones……... 126

Figure 7.16. (a) Original image “Parrots” and halftone results of (b) [Floyd 76], (c) CH-SED, (d) CH-AED, (e) CH_p-MED and (f) the proposed algorithm……..... 127

Figure 7.17. Down-sampling results of Figures 7.16a-7.16f…….. 128

Figure 8.1. Color quantization results of “Pool” (Palette size = 16): (a) Original, (b) [Orchard 91], (c) [Akarun 97], (d) [Özdemir 00], (e) [Breaux 99], (f) the proposed algorithm and (f) the proposed block-based algorithm…….....…. 142

Figure 8.2. Color quantization results of “Caps” (Palette size = 32): (a) Original, (b) [Orchard 91], (c) [Akarun 97], (d) [Özdemir 00], (e) [Breaux 99], (f) the proposed algorithm and (g) the proposed block-based algorithm……... 144

Figure 8.3. Color quantization results of full-scaled “Caps” (palette size = 64): (a) Original (b) [Orchard 91], (c) [Özdemir 00], (d) [Akarun 97], (e) proposed algorithm……..……..……. 153

Figure 8.4. Color quantization results of down-sampled versions of “Caps” with s_r =2 (Palettes were generated with median-cut algorithm [Heckbert 82] of size 64): (a) Original (b) [Orchard 91], (c) [Özdemir 00], (d) [Akarun 97], (e) proposed algorithm…… 154

Figure 8.5. Color quantization results of down-sampled versions of “Caps” with s_r =3 (Palettes were generated with median-cut algorithm [Heckbert 82] of size 64): (a) Original (b) [Orchard 91], (c) [Özdemir 00], (d) [Akarun 97], (e) proposed algorithm…… 155

Figure 8.6. Color quantization results of down-sampled versions of “Caps” with s_r =4 (Palettes were generated with median-cut algorithm [Heckbert 82] of size 64): (a) Original (b) [Orchard 91], (c) [Özdemir 00], (d) [Akarun 97], (e) proposed algorithm…… 156

List of Tables

Table 3.1. SNR Improvements of various algorithms in restoring images color-quantized with (a) median cut algorithm [Heckbert 82], (b) octree algorithm [Gervautz 90]……..…… 45

Table 3.2. CIELAB difference ΔE measurement of outputs of various algorithms in restoring images color-quantized with (a) median cut algorithm [Heckbert 82], (b) octree algorithm [Gervautz 90]……. 46

Table 3.3. Percentage of pixels whose CIELAB difference is not detectable (ΔE <3) after restoration when testing images were color-quantized with (a) median cut algorithm [Heckbert 82], (b) octree algorithm [Gervautz 90]……..……. 47

Table 3.4. Discrepancy measured when $\bar{\sigma}_k$ is evaluated with the blurred observed image instead of the original image……..…….. 51

Table 4.1. SNR Improvements of various algorithms in restoring images color-quantized with (a) median cut algorithm [Heckbert 82], (b) octree algorithm [Gervautz 90]……..…….…… 59

Table 4.2. CIELAB difference ΔE measurement of outputs of various algorithms in restoring images color-quantized with (a) median cut algoroithm [Heckbert 82], (b) octree algorithm [Gervautz 90]…….. 60

Table 4.3. Percentage of pixels whose CIELAB difference is not detectable (ΔE <3) after restoration when testing images were color-quantized with (a) median cut algorithm [Heckbert 82], (b) octree algorithm [Gervautz 90]…….. 61

Table 5.1. SNR Improvements of various algorithms in restoring halftoned color-quantized images……..……. 80

Table 5.2. CIELAB difference ΔE measurement of the outputs of various algorithms in restoring halftoned color-quantized images…….. 81

Table 5.3. Percentage of pixels whose CIELAB difference is not detectable ($\Delta E < 3$) after restoration…….…… 82

Table 5.4. Average performance of various algorithms in restoring halftoned color-quantized images in various aspects (Palette was generated with median-cut algorithm [Heckbert 82])……. 84

Table 5.5. Average performance of the proposed algorithm when different error diffusion filters are assumed in restoration (Palette was generated with median-cut algorithm [Heckbert 82])……..……. 89

Table 6.1. SNR Improvements of various algorithms in restoring halftoned color-quantized images with the palette generated by median-cut algorithm [Heckbert 82]……. 99

Table 6.2. CIELAB difference ΔE measurement of the outputs of various algorithms in restoring haftoned color-quantized images with the palette generated by median-cut algorithm [Heckbert 82] of size….. 100

Table 6.3. Percentage of pixels whose CIELAB difference is not detectable ($\Delta E < 3$) after restoration when testing images were color-quantized with the palette generated by median-cut algorithm [Heckbert 82] of size…….. 101

Table 6.4. Average performance of various algorithms in restoring halftoned color-quantized images in various aspects (palette was generated by the octree algorithm [Gervautz 90])……..……. 104

Table 7.1. Summary of the algorithms evaluated for comparison……..……. 123

Table 8.1. Average of S-CIELab difference ($\Delta\mathbf{E}$) metric of the halftoned color-quantized outputs of various algorithms (Palettes were generated with median-cut algorithm [Heckbert 82].)…… 140

Table 8.2. Average of S-CIELab difference ($\Delta\mathbf{E}$) metric of the halftoned color-quantized outputs of various algorithms (Palettes were generated with octree algorithm [Gervautz 90].)……..……. 141

Table 8.3. Average of S-CIELAB color difference ($\Delta\mathbf{E}$) metric of the halftoned color-quantized outputs of various algorithms with s_r = 1,2,3 and 4. (Palettes were generated with median-cut algorithm [Heckbert 82] of size 16.)……..……. 149

Table 8.4. Average of S-CIELAB color difference ($\Delta\mathbf{E}$) metric of the halftoned color-quantized outputs of various algorithms with s_r = 1,2,3 and 4. (Palettes were generated with median-cut algorithm [Heckbert 82] of size 32.)……..……. 150

Table 8.5. Average of S-CIELAB color difference ($\Delta\mathbf{E}$) metric of the halftoned color-quantized outputs of various algorithms with s_r = 1,2,3 and 4. (Palettes were generated with median-cut algorithm [Heckbert 82] of size 64.)…….. 151

Table 8.6. Average of S-CIELAB color difference ($\Delta\mathbf{E}$) metric of the halftoned color-quantized outputs of various algorithms with s_r = 1,2,3 and 4. (Palettes were generated with median-cut algorithm [Heckbert 82] of size 128.)…….…… 152

List of Abbreviations

The following list summarizes the abbreviations used throughout this thesis.

2D	Two Dimensional
3D	Three Dimensional
GIF	Graphics Interchange Format
HVS	Human Visual System
JPEG	Joint Photographers Expert Group
KL	Karhunen-Loeve
LMS	Least mean square
MSE	Mean Square Error
NWE	Normalized Weighted Error
POCS	Projection onto Convex Sets
PSF	Point Spread Function
QoS	Quality of Services
RAPSD	Radially Averaged Power Spectrum Density
SA	Simulated Annealing
SNRI	Signal-to-Noise Ratio Improvement
VGA	Video Graphics Array
CH	Constrained Halftoning

Statement of Originality

The following contributions reported in this thesis are claimed to be original

1. Restoration of color-quantized images is rarely addressed in literature, and direct applications of existing restoration techniques are generally inadequate to deal with the problem. The degradation process of color quantization in which no error diffusion is involved is formulated and *a priori* information about the original image is devised in Chapter 3, Section 3.2.

2. Based on the *a priori* information we obtained, a restoration algorithm is specially devised by using the regularization theory in Chapter 3, Section 3.4 for restoring color-quantized images in which error diffusion is not involved. It has been demonstrated by the detailed simulation results that the algorithm can achieve a remarkable improvement.

3. A POCS-based restoration algorithm is developed in Chapter 4, Section 4.4 to restore color-quantized images in which error diffusion is not involved. It was shown by simulation results that the algorithm could improve the quality of the restored images as compared with other existing restoration approaches.

4. A novel restoration algorithm is proposed in Chapter 5, Section 5.3 for restoring images which were color-quantized with error diffusion. The proposed algorithm is based on the POCS theory. The algorithm makes use of the available color palette and the mechanism of a error diffusion process to derive useful *a priori* information for restoration. It was

demonstrated by simulation results that it could improve the quality of a halftoned color-quantized image remarkably.

5. The noise introduced by color quantization in which error diffusion is involved is basically signal dependent and is not white. This violates the assumptions adopted in most of the current multichannel restoration algorithms. A dedicated restoration algorithm for restoring halftoned color-quantized images is developed in Chapter 6, Section 6.2. The developed algorithm is based on simulated annealing which very well on solving discrete optimization problems.

6. A constrained halftoning framework for multiscale error diffusion algorithms to generate scalable color halftones for printing applications is proposed in Chapter 7, Section 7.2. The proposed algorithm can produce a set of color prints of different resolutions and embeds those of lower resolutions into the full-scale color print. It was shown in the detailed simulation results that the proposed algorithm can provide good color prints at different resolutions.

7. A multiscale vector error diffusion algorithm for color quantization is proposed in Chapter 8, Section 8.2. This does not handle color planes separately and is able to handle color quantization in which any arbitrary palettes can be used. The proposed algorithm can completely eliminate directional hysteresis and reduce more color impulses as compared with other algorithms. It is demonstrated in the detailed simulation results that better image quality of halftoned color-quantized image can be obtained.

8. A color quantization algorithm for generating scalable color-indexed images was proposed in Chapter 8, Section 8.3 which is based on a multiscale vector error diffusion framework proposed in Chapter 8, Section 8.2. Images of lower resolutions are embedded in the outputs such that a simple down-sampling process can extract images of any desirable resolutions. It was shown in the simulation results that the algorithm can produce a high-quality directional-hysteresis free output and simultaneously embed a set of color quantization results of the downsampled versions.

Chapter 1.

Introduction

1.1. Background

Color quantization is the process of reducing the number of colors in a digital image by replacing them with a representative color selected from a palette [Orchard 91]. It is widely used nowadays as it lessens the burden of massive image data on storage and transmission bandwidth in many multimedia applications and the palette involved is usually image dependent and can be of any arbitrary size and contain arbitrary colors.

In color quantization, each pixel vector of full color image is compared with a set of representative color vectors which are stored in a previously generated color palette. The best-matching color is then selected based on a criterion, which is usually the minimum Euclidean distance criterion, to represent the input full color vector. Once the best-matching colors for all pixel vectors of the source image have been selected from the color palette, the indices of the selected colors are transmitted to the receiver with the color palette. At the receiver, with the same color palette, the color-quantized image can be reconstructed based on the received indices.

When color quantization is performed, certain types of degradation are introduced due to the limited colors used to produce the output image. The most common artefact is false contour. False contour occurs when the available palette colors are not sufficient to represent a gradually changing region. Another common artefact is color shift. In general, the smaller the color palette size, the more severe the defects are.

Digital halftoning [Ulichney 91, Lau 01] would be helpful to eliminate these defects by making use of the fact that human eyes act as spatial low-pass filters. During color quantization, the quantization error of a pixel is diffused to neighboring pixels so as to hide the defects and to achieve a more faithful reproduction of colors. At the moment, the most popular halftoning method is error diffusion and several well-known error diffusion filters such as Floyd–Steinberg filter [Floyd 76], Jarvis-Judice-Ninke filter [Jarvis 76] and Stucki filter [Stucki 81] are generally used to achieve the goal.

Applications of color quantization can be found in three major areas. They are (1) displaying a color image with a low-end display device, (2) delivering color images over the Internet and (3) producing hardcopies of color images.

To reduce its hardware cost, a display device may want to reduce the size of its high-speed display memory buffer or limit the color resolution of its digital-to-analog converter. In both cases, it limits the number of displayable colors of the display device. Color quantization reduces the number of colors at a minimum cost of image quality to make a color image displayable.

With the rapidly evolving computer and communication technologies, the Internet has become the most popular media for exchanging information among near

and remote sites throughout the world. There are many forms of information available over the Internet today and digital color image is very popular. Besides, the increasing demand of large image databases would lead to the requirement of efficient storage and rapid transmission of digital color images. In order to reduce the bandwidth and the storage requirements, images are usually compressed before being transmitted over the Internet and stored in the database. The GIF format is a popular image format generated with color quantization and has been widely used in a number of Internet applications. It is because it inherits the advantage that the decompression process of a compressed image is just a simple index mapping with the given palette and it can faithfully report the color of the source image as the image producer designs in most cases.

Color quantization is also commonly used in a printing system to produce high quality hardcopies of color images. A printer cannot produce a multi-color image due to its printing mechanism. The number of colors that a printer can support depends on the number of its available ink cartridges. Color quantization is hence required to reduce the colors when printing a color image. In practice, a true color image is first decomposed into three-color planes and each plane is halftoned with a binary halftoning algorithm independently. By incorporate digital halftoning in color quantization, the hardcopy of a color image can be emulated by printing three different halftoned binary color planes separately.

1.2 Problems Addressed

Four main issues on color quantization are addressed in this thesis. Specifically, they are 1) Restoration of color-quantized images in which no error diffusion is involved, 2) Restoration of color-quantized images in which error

diffusion is involved, 3) Formation of scalable color prints and 4) Formation of halftoned color-quantized images and scalable halftoned color-indexed images

1.2.1 Restoration of Color-quantized Images

Color quantization introduces distortion and hence is a kind of degradation to the original full-color image. Sometimes it is necessary to recover the original image from its color-quantized version. This is especially true when the color-quantized version is required to be further processed or compressed.

Processing a color-quantized image is different from processing a continuous tone image because even a simple process may cause severe degradation to the color-quantized image. For instance, directly downsampling a halftoned color-quantized image would result in a very poor output while a much better output can be obtained with a true color image. A restored color-quantized image is closer to the original image and can also provide a better processing result as compared with the color-quantized image.

Another example is color palette conversion. A user may want to render a color-quantized image for display units of different color resolutions. Color quantization is then required to convert a color-quantized image into another color-quantized image in which a switch of color palettes is involved. Both the content and the size can be different in both palettes. To have a new color palette for the color quantization, it would be better to restore the color-quantized image into its original first and then color-quantize it again afterward to suit for a particular application.

Though there are a lot of reported works on the restoration of noisy and blurred color images [Altunbasak 01, Angelopoulos 94, Barni 00, Galatsanos 89, Galatsanos 91a, Galatsanos 91b, Hunt 84, Kaulgud 99], little effort has been seen in

the literature for restoring color-quantized images and halftoned color-quantized images. Obviously, their degradation models are different from that of blurring and additive noise. The degradation model of color quantization is nonlinear as the quantization process involved is basically a nonlinear error source. Hence, direct adoption of conventional restoration algorithms does not work effectively. In order to have a good restoration performance, dedicated restoration algorithm should be developed.

In this thesis, restoration of color-quantized images and halftoned color-quantized images are addressed differently as their degradation models are different. Various approaches are exploited to tackle the problems and two effective restoration algorithms are proposed for each of them.

1.2.2 Formation of Scalable Color Prints

To produce a hardcopy of a color image, the color image is first decomposed into three-color planes and each plane is halftoned with a binary halftoning algorithm independently.

In order to efficiently render halftone images for various printers and displays which support different resolutions, it is desirable that the output can be scalable in a way that it embeds low resolution halftone images into a full-scale halftone image and, through a very simple procedure such as down-sampling, the low resolution halftone images can be obtained from the high resolution halftone image directly. This so-called multi-resolution halftoning issue was first addressed by Wong in [Wong 96] and [Wong 03]. To achieve this objective, Wong deals with it as a constrained halftoning (CH) problem.

In general, a constrained halftoning problem can be solved as follows. First, a halftone of lower resolution, say *I*, is generated by whatever means. This image then defines the down-sampled pixels of the halftone of higher resolution to be generated. In the generation of the halftone of higher resolution, pixels are processed in a predefined order with error diffusion. When the pixel encountered corresponds to a pixel of the halftone of lower resolution, say $I_{(i,j)}$, after down-sampling, its output value is assigned to be the binary value of $I_{(i,j)}$. Otherwise, it is determined by the thresholding result of the quantizer as usual. In either case, the error is then diffused with a causal filter. For the sake of reference, those pixels whose values are determined by the halftone image of lower resolution instead of the thresholding result of the quantizer during error diffusion are referred to as constrained pixels in this thesis.

This constrained halftoning framework (CH) can work with any conventional error diffusion algorithms which process pixels in a predefined scanning order, producing scalable halftone images of different quality. However, since pixels are processed one by one according to a predefined order and this framework does not take a constrained pixel into account until the pixel is encountered in the course, it is very likely that the value assigned to a constrained pixel is against the natural result of thresholding. This mismatch disturbs the harmony of a local region and degrades the quality of the output.

In [Wong 96], Wong successfully reduces this problem by using an adaptive error diffusion filter. However, pattern artefacts and directional hysteresis still exist due to the causal nature of the error diffusion filter used in his approach. In this thesis,

based on our analysis on the constrainted halftoning, we address this issue and present an approach for producing scalable color prints of higher quality.

1.2.3 Formation of Halftoned Color-quantized Images

Most halftoning algorithms are originally proposed for binary halftoning which emulates a gray level image with a binary image [Ulichney 91, Lau 00, Floyd 76, Jarvis 76, Stucki 81]. To apply them to color quantization, the most straightforward approach is to consider each color component plane as an individual grey scale image and handle them separately [Gentile 90, Zakhor 93, Damera-Venkata 03].

However, this approach may only work in printing applications in which the output colors are composed of 3 or 4 bi-level fixed color components (CMY or CMYK). Color quantization is actually a vector quantization instead of a bi-level uniform scalar quantization as in the case of binary halftoning. It is not a combination of several independent bi-level uniform scalar quantization processes either.

The color reduction process involved in printing applications is only a special case of color quantization in where the three-dimensional color space is separated into three one-dimensional spaces and processed separately. In this special case, the involved palette contains colors which are uniformly distributed over the color space. In practice, the aforementioned straightforward extension of binary halftoning only works when a uniform palette is used in a color quantization process.

In general, when a low-end display unit such as a VGA monitor is involved, the palette colors are not uniformly distributed in the color space and hence the extension of binary halftoning to color halftoning is not as straightforward as most

people assume. Besides, this approach is not an effective approach as it does not take the correlation among color components into account.

When one delivers media information to diverse clients over heterogeneous networks such as the Internet, clients may support different display resolutions and systems may have different caching capabilities. In that case, it is desirable to make media information scalable such that it can be delivered efficiently and reliability. Since color-quantized images are widely used in multimedia applications nowadays, it is desirable to make them scalable such that their downscaled versions can be obtained directly with the images through some simple operations.

In this work, we devote part of our effort in search of a better color quantization algorithm to generate a halftoned color quantized image without directional hysteresis and extend it to produce a scalable output for Internet applications..

1.3 Organization of the Thesis

This thesis is composed of four sections according to the issues addressed. It is organized as follows.

Chapter 2 provides a brief review of relevant works on which the present works is based. It covers various important streams of image restoration techniques, digital halftoning algorithms and color quantization methods.

Chapters 3 and 4 present two proposed solutions for restoring color-quantized images. They are devoted to the case in which no error diffusion is involved in the color quantization process. The one presented in Chapter 3 tackles the problem with regularization theory while the one presented in Chapter 4 tackles the problem with POCS theory.

Chapters 5 and 6 present another two algorithms for restoring color-quantized images. They are devoted to the case in which error diffusion is involved. A POCS-based algorithm is introduced in Chapter 5. Chapter 6 presents an algorithm which is developed based on simulated annealing.

Chapter 7 presents an algorithm for producing scalable color prints. An analysis on the performance of various multiscale error diffusion algorithms on which the presented algorithm is based are also provided in this Chapter.

Chapter 8 presents the framework of multiscale vector error diffusion algorithm for color quantization. The idea presented in Chapter 7 is also extended in this Chapter to work with the proposed framework to generate scalable halftoned color-quantized images.

Finally, the thesis is concluded in Chapter 9 with a summary of the work which was done in this project. Future possible extensions of the present work are also discussed in this final Chapter.

Chapter 2.

A Comprehensive Literature Review

2.1 Introduction

This Chapter provides reviews on some existing works that are relevant to our present works. First, in Section 2.2, the background of image restoration and some common restoration techniques are reviewed. Second, in Section 2.3, the background of digital halftoning and algorithms are reviewed. Third, in Section 2.4, the background of color quantization and some related color quantization methods that appeared in the literature are reviewed.

2.2 Image Restoration

Image restoration is a field of concerning the reconstruction or estimation of an uncorrupted image from a distorted and noisy one. It is known as an ill-posed problem because small data changes can cause large changes in the results. There are two basic tasks. The first one is to estimate the information that is available in the uncorrupted image but not in the data. The second is to present the information retained in the data in the same format as the uncorrupted image. To achieve these two tasks, utilization of *a priori* information is necessary.

In image restoration, *a priori* information are quantitatively formulated and incorporated into restoration. In general, three different areas of *a priori* information are involved. The first one is the information about the degradation. It is used in the modelling of image formation process so that we can quantitatively describe how the observed image is related to the uncorrupted image. In general, the image formation process can be adequately modelled by a linear degradation model as shown in Figure 2.1.

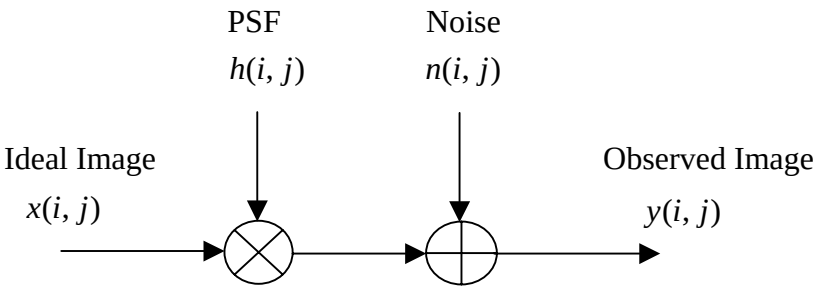


Figure 2.1. Linear degradation model

where $x(i, j)$ denotes the ideal image, $h(i, j)$ represents a deterministic degradation which is also known as the Point Spread Function (PSF) with a finite support say R_k, $n(i, j)$ is the statistical degradation introduced by the imaging system and $y(i, j)$ is the degraded image. In formulation, the model is given as

$$y(i, j) = \sum_{(k,l)\in R_k} h(k, l)x(i-k, j-l) + n(i, j) \tag{2.1}$$

Usually, the linear degradation model in Eq. (2.1) is reordered lexicographically by stacking either the rows or the columns of the image into a vector and is rewritten in terms of a matrix-vector form as

$$\mathbf{y} = \mathbf{Hx} + \mathbf{n} \tag{2.2}$$

where $\mathbf{y}$, $\mathbf{x}$ and $\mathbf{n}$ are the observed image, the ideal image and the noise, respectively. If the original image is of size $N \times N$, then $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{n}$ are represented by vectors of size $N^2 \times 1$, and $\mathbf{H}$ is represented by a $N^2 \times N^2$ matrix [Andrews 77] and [Banham 97].

The second *a priori* information is the characteristics of the ideal image such as smoothness, non-negativity and boundedness. Based on this information, an image model is established and is used to express our *a priori* knowledge about the ideal image. Then it will be applied in the restoration process to convey our expectation on the restoration result.

The third *a priori* information is the statistical information about the noise such as its variance. This information expresses uncertainty about the observed data. It is useful in image restoration when a decision has to be made on the trade-off between the fidelity to the observed data and that to the ideal image.

Since each piece of *a priori* information has it owns distinct function in restoration and imposes different requirement on the solution that may be in conflict with each other. In order to make a good use of them, all available *a priori* information should be collectively applied in a complementary fashion.

In general, by assuming that the deterministic degradation is known *a priori*, three basic tasks are involved in image restoration. The first is the formulation of *a priori* information about the noise and the ideal image. The second is the estimation of noise statistics and the parameters of the image model and the third is the formation of a restoration algorithm which makes the best use of the formulated *a priori* information to counteract the distortion in the observed image.

2.2.1 Image Restoration Techniques

Plenty of techniques have been proposed to solve the image restoration problems in the literature [Hunt 77, Trussell 79, Miller 70, Youla 82, Geman 84, Dines 77, Hunt 73, Sezan 90, Sezan 91] and they span a large variety of mathematical and heuristic concepts. So, it is too ambitious to give a comprehensive review of them. This Section aims to provide a review on some techniques that are related in our present works. Specifically, they are Miller Regularization [Miller 70], Projection onto Convex Sets (POCS) [Youla 82] and Simulated Annealing (SA) [Geman 84, Ingber 93].

2.2.1.1 Miller Regularization

Image restoration using Miller Regularization [Miller 70] is formulated as finding the image that satisfies both of the following constraint sets

$$S_n = \{\hat{\mathbf{x}} : \|\mathbf{y} - \mathbf{H}\hat{\mathbf{x}}\|^2 \le \varepsilon\} \tag{2.3}$$

and

$$S_s = \{\hat{\mathbf{x}} : \|\mathbf{C}\hat{\mathbf{x}}\|^2 \le e\} \tag{2.4}$$

and minimizes objective functional

$$J(\hat{\mathbf{x}}) = \|\mathbf{y} - \mathbf{H}\hat{\mathbf{x}}\|^2 + (\varepsilon / e)\|\mathbf{C}\hat{\mathbf{x}}\|^2 \tag{2.5}$$

The choice of operator **C** in S_s reflects the incorporation of some kind of *a priori* knowledge about the ideal image. In general, the smoothness property of an image is applied by choosing **C** to be the Laplacian operator. Parameters ε and e are assumed to be known *a priori*, and this implies further use of *a priori* knowledge

about the image and the noise. In practice, these parameters can be determined from an estimate of the signal-to-noise ratio.

The minimization of $J(\hat{\mathbf{x}})$ leads to the following solution,

$$[\mathbf{H}^t\mathbf{H} + (\varepsilon / e)\mathbf{C}^t\mathbf{C}]\hat{\mathbf{x}} = \mathbf{H}^t\mathbf{y} \tag{2.6}$$

The commonly used technique to determine $\hat{\mathbf{x}}$ is the iterative algorithm based on the method of successive approximation which is given by

$$\hat{\mathbf{x}}_{k+1} = \hat{\mathbf{x}}_k + \beta[\mathbf{H}^t(\mathbf{y} - \mathbf{H}\hat{\mathbf{x}}_k) - \alpha\mathbf{C}^t\mathbf{C}\hat{\mathbf{x}}_k] \tag{2.7}$$

where $\alpha = \varepsilon / e$, and β is a relaxation parameter of the iteration, which controls the rate of convergence. It can be proved that the iteration will converge to $\hat{\mathbf{x}}$ when $0 < \beta < 2/|\lambda_{\max}|$, where $\lambda_{\max}$ is the largest eigenvalue of $(\mathbf{H}^t\mathbf{H} + \alpha\mathbf{C}^t\mathbf{C})$ [Katsaggelos 89].

2.2.1.2 Projection onto Convex Sets (POCS)

The method of Projection onto Convex Sets (POCS) is a successive projection algorithm which is used extensively in a number of areas such as image coding [Yu 98] and image restoration [Youla 82]. In image restoration, a number of sets are first defined based on *a priori* constraints on the ideal image. Then, any image that satisfies all these constraint sets is considered as a feasible solution or an estimate of the ideal image. In such a method, any *a priori* information concerning the image formation process, the noise, and the ideal image can be incorporated into image restoration, provided that it can be expressed in the form of a closed convex set [Youla 82] called a convex constraint set. There are some useful convex constraints for image restoration [Sezan 90, Sezan 91, Trussell 84]. They are, respectively, that the norm of the residual $\mathbf{y} - \mathbf{Hx}$ is bounded, that the image has a bounded and

positive intensity range, and that the norm of the modelling error for an image model is bounded.

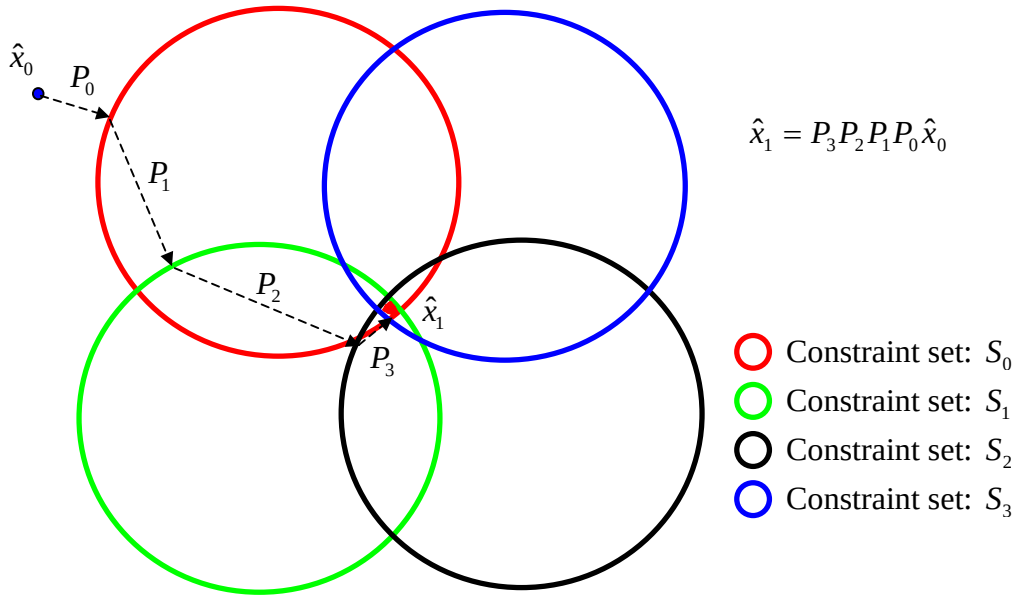


Figure 2.2. Projections among different closed convex constraint sets

Given a set of convex constraint sets for image restoration, it is generally impossible to find an analytical expression for an image that satisfies all the constraint sets. The theory of the projection onto convex sets was developed to find an image in the intersection of a number of closed convex sets [Bregman 65]. As shown in Figure 2.2, a restored image is obtained by performing successive projections onto each of the constraint sets from an initial estimate $\hat{\mathbf{x}}_0$. This process is repeated iteratively until it converges to a feasible solution in the intersection of all the constraint sets [Sezan 82]. Mathematically, the image restoration process can be described as

$$\hat{\mathbf{x}}_k = (P_1 P_2 \cdots P_m)^k \hat{\mathbf{x}}_0 \tag{2.8}$$

where $\hat{\mathbf{x}}_k$ denotes the restored image after k iteration, and P_i represents a non-expansive projector of the i^{th} convex set S_i [Sezan 82, Youla 82]. The POCS algorithm converges to the intersection $\mathbf{S} = \bigcap S_i$, $\forall i$ as long as the intersection of the constraint sets is non-empty. If the intersection is empty, the POCS algorithm will reach a limit cycle. Two basic features of the POCS that make it distinct from other restoration methods are that the restoration solution is affected by the choice of $\hat{\mathbf{x}}_0$, and that the restoration is a non-linear process since the projectors are in general non-linear operators.

2.2.1.3 Simulated Annealing (SA)

Simulated annealing (SA) [Ingber 93] is a general adaptive heuristic iterative technique that exploits an analogy between the way in which a metal cools and freezes into a minimum energy crystalline structure for solving optimization problems and is belonging to the class of non-deterministic algorithms. It has been applied to several combinatorial optimization problems from various fields of science and engineering. One of its typical features is that, besides accepting solutions with improved cost, it also, to a limited extent, accepts solution with deteriorated cost. This gives the heuristic the hill climbing capability. Initially the probability of accepting inferior solutions is large, but as the search progresses, only smaller deteriorations are accepted, and finally only good solutions are accepted. It is both effective and robust. Regardless of the choice of the initial configuration it produces high-quality solutions.

2.3 Digital Halftoning and Its Analysis

Digital halftoning technique converts gray-level image into bilevel image by maintaining the visual appearance as close as possible to the original and has been

widely used in a number of applications such as printing devices and fax machines. It is necessary for displaying gray-level images on device in which direct rendition of gray-level is impossible. It relies on the properties of the human visual system, especially those properties relating to its frequency response. There are a number of digital halftoning methods such as clustered-dot ordered dithering, dispersed-dot ordered dithering and error diffusion. Among them, error diffusion is widely used because it can provide a better quality image than order dithering with a reasonable computational cost.

2.3.1 Analytical Tools

Since different halftoning algorithms would introduce different distribution of quantization error into an image, it is difficult to justify which one is better. In 1988, Ulichney [Ulichney 88] surveyed various halftoning methods and analyzed the frequency content of their outputs. In his work, two spectral statistics were used to analyze a halftone pattern. Both of them rely on estimating the power spectrum through Bartlett's method of averaging periodograms.

The first used spectral statistic is *radially averaged power spectrum density* (RAPSD) $P(f_p)$ [Ulichney 88]. It is defined to be the average power in an annular ring with center radius f_p. In formulation, we have

$$P(f_p) = \frac{1}{N(R(f_p))} \sum_{f \in R(f_p)} \hat{P}(f) \tag{2.9}$$

where $R(f_p)$ is an annular ring of width Δ_p partitioned from the spectral domain, $N(R(f_p))$ is the number of frequency samples in $R(f_p)$ and $\hat{P}(f)$ is the average magnitude square of the Fourier transform coefficient of the pattern.

The second used spectral statistic is *anisotropy* $A(f_p)$ [Ulichney 88]. It is defined as

$$A(f_p) = \frac{1}{N(R(f_p)) - 1} \sum_{f \in R(f_p)} \frac{(\hat{P}(f) - P(f_p))^2}{P^2(f_p)} \tag{2.10}$$

It is the noise-to-signal ratio of the frequency samples of $\hat{P}(f)$ in $R(f_p)$. It was purposed to measure the strength of directional artefact. Ulichney found that halftoning methods that could produce noise with a blue noise characteristic [Ulichney 88] were the ideal methods from a perceptual point of view. He also showed that halftones created by error diffusion [Floyd 76] have such a characteristic. The idea is based on the fact that eyes act as low pass filters. If most of the noisy energy is in the low frequency regions, then the artefacts will be highly visible.

Lau [Lau 01] introduced a spatial domain statistic for the analysis of halftones. It offers a more intuitive understanding of the underlying point process as compared with the conventional spectral domain statistics. In particular, Lau [Lau 01] developed a directional distribution function $D_{r_1,r_2}(\alpha)$ to measure the directional distribution of a pattern and it is defined as

$$D_{r_1,r_2}(\alpha) = \frac{E\{\phi(\Gamma_m^a) \mid y \in \phi\} / N(\Gamma_m^a)}{E\{\phi(\Gamma_m) \mid y \in \phi\} / N(\Gamma_m)} \tag{2.11}$$

It is actually the expected number of points per unit area in a segment Γ_m^a of the ring $\Gamma_m = \{n : r_1 \le \mid n - m \mid < r_2, m \in \phi\}$. In $D_{r_1,r_2}(\alpha)$, inner radius r_1 and outer radius r_2 define a ring region for analysis, and angular parameter α indexes a particular segment in the region. Note that $D_{r_1,r_2}(\alpha) > 1$ and $D_{r_1,r_2}(\alpha) < 1$, respectively, indicate a favoring

and an inhibition of dots in direction α. In ideal case, we have $D_{r_1,r_2}(\alpha)=1$ for all α, which indicates an isotropic distribution in the output.

2.3.2 Digital Halftoning Methods

The aim of this Section is to provide reviews of some digital halftoning methods that are related in our present works. Three state of the art digital halftoning methods are reviewed in details in this Section. They are, respectively, error diffusion [Floyd 76], adaptive error diffusion [Wong 96] and multiscale error diffusion [Peli 91, Katsavounidis 97, Chan 98, Chan 04].

2.3.2.1 Stardand Error Diffusion Algorithm

Error diffusion was introduced by Floyd and Steinberg in 1976 [Floyd 76]. It is an image halftoning method that produces higher quality outputs than order dithering. The algorithm processes the image in a raster scanning fashion in which pixels are scanned from left to right and top to bottom. For each pixel, the algorithm performs thresholding and the quantization error of that pixel is diffused to its neighbours with a casual filter.

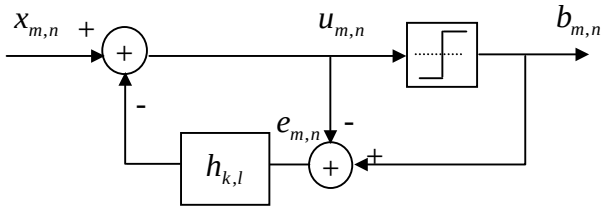


Figure 2.3: The system of error diffusion

Figure 2.3 shows the error diffusion system for digital halftoning. The input pixel $x_{m,n}$ with a pixel value in the range of [0,1], is processed in a raster scanning fashion. As the algorithm proceeds, each input pixel is modified with the weighted

error diffused from the previously processed pixels to produce $u_{m,n}$. Then thresholding is performed to output $b_{m,n}$ and the quantization error $e_{m,n}$ is diffused to neighbouring pixels of $x_{m,n}$ with a casual filter $h_{k,l}$ as shown in Figure 2.3. Mathematically, the process can be formulated as

$$u_{m,n} = x_{m,n} - \sum_{(k,l)\in R} h_{k,l} e_{m-k,n-l} \tag{2.12}$$

$$b_{m,n} = Q(u_{m,n}) = \begin{cases} 1 & if\ u_{m,n} \geq 1/2 \\ 0 & otherwise \end{cases} \tag{2.13}$$

$$e_{m,n} = b_{m,n} - u_{m,n} = Q(u_{m,n}) - u_{m,n} \tag{2.14}$$

where $e_{m,n}$ is the binary quantization error and $u_{m,n}$ is the state variable representing the modified input of the error diffusion system.

Though Floyd and Steinberg error diffusion [Floyd 76] can provide better image quality than order dithering, it suffers from some artefacts such as worm like texture, pattern noise and directional hysteresis. There are some other filters such as Jarvis, Judice and Ninke [Jarvis 76] and Stucki [Stucki 81] proposed to reduce these artefacts. Artefacts can also be reduced if the scanning order is switched from a raster scanning order to a serpentine order [Ulichney 88] or a Peano scanning order [Witten 82].

Kolpatzik and Bouman [Kolpatzik 92] use a weighted mean square error (MSE) criterion

$$\varepsilon = E[(v_{k,l} * (b_{k,l} - x_{k,l}))] \tag{2.15}$$

to consider the optimization of an error diffusion system. The $v_{k,l}$ is an impulse response specified by the characteristics of the human visual system. As a result, ε

can be interpreted as the error between $x_{m,n}$ and $b_{m,n}$ as perceived by the human visual system. Because the exact characteristics of $e_{m,n}$ are not known, Kolpatzik and Bouman make an assumption that $e_{m,n}$ is white noise. The optimum $h_{k,l}$ that minimize ε are then derived based on this assumption and is used for error diffusion.

2.3.2.2 Adaptive Error Diffusion Algorithm

Wong proposed an adaptive error diffusion system [Wong 96] in which a local error criterion was minimized concurrently with the error diffusion process by adjusting the error diffusion kernel. As a result, one does not need to make an assumption on the behaviour of $e_{m,n}$. Furthermore, since the minimization is performed locally along with the error diffusion procedure, the error criterion can be refined so that it reflects more precisely the local characteristics of the images. To minimize the cost subject to the error criterion, the error diffusion kernel is adjusted along the error diffusion process and techniques in adaptive signal processing are applied to perform the error minimization and adjust the kernel.

One of the applications of adaptive error diffusion is to embed lower resolution halftone images into higher resolution halftone images. By doing so, when the larger image is downsampled, a halftone image of lower resolution can be obtained directly.

2.3.2.3 Multiscale Error Diffusion Algorithm

Peli's algorithm [Peli 91] combined the best properties of order dithering and error diffusion for halftoning gray level images. Its basic idea is to quantize the image first, and then refine it by readjusting certain pixels. The pixels are the centers of overlapping square windows over which a weighted average is taken. If switching the

center pixel's value results in a lowering of the average error in the window, then the value is readjusted. 3x3 windows are used initially, then, 5x5, 7x7 and 9x9 are used subsequently.

Kat's algorithm [Katsavounidis 97] is an iterative method that repeatly searches the brightest region of a given image via "maximum intensity guidance" to assign dots and diffuses the quantization error at each iteration until the total energy of the processing output is equivalent to that of the given image. This method is superior to conventional error diffusion methods in a way that no sequential predetermined order is required for error diffusion. Accordingly, non-casual filters can be used in diffusing quantization error to avoid directional hysteresis.

Chan's algorithms [Chan 98, Chan 04] are modified versions of Kat's algorithm [Katsavounidis 97]. Instead of iteratively dividing an input image into four non-overlapping segments, Chan [Chan 98] divides it into nine overlapping regions. By doing so, boundary effect becomes less prominent. Besides, it was found by Chan [Chan 98] that error leakage occurred during error diffusion process in [Katsavounidis 97] and it affected the quality of the halftoned image. A solution for solving this problem was proposed and better performance was provided.

The modification made in [Chan 04] puts its focus on feature preserving. Instead of using maximum error intensity guidance to locate a pixel position for a new dot, it uses extreme error intensity guidance to avoid the bias caused by only assigning white dots. Conceptually, minority dots in a local region provide features in the region and should be taken care of it no matter whether they are white dots or black dots. A simple trick is also introduced to reduce the boundary effect.

2.4 Color Quantization

Color quantization is an image processing technique which uses a limited set of colors to represent a true color image with as little degradation as possible. Two steps are involved in color quantization. The first step is to design a color palette which selects the best possible set of colors for a particular image and is normally based on the content of the image being quantized. The second step is pixel mapping in which each color pixel of the original color image is quantized with a color from the palette to yield the output image.

A color palette can be a system palette or a custom palette. A system palette is provided by a display system for all images to be displayed. By sharing a system palette, different images can be simultaneously displayed with compromised quality. A custom palette is an optimized palette designed individually for a color image. With a custom palette, the quality of a quantized image can be approximated as close as possible. The problem is that most of the displaying hardware devices can only support one color palette at a time. It is also possible that a custom palette can not be supported by the display unit. In short, the generation of a color palette is application-dependent and is subject to both quality criterion and practical constraints.

Pixel mapping can be done in one of the two ways with a color palette. The first way is to simply replace the input image by mapping the color of each pixel directly to a representative color that is stored in the palette. However, certain types of degradation can be introduced due to the limited colors used to produce the output image. There are two most disturbing defects. One is the false contour which appears in smoothly changing regions and the other is the color shift in a color-quantized image. The smaller the color palette size, the more severe the defects are. For the sake

of reference, a color image quantized in this way is referred as a *color-quantized image*.

The second way of pixel mapping is to make use of the spatial integration technique to create the illusion of varying shades of colors where none actually exists. Small areas of colors in the original image are manipulated first before they are quantized. It exploits the lower sensitivity of eyes to spatial resolution and exchanges higher color resolution with lower spatial resolution. The eye averages the colors in a neighborhood of the point of interest and creates the illusion of more colors. This spatial integration technique is commonly termed to halftoning. For the sake of reference, we will refer the images produced with this technique as *halftoned color-quantized images*.

When a custom palette is used, [Özdemir 01, Akarun 96, Scheunders 98, Puzicha 00, Akarun 97, Özdemir 00, Breaux 99] it is possible to jointly optimizes the palette generation and the halftoning processes so as to optimize a halftoned color-quantized image [Özdemir 01, Akarun 96, Scheunders 98, Puzicha 00]. For example, Özdemir‘s algorithms [Özdemir 01] take the quantization error into account in training the palette for color quantization by making use of the fuzzy quantization technique. In [Scheunders 98], Scheunders proposed a competitive learning scheme to embed a dithering process in the color quantization process. In [Puzicha 00], Puzicha proposed a new model to simultaneously quantize and halftone color images. It is based on a rigorous cost-function that optimizes a quality criterion derived from a simplified model of human perception. These algorithms can provide good halftoned color-quantized images. However, since dedicated palettes for corresponding algorithms have to be used, the applications of these algorithms are restricted to a certain extent. Not any display device can support a dedicated palette. Besides, the

palette-training processes involved in these algorithms are usually very time-consuming.

It would be more flexible if an algorithm can handle any arbitrary given palettes without restriction [Akarun 97, Özdemir 00, Breaux 99]. The approach presented by Orchard [Orchard 91] forms a common framework that most of these algorithms adopt. Both Akarun's algorithm [Akarun 97] and Özdemir algorithm [Özdemir 00] adopt this framework. Akarun's algorithm [Akarun 97] uses an adaptive error diffusion filter to prevent textural contours, color impulses and color shifts. Instead of the conventional Euclidean distance criterion, Özdemir's algorithm [Özdemir 00] uses a weighted sum of the distances among color vectors as a searching criterion in color quantization process to prevent excess accumulation of quantization errors.

2.4.1 Color Quantization Methods

The aim of this Section is to provide a review of some color quantization algorithms that are related in our present works. In particular, we will address Color Quantization of Images [Orchard 91], Adaptive Methods for Dithering Color Images [Akarun 97] and Fuzzy Error Diffusion [Özdemir 00] in this Chapter.

2.4.1.1 Color Quantization of Images

Color quantization can be carried out with or without error diffusion. They are, respectively, introduced as follows.

2.4.1.1.1 Color Quantization Without Error Diffusion

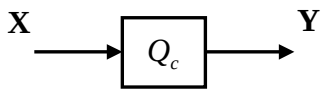


Figure 2.4. Color quantization without error diffusion

A pixel of a full color image $\mathbf{X}$ generally consists of three color components. The intensity values of these three components form a vector in a 3D space. In color quantization, each pixel vector is compared with a set of representative color vectors $\hat{\mathbf{v}}_i$, *for* $i = 1,2,\cdots,N_c$. There representive color vectors are stored in a previously generated *color palette*. The best-matching representive color is selected based on the minimum Euclidean distance criterion, where the Euclidean distance measure between vectors $\vec{\mathbf{v}}_1$ and $\vec{\mathbf{v}}_2$ is defined as $d(\vec{\mathbf{v}}_1, \vec{\mathbf{v}}_2) \equiv \left\| \vec{\mathbf{v}}_1 - \vec{\mathbf{v}}_2 \right\|$. In other words, a pixel vector $\vec{\mathbf{v}}$ is represented by color $\hat{\mathbf{v}}_k$ if and only if $d(\vec{\mathbf{v}}, \hat{\mathbf{v}}_k) \le d(\vec{\mathbf{v}}, \hat{\mathbf{v}}_j)$ for all $j = 1,2,\cdots,N_c$. Once the best-matching colors for all pixel vectors of the source image have been selected from the color palette, the indices of the selected colors are recorded or transmitted to the receiver with the color palette. With the same color palette, the color-quantized image can be reconstructed based on the stored or received indices.

2.4.1.1.2 Color Quantization With Error Diffusion

A 24-bit full-color image $\mathbf{X}$ generally consists of three 8-bit color planes, say, $\mathbf{X}_r$, $\mathbf{X}_g$ and $\mathbf{X}_b$, which represents the red, the green and the blue color planes of the image respectively. A pixel is then a vector represented as $\vec{\mathbf{X}}_{(i,j)} = (\mathbf{X}_{(i,j)r}, \mathbf{X}_{(i,j)g}, \mathbf{X}_{(i,j)b})$, where $\mathbf{X}_{(i,j)c} \in [0,1]$ is the intensity value of the c^{th} color component of the $(i,j)^{th}$

pixel. Here, we assume that the image is of size $N \times N$ and the maximum and the minimum intensity values of a pixel are, respectively, 1 and 0 .

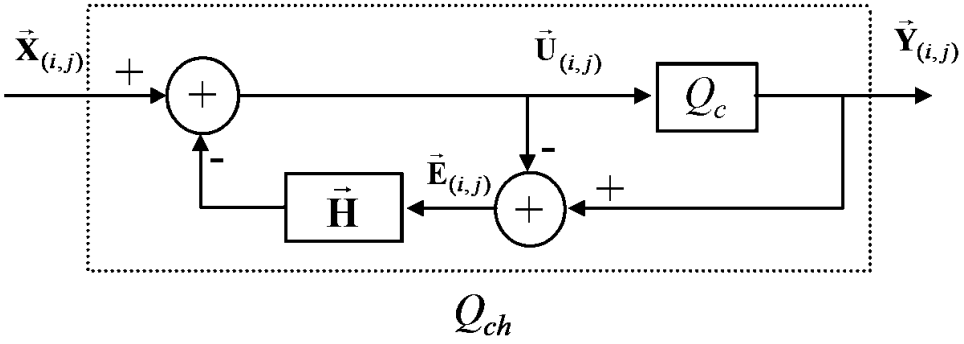


Figure 2.5. Color quantization with error diffusion

Figure 2.5 shows the system which performs color quantization with error diffusion proposed by Orchard [Orchard 91]. The input image $\mathbf{X}$ is scanned in a row-by-row fashion and processed as follows to produce the encoded image $\mathbf{Y}$.

$$\mathbf{U}_{(i,j)c} = \mathbf{X}_{(i,j)c} - \sum_{(k,l)\in S} \mathbf{H}_{(k,l)c} \mathbf{E}_{(i-k,j-l)c} \tag{2.16}$$

$$\vec{\mathbf{Y}}_{(i,j)} = Q_c[\vec{\mathbf{U}}_{(i,j)}] \tag{2.17}$$

and

$$\vec{\mathbf{E}}_{(i,j)} = \vec{\mathbf{Y}}_{(i,j)} - \vec{\mathbf{U}}_{(i,j)} = Q_c[\vec{\mathbf{U}}_{(i,j)}] - \vec{\mathbf{U}}_{(i,j)} \tag{2.18}$$

where $\vec{\mathbf{U}}_{(i,j)}$ is a state vector of the system, $\vec{\mathbf{E}}_{(i,j)}$ is the quantization error of the pixel at position (i, j) and $\mathbf{H}_{(k,l)c}$ is a coefficient of the error diffusion filter for the c^{th} color component. S is the casual support region of $\mathbf{H}_{(k,l)c}$. The operator $Q_c[\bullet]$ performs a 3D vector quantization. Specifically, the 3D vector $\vec{\mathbf{U}}_{(i,j)}$ is compared with a set of representative color vectors stored in a previously generated color palette $C = \{ \hat{\mathbf{v}}_i : i=1,2, \ldots N_c \}$. The best-matched vector in the palette is selected based on the minimum Euclidean distance criterion. In other words, a state vector $\vec{\mathbf{U}}_{(i,j)}$ is

represented by color $\hat{\mathbf{v}}_k$ if and only if $\| \vec{\mathbf{U}}_{(i,j)} - \hat{\mathbf{v}}_k \| \leq \| \vec{\mathbf{U}}_{(i,j)} - \hat{\mathbf{v}}_l \|$ for all $l = 1,2 \ldots N_c$. Once the best-matched vector is selected from the color palette, its index is recorded and the quantization error $\vec{\mathbf{E}}_{(i,j)} = \hat{\mathbf{v}}_k - \vec{\mathbf{U}}_{(i,j)}$ is diffused to pixel (i, j) 's neighborhood with eqn. (2.16). To handle the boundary pixels, $\vec{\mathbf{E}}_{(i,j)}$ is defined to be zero when (i, j) falls outside the image. After the scanning is finished, the recorded indices can be used in the future to reconstruct the color-quantized image with the same color palette.

2.4.1.2 Adaptive Method for Dithering Color Images

Akarun's algorithm [Akarun 97] is an adaptive method for dithering color images. It is developed to improve the performance of Orchard's algorithm [Orchard 91]. The coefficients of the error diffusion filter are updated by a normalized LMS - type algorithm to prevent textural contours, color impulse and color shift. The error diffusion filter matrix $\vec{\mathbf{H}}$ is obtained by minimizing the mean square error (MSE) between the average color value of the original image and the dithered image. The optimum error diffusion filter coefficients can be obtained by minimizing the following functional

$$\mathbf{E}\{\left\| \vec{\mathbf{X}}_{(i,j)} - Q_c(\vec{\mathbf{U}}_{(i,j)}) \right\|^2\} \tag{2.19}$$

where $\vec{\mathbf{X}}_{(i,j)}$ is the input pixel, $\vec{\mathbf{U}}_{(i,j)}$ is the state vector in the system and Q_c is the quantization process.

2.4.1.3 Fuzzy Error Diffusion

Özdemir's algorithm [Özdemir 00] is a fuzzy error diffusion algorithm proposed for color quantization with error diffusion. It makes use of a membership function for pixels. The amount of error to be diffused is determined by considering the relative location of the pixel not only to the closest codebook vector, but also to all other palette entries. Its objective is to hide the quantization errors by error diffusion, while preventing the excess accumulation of errors. It is achieved through an attraction repulsion scheme according to a fuzzy membership function.

Chapter 3

A Constrained Least Square Restoration Algorithm for Restoring Color-quantized Images

3.1 Introduction

Chapters 3 and 4 address the restoration of color-quantized images. As mentioned in Chapter 1, Section 1.2.1, restoration of color-quantized image is rarely addressed in literature and direct applications of existing restoration techniques are generally inadequate to deal with the problem. These two Chapters are devoted to formulating the problem of color-quantized image restoration and developing dedicated restoration algorithms for the restoration of color-quantized images. In particular, the restoration algorithm presented in this Chapter is based on the regularization theory [Miller 70].

This Chapter is organized as follows. In Section 3.2, we formulate the *a priori* information about the degradation process of color quantization and about the original image. Then, in Section 3.3, we formulate the constraints for restoration. In Section 3.4, we present the derivation of a restoration algorithm for restoring color-quantized

images. In Section 3.5, simulation results and comparative study are provided to evaluate the performance of the proposed algorithm. Finally, a summary is given in Section 3.6.

3.2 Formulation of *A priori* Information

This section formulates the *a priori* information of a system of color quantization in which no error diffusion is involved with the model presented in Section 2.4.1.1.1.

Consider the case that image $\mathbf{X}$ is encoded as $\mathbf{Y}$ by color quantization with a color palette C containing N_c colors:

$$C = \{\hat{\mathbf{v}}_i \mid i = 1,2,\cdots, N_c\} \tag{3.1}$$

According to color quantization theory, the N_c colors in the palette partition the whole color vector space, say Γ, into N_c non-overlapped Voronoi regions.

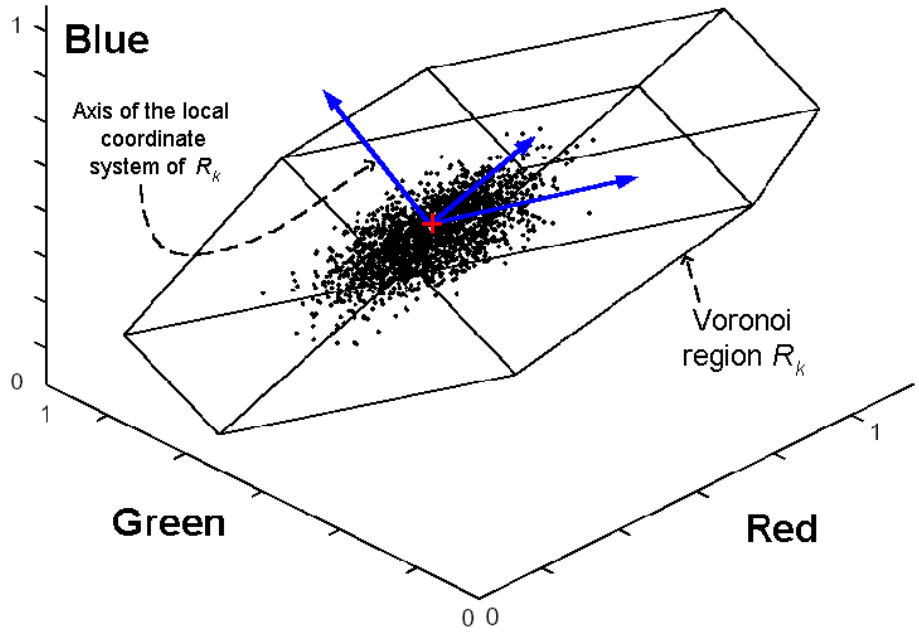


Figure 3.1. The local coordinate system associated with a Voronoi region R_k.

Without loss of generality, let $R_k = \{\vec{\mathbf{v}} \mid \vec{\mathbf{v}} \in \Gamma \text{ and } d(\vec{\mathbf{v}}, \hat{\mathbf{v}}_k) \le d(\vec{\mathbf{v}}, \hat{\mathbf{v}}_j) \text{ for } \forall \hat{\mathbf{v}}_j \in C\}$, where $\hat{\mathbf{v}}_k \in C$, be a particular Voronoi region in the R-G-B space as shown in Figure 3.1. Here, we assume that the maximum and the minimum intensity values of each color component of a pixel are, respectively, 1 and 0. In this Figure, the surface mounted to the framework shows the border of the Voronoi region and the dots in the region represent the pixels of the image $\mathbf{X}$ which belong to R_k. All these pixels form a set of vectors denoted as $R_{k(X)}$.

Through a principal component analysis [Jolliffe 86], the three principal components of the elements in $\{\vec{\mathbf{v}} - \hat{\mathbf{v}}_k \mid \vec{\mathbf{v}} \in R_{k(X)}\}$ can be obtained. In the analysis, we assume that the mean of $\vec{\mathbf{v}} \in R_{k(X)}$ is $\hat{\mathbf{v}}_k$. In vector quantization, the codeword associated with a Voronoi region is selected to be the mean of the training vectors in the Voronoi region such that it can be the best representative of the training vectors in MSE sense [Gersho 90]. Since training vectors are considered to be typical examples of the input vectors to be encountered, this assumption is plausible. The three arrows marked in Figure 3.1 show the directions of the three principal components in such a case and their intersection point corresponds to $\hat{\mathbf{v}}_k$. Accordingly, a local coordinate system of R_k can be defined by using them as the three axes.

Voronoi region R_k is then moved to align this local coordinate system to the R-G-B coordinate system with a linear transformation M_k. Specifically, it is realized by

$$M_k(\vec{\mathbf{v}}) = T_k(\vec{\mathbf{v}} - \hat{\mathbf{v}}_k) \quad \textit{for } \vec{\mathbf{v}} \in R_k \tag{3.2}$$

where T_k is a 3×3 matrix whose rows are the corresponding eigenvectors of matrix A_k whose element $A_k(i, j)$ is defined as

$$A_k(i, j) = \frac{1}{N_{o,k}} \sum_{\vec{\mathbf{v}} \in R_{k(X)}} (v_i - \hat{v}_{k,i})(v_j - \hat{v}_{k,j}) \quad for\ \ i,j \in \{1, 2, 3\}. \tag{3.3}$$

Here, $N_{o,k}$ is the total number of vectors in $R_{k(X)}$, v_i and v_j are, respectively, the i^{th} and j^{th} color components of $\vec{\mathbf{v}}$, and, $\hat{v}_{k,i}$ and $\hat{v}_{k,j}$ are, respectively, the i^{th} and j^{th} color components of $\hat{\mathbf{v}}_k$ where $i, j \in \{1,2,3\}$. The first, the second and the third color components of a pixel correspond to its red, green and blue components, respectively. Note that T_k is actually the KL transform kernel for the elements in $\{\vec{\mathbf{v}} - \vec{\mathbf{v}}_k \mid \vec{\mathbf{v}} \in R_{k(X)}\}$. The inverse of transformation T_k is given as T_k^t.

After the transformation, the first most, the second most and the least significant principal components align, respectively, with the R-, G- and B-axis. Then, for all $\vec{\mathbf{v}} \in R_{k(X)}$, a corresponding variance vector can be defined as

$$\vec{\sigma}_k = (\sigma_{k,1}^2, \sigma_{k,2}^2, \sigma_{k,3}^2) \tag{3.4}$$

where

$$\sigma_{k,s}^2 = \frac{1}{N_{o,k}} \sum_{\vec{\mathbf{v}} \in R_{k(X)}} \left(\left[T_k(\vec{\mathbf{v}} - \hat{\mathbf{v}}_k) \right]_s \right)^2 \quad for\ s \in \{1,2,3\}. \tag{3.5}$$

Here, $\left[T_k(\vec{\mathbf{v}} - \hat{\mathbf{v}}_k)\right]_s$ denotes the s^{th} most significant principal component of $T_k(\vec{\mathbf{v}} - \hat{\mathbf{v}}_k)$. Note $\sigma_{k,s}^2$ is actually the variance of the s^{th} most significant principal component of $\{\vec{\mathbf{v}} - \hat{\mathbf{v}}_k \mid \vec{\mathbf{v}} \in R_{k(X)}\}$, which provides us an important constraint to seek the original image $\mathbf{X}$.

In practice, $\mathbf{X}$ is not available and hence $R_{k(X)}$ has to be estimated. Different approaches of estimation provide different results. The most straightforward approach is to select some typical images to form a training set Ω_t and then approximate $\vec{\sigma}_k$ with vectors in $\Omega_t \cap R_k$ instead of $R_{k(X)}$. In our approach, we blur the available $\mathbf{Y}$, the color-quantized image on hand, with a low pass filter and then estimate $R_{k(X)}$ to be $R'_{k(X)} = \{\vec{\mathbf{v}} \mid \vec{\mathbf{v}} \in R_k$ and $\vec{\mathbf{v}}$ is a pixel in the blurred $\mathbf{Y}\}$.

By doing so, the training pixel vectors are biased to the blurred $\mathbf{Y}$ and, to a certain extent, $\mathbf{X}$ instead of the training images, which may be very different from $\mathbf{X}$. Besides, the correlation among adjacent pixels in a particular local region of $\mathbf{Y}$ can be taken into account to estimate a particular pixel of $\mathbf{X}$ in a corresponding position. After obtaining $R'_{k(X)}$, all $R_{k(X)}$-dependent parameters such as $A_{k(i,j)}$, T_k and $\vec{\sigma}_k$ can be approximated with vectors in $R'_{k(X)}$ instead of $R_{k(X)}$.

Note that all the $\vec{\sigma}_k$ s are solely determined by the available image $\mathbf{Y}$ and the color palette which is attached with the image or known by default. No extra information is required for their estimation.

3.3 Formulation of Constraints for Restoration

Let $\vec{\mathbf{x}}_m = (x_{m,r}, x_{m,g}, x_{m,b})^t$ and $\vec{\mathbf{y}}_m = (y_{m,r}, y_{m,g}, y_{m,b})^t$ be the 3-dimensional vectors representing the m^{th} pixels of the original image $\mathbf{X}$ and the encoded image $\mathbf{Y}$, respectively. If $\hat{\mathbf{v}}_k \in C$ is the color used to represent $\vec{\mathbf{x}}_m$, then we will have $\vec{\mathbf{y}}_m = \hat{\mathbf{v}}_k$, and the distance between $\vec{\mathbf{y}}_m$ and $\vec{\mathbf{x}}_m$ will be bounded by the boundary of the Voronoi region R_k. More specifically, any particular principal component of $\vec{\mathbf{x}}_m - \vec{\mathbf{y}}_m$, say, $[T_k(\vec{\mathbf{x}}_m - \vec{\mathbf{y}}_m)]_s$, is also bounded. In formulation, we have

$$([T_k(\vec{\mathbf{x}}_m - \vec{\mathbf{y}}_m)]_s)^2 \le \varepsilon_{m,s} \tag{3.6}$$

where $\varepsilon_{m,s}$ is the corresponding bound.

Since we have $\vec{\mathbf{y}}_m = \hat{\mathbf{v}}_k$, the bound $\varepsilon_{m,s}$ can be estimated to be a function of $\sigma_{k,s}^2$. By assuming that $\varepsilon_{m,s}$ is proportional to $\sigma_{k,s}^2$, we have $\varepsilon_{m,s} = K\sigma_{k,s}^2$, where K is a constant. Note that this assumption can easily be satisfied as long as K is sufficiently large. In practice, by assuming a normal distribution, $K \approx 10$ is a reasonable estimate as the range of $[-3\sigma_{k,s}, 3\sigma_{k,s}]$ covers more than 99% of the possible values of $[T_k(\vec{\mathbf{x}}_m - \vec{\mathbf{y}}_m)]_s$.

The information carried by $\sigma_{k,s}^2$ provides us an important constraint to seek the original image $\mathbf{X}$. To formulate this constraint, we first rewrite equation (3.6) as

$$\omega_{m,s}^2([T_k(\vec{\mathbf{x}}_m - \vec{\mathbf{y}}_m)]_s)^2 \le K \tag{3.7}$$

where $\omega_{m,s}^2 = 1/\sigma_{k,s}^2$. The overall distortion in $\mathbf{Y}$ can then be described as

$$\sum_{m=1}^{N}\sum_{s=1}^{3}\omega_{m,s}^2([\mathbf{T}_m(\vec{\mathbf{x}}_m - \vec{\mathbf{y}}_m)]_s)^2 \le 3NK \equiv \varepsilon_d \tag{3.8}$$

where N is the total number of pixels of the original image, and $\mathbf{T}_m$ is the KLT kernel associated with the codeword used to represent $\vec{\mathbf{x}}_m$.

In matrix-vector formulation, we have

$$J_d \equiv \left\|\mathbf{AT}(\vec{\mathbf{y}} - \vec{\mathbf{x}})\right\|^2 \le \varepsilon_d \tag{3.9}$$

where $\vec{\mathbf{y}} = (y_{1,r}, y_{1,g}, y_{1,b}, y_{2,r}, \cdots, y_{N,b})^t$, $\vec{\mathbf{x}} = (x_{1,r}, x_{1,g}, x_{1,b}, x_{2,r}, \cdots, x_{N,b})^t$, $\mathbf{A}$ is a $3N \times 3N$ diagonal matrix whose diagonal elements are $\{\omega_{1,1}, \omega_{1,2}, \omega_{1,3}, \omega_{2,1}, \cdots, \omega_{N,1}\}$ and

$$\mathbf{T} = \begin{bmatrix} \mathbf{T}_1 & & & 0 \\ & \mathbf{T}_2 & & \\ & & \ddots & \\ 0 & & & \mathbf{T}_N \end{bmatrix}. \tag{3.10}$$

Typical images would generally have weak high frequency components as the intensity of neighboring pixels is highly correlated. This feature can be exploited as an additional *a priori* information in the restoration of color-quantized images. To incorporate this information into restoration, we assume that the total energy of the high frequency components of a particular image color plane, say $\mathbf{X}_c$, is bounded. In formulation, the smoothness constraint for the restored image can be described by

$$J_s \equiv \sum_{c=1}^{3} \left\| \mathbf{H}\mathbf{M}_c \vec{\mathbf{x}} \right\|^2 / \varepsilon_{s_c} \le 3 \tag{3.11}$$

where ε_{s_c} is the upper energy bound of the high frequency components of $\mathbf{X}_c$, $\mathbf{H}$ is a linear highpass operator, which is generally a spatial 2D Laplacian filter, and $\mathbf{M}_c$ is a $N \times 3N$ matrix whose element is defined as

$$\mathbf{M}_c(i,j) = \begin{cases} 1 & \text{if} \quad j = 3i - 3 + c \\ 0 & \text{else} \end{cases} \qquad \text{for} \;\; i = \{1, \cdots, N\} \;\; \text{and} \;\; c = 1, 2, 3 \tag{3.12}$$

Note $\mathbf{M}_1\vec{\mathbf{x}}$, $\mathbf{M}_2\vec{\mathbf{x}}$ and $\mathbf{M}_3\vec{\mathbf{x}}$ are, respectively, the lexicographically ordered red, green and blue color planes of $\mathbf{X}$.

3.4 Formulation of Restoration Algorithm

Based on the *a priori* information concerning the color quantization process and the original image, two constraint sets have been defined for the restoration of color-quantized images. By assuming that constraints (3.9) and (3.11) are equally

important, the restored image can then be obtained by minimizing the objective function

$$J(\vec{\mathbf{x}}') = \left\|\mathbf{AT}(\vec{\mathbf{y}} - \vec{\mathbf{x}}')\right\|^2 + \sum_{c=1}^{3} \alpha_c \left\|\mathbf{HM}_c \vec{\mathbf{x}}'\right\|^2 \tag{3.13}$$

where $\vec{\mathbf{x}}'$ is the estimate of $\vec{\mathbf{x}}$ and $\alpha_c = \varepsilon_d / 3\varepsilon_{s_c}$ [Tikhonov 77]. The minimization of $J(\vec{\mathbf{x}}')$ with respect to $\vec{\mathbf{x}}'$ leads to the normal equation

$$\left(\mathbf{T}^t\mathbf{A}^t\mathbf{AT} + \sum_{c=1}^{3} \alpha_c \mathbf{M}_c^t \mathbf{H}^t \mathbf{HM}_c\right) \vec{\mathbf{x}}' = \mathbf{T}^t\mathbf{A}^t\mathbf{AT}\vec{\mathbf{y}} \tag{3.14}$$

In practice, an iterative technique is applied to successively approximate the solution. Different iterative algorithms can be formulated with different techniques. When the steepest descent method [Gilyazov 00] is used, it results in the following iteration for the restoration solution:

$$\vec{\mathbf{x}}'_{(k+1)} = \vec{\mathbf{x}}'_{(k)} + \beta\left\{\mathbf{T}^t\mathbf{A}^t\mathbf{AT}(\vec{\mathbf{y}} - \vec{\mathbf{x}}'_{(k)}) - \sum_{c=1}^{3} \alpha_c \mathbf{M}_c^t \mathbf{H}^t \mathbf{HM}_c \vec{\mathbf{x}}'_{(k)}\right\}, \tag{3.15}$$

where k is the iteration index ($k = 0, 1, 2, \ldots$) and β is the relaxation parameter. A sufficient condition for the convergence of this iteration is

$$0 < \beta < 2 / \lambda_{\max} \tag{3.16}$$

where $\lambda_{\max}$ is the largest eigenvalue of matrix $\left(\mathbf{T}^t\mathbf{A}^t\mathbf{AT} + \sum_{c=1}^{3} \alpha_c \mathbf{M}_c^t \mathbf{H}^t \mathbf{HM}_c\right)$ [Schafer 81].

In order to make sure that the color-quantized output of the estimate is $\vec{\mathbf{y}}$, iteration (3.15) is modified to be

$$\vec{\mathbf{x}}'_{(k+1)} = P\left[\vec{\mathbf{x}}'_{(k)} + \beta\left\{\mathbf{T}^t\mathbf{A}^t\mathbf{AT}(\vec{\mathbf{y}} - \vec{\mathbf{x}}'_{(k)}) - \sum_{c=1}^{3}\alpha_c\mathbf{M}_c^t\mathbf{H}^t\mathbf{H}\mathbf{M}_c\vec{\mathbf{x}}'_{(k)}\right\}\right], \quad (3.17)$$

where $P[\bullet]$ is a projection defined as

$$P:\vec{\mathbf{x}}'_m = \begin{cases} \vec{\mathbf{x}}'_m & if \quad Q[\vec{\mathbf{x}}'_m] = \vec{\mathbf{y}}_m \\ \vec{\mathbf{y}}_m & else \end{cases} \qquad \forall m \quad (3.18)$$

where $\vec{\mathbf{x}}'_m$ is the current estimate of $\vec{\mathbf{x}}_m$ and $Q[\bullet]$ is the color quantization operator. Since $P[\bullet]$ is a projection onto a closed convex set, the convergence of iteration (3.17) can also be guaranteed as long as (3.16) is satisfied [Biemond 90]. It is common to set the initial estimate $\vec{\mathbf{x}}'_{(0)}$ to be $\vec{\mathbf{y}}$.

Parameters α_c should be determined prior to performing the iteration. Recall that parameter α_c is defined as $(NK)/\varepsilon_{s_c}$, where ε_{s_c} is the bound of the smoothness constraint set defined in (3.11), N is the image size, and K is a constant to make (3.6) hold. In practice, since image $\mathbf{X}$ is unavailable, we set ε_{s_c} to be $K'\left\|\mathbf{H}\mathbf{M}_c\vec{\mathbf{y}}\right\|^2$, where K' is a tuning parameter for image smoothness. As for K, as we have mentioned previously, $K = 10$ is a reasonable estimate.

We note here that the iteration given in eqn.(3.15) is derived with the steepest descent method [Gilyazov 00]. The convergence speed of this method is slow. To improve the efficiency, one can make use of some conjugate direction methods such as the conjugate gradient method [Gilyazov 00] to derive another iteration algorithm. In this Chapter, only the approach using the steepest descent method is presented as the focus of the Chapter is on the restoration performance.

In this Chapter, the proposed algorithm is formulated in R-G-B space. This is based on the observation that a palette for a video color display is usually defined in

R-G-B space and the color space associated with a video color display is generally R-G-B. The algorithm can also be formulated in other color spaces such as Y-I-Q space with a similar approach presented in the Chapter.

3.5 Performance Evaluation

In this Section, some performance evaluation of the proposed algorithm is provided. Firstly, in Section 3.5.1, the details of the realization of the proposed algorithm in our evaluation are described. Secondly, in Section 3.5.2, some other restoration algorithms used for comparative studies are introduced. Finally, in Section 3.5.3, evaluation results are provided and analyzed.

3.5.1 Realization Details of the Proposed Algorithm

Simulation was carried out to evaluate the performance of the proposed algorithm on a set of color-quantized images. In our simulation, a number of 24-bit full color test images were first color-quantized with a color palette of 256 colors. In our study, three color palettes were generated with median cut algorithm [Heckbert 82] and octree algorithm [Gervautz 90] respectively. They were used to investigate if the proposed algorithm worked with inputs obtained with different color palettes. The test images applied were a set of *de facto* standard 24-bit full color images of size 256×256 each.

The proposed restoration algorithm was applied to restore the color-quantized images. In the realization of the proposed algorithm, the observed color image $\mathbf{Y}$ was blurred with a lowpass filter to generate a training set to estimate T_k and $\vec{\sigma}_k$. Both Gaussian and average filters of size 3x3 and 5x5 were tried in our simulations to study the influence of filter type and filter support to the restoration performance. It was

found that their performance were more or less the same. In this Chapter, only the results obtained with a 3x3 average filter is presented. K' was assigned to be 0.5 and $\mathbf{H}$ was a 3x3 Laplacian filter defined as [0,-1,0;-1,4,-1;0,-1,0]. Iterative terminated when $\left\|\vec{\mathbf{x}}'_{(k+1)} - \vec{\mathbf{x}}'_{(k)}\right\|^2 \Big/ \left\|\vec{\mathbf{x}}'_{(k)}\right\|^2 < 10^{-6}$ was satisfied.

By considering that the extreme case happens when the whole color space is uniformly partitioned into slices of equal width and that colors are uniformly distributed in the color space, the lower bound of $\sigma_{k,s}^2$ is set to be $\frac{1}{12}\left(\frac{1}{N_c}\right)^2$, where N_c is the total number of colors in the palette.

3.5.2 Algorithms for Comparative Study

Some other restoration algorithms which were originally proposed for restoring noisy and blurred color images were also evaluated for comparison. They were simulated here for comparative study as few schemes had been proposed for restoring color-quantized images and they are typical examples of the type [Altunbasak 01, Angelopoulos 94, Barni 00, Galatsanos 91a, Galatsanos 89, Galatsanos 91b, Hunt 84, Kaulgud 99]. In particular, Galatsanos's algorithm [Galatsanos 91a] is based on the constrained least square approach and Hunt's algorithm [Hunt 84] is based on Wiener filtering. Two algorithms were presented in Altunbasak's work [Altunbasak 01]. One makes use of the correlation among the color components of a pixel while the other one does not during the restoration. They are, respectively, referred to as KL and IND in [Altunbasak 01]. Both algorithms take the colorimetric aspects into account and try to minimize the error in CIELAB space [CIE 78]. Kim's Algorithm [Kim 01] was designed for restoring vector quantized

image and it was realized for comparison here due to that the problem nature of vector quantization is similar to color quantization.

In realizing Galatsanos's algorithm [Galatsanos 91a], the noise power of each channel was estimated with the original full-color image. In realizing Hunt's algorithm [Hunt 84], three separate Wiener filters were used in three different channels and, during the design of the filters, the noise spectrum of each channel was estimated with the original full-color image. Similarly, the original full-color image was used to estimate the power spectra of different channels in realizing Altunbasak's algorithms [Altunbasak 01]. In a practical situation, no original image is available and hence all parameters must be estimated from the degraded image. In other words, in practice, the restoration results of [Galatsanos 91a, Hunt 84, Altunbasak 01] may not be as good as those presented in this Chapter. In realising Kim's algorithm, [Kim 01], the blurred version of the observed image was used as the training set and the projection ratio η was set to be 0.05. As it is not necessary to use the original full-color image to extract information for the realization of Kim's algorithm [Kim 01] and the proposed algorithm, additional credit should be added to the simulation results of these algorithms indeed.

There could be some other approaches to restore a color-quantized image. For instance, one can use some training images to pre-train a linear prediction filter and then use the trained filter to restore a color-quantized image later on. Chang's [Chang 01] and Mese's [Mese 01] algorithms are typical examples of this approach though they are actually algorithms for restoring binary halftoned images instead of color-quantized images. Specifically, the former trains the filter with a LMS adaptive filtering algorithm while the latter does it with a best linear estimator. These algorithms were also evaluated in our study for comparison. However, they were

modified to handle color-quantized images. Since these algorithms assumes bilevel halftones as input, the number of input patterns appeared in the filter support is expected to be very limited and hence they try to construct a lookup table with the training images to reduce the computation effort with a table lookup technique in future restoration. This is not practical when the input is a color-quantized image. Accordingly, we eliminated this part in our simulations.

Standard images *Fruits* and *Peppers*, and their corresponding color-quantized outputs, were used for training in realizing Chang's and Mese's algorithms.

3.5.3 Simulation Results

The SNR improvement (SNRI) achieved by different algorithms are used to compare their performance in color quantization. Mathematically, SNRI is defined as

$$\mathbf{SNRI} = 10\log\frac{\|\mathbf{X}-\mathbf{Y}\|^2}{\|\mathbf{X}-\mathbf{X}'\|^2} \tag{3.19}$$

where $\mathbf{X}$, $\mathbf{Y}$ and $\mathbf{X}'$ are the original, the color-quantized and the restored images, respectively. Table 3.1 shows the simulation results.

From Table 3.1, one can see that the performance of the proposed algorithm is better as compared with the others and it is consistent even though the input is color-quantized with different color palettes. Two schemes of the proposed algorithm were evaluated and their results are presented in the Table. Scheme A uses the observed $\mathbf{Y}$ as the initial estimate in the realization of iteration (3.17) while Scheme B uses a blurred $\mathbf{Y}$. Specifically, the blurred $\mathbf{Y}$ was obtained by filtering $\mathbf{Y}$ with a 3x3 Gaussian filter.

On average, by applying scheme A of the proposed algorithm to restore the color-quantized images, a SNRI of 2.0dB and a SNRI of 2.0dB in image quality were, respectively, achieved for inputs color-quantized with median-cut [Heckbert 82], octree [Gervautz 90]. Note the improvements with respect to the second best algorithm are, respectively, 0.72 and 0.68dB for different types of input.

When scheme B of the proposed algorithm was used, the restoration performance was even better when the color-quantized inputs were obtained with a median cut algorithm [Heckbert 82] or an octree algorithm [Gervautz 90].

Table 3.2 and Table 3.3 show the performance of the evaluated algorithms in terms of the CIELAB color difference (ΔE) metric. The CIELAB color difference (ΔE) metric is defined as the Euclidean distance between the original color of a pixel and its reproduction in CIELAB color metric space [CIE 78]. In formulation, we have $\Delta E = \sqrt{(\Delta L)^2 + (\Delta a)^2 + (\Delta b)^2}$, where ΔL, Δa and Δb represent the difference in L, a and b values of the colors. The L, a and b values of a color are, respectively, defined as $L = 116\left(\frac{Y}{Y_n}\right)^{1/3} - 16$, $a = 500\left[\left(\frac{X}{X_n}\right)^{1/3} - \left(\frac{Y}{Y_n}\right)^{1/3}\right]$ and $b = 200\left[\left(\frac{Y}{Y_n}\right)^{1/3} - \left(\frac{Z}{Z_n}\right)^{1/3}\right]$, where X, Y and Z are defined according to the color's red (R), green (G) and blue (B) components as

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} 0.4303 & 0.3416 & 0.1784 \\ 0.2219 & 0.7068 & 0.0713 \\ 0.0202 & 0.1296 & 0.9393 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} \tag{3.20}$$

(X_n, Y_n, Z_n) denotes the (X, Y, Z) that corresponds to a white point. It is well accepted that a rule of thumb for visually detectable color errors is $\Delta E > 3$

[Altunbasak 01, Connolly 95]. Table 3.2 shows the average of the ΔE values of all pixels in a restoration output and Table 3.3 shows the percentage of pixels whose color error is not visually detectable in a restoration output. Again, one can see that the proposed algorithm is superior to the others no matter which color palette is used to color-quantize the testing inputs. With this metric, it can be found that scheme B of the proposed algorithm is always better than scheme A no matter which color-quantization algorithm is used to generate the inputs. Based on the observations we have in the Tables, it is recommended to use scheme B instead of scheme A in restoring a color-quantized image.

	SNR Improvement (dB)								
	[Galatsanos 91a]	[Hunt 84]	[Altunbasak 01]-IND	[Altunbasak 01]-KL	[Kim 01]	[Mese 01]	[Chang 01]	Proposed Scheme A	Proposed Scheme B
Baboon	0.2082	0.2381	0.6678	0.6302	0.5558	-1.1673	0.1737	1.0313	1.0588
Boat	0.1985	0.3990	1.0932	1.0244	0.6423	0.6611	-0.6937	1.6678	1.8548
Couple	0.4932	1.1115	1.5100	1.4709	0.6700	1.7842	0.0949	2.2979	2.3840
Cycles	0.1333	1.2146	1.4514	1.3206	0.5573	-0.9133	-1.6107	1.5446	2.5069
Fruits	0.3048	0.8707	0.8959	0.6666	0.8438	0.7930	-0.0569	2.2209	3.0551
Girl	0.4993	1.0614	1.9525	1.9037	0.8186	2.1615	0.6053	2.7451	2.6816
Lenna	0.3881	1.4077	1.6075	1.6334	0.9283	1.1651	-0.0913	2.5159	3.1592
Peppers	0.1881	1.5510	2.1667	2.1799	0.9787	2.1976	0.4693	3.2713	3.9301
Tiffany	0.1744	0.2217	0.9405	0.8640	0.7539	-0.3670	-1.3948	1.4691	2.0945
Melon	0.0516	0.1304	0.9403	0.9846	0.7118	0.6324	-0.6235	1.6176	2.0908
Average	0.2639	0.8206	1.3226	1.2678	0.7461	0.6947	-0.3128	2.0381	2.4816

(a)

	SNR Improvement (dB)								
	[Galatsanos 91a]	[Hunt 84]	[Altunbasak 01]-IND	[Altunbasak 01]-KL	[Kim 01]	[Mese 01]	[Chang 01]	Proposed Scheme A	Proposed Scheme B
Baboon	0.1592	0.6033	0.9306	0.8715	0.5567	-0.6193	0.3205	1.0728	1.0100
Boat	0.2956	0.1443	1.0104	0.8662	0.6533	0.3747	-1.1013	1.4712	1.5766
Couple	0.5472	0.8953	1.3941	1.1468	0.7507	1.7055	0.0233	2.1436	2.1747
Cycles	0.2109	0.9742	1.1289	0.9894	0.4232	-1.0389	-1.8684	1.2543	1.4674
Fruits	0.1269	0.9274	0.8008	0.3399	0.8560	0.8228	-0.1856	2.2235	3.0747
Girl	0.6852	1.2057	2.1050	2.0058	0.8837	2.3444	0.7137	2.8174	2.7059
Lenna	0.3493	0.8224	1.2142	1.0759	0.8824	0.4260	-0.9558	2.1136	2.6365
Peppers	0.3453	1.4042	2.1813	2.0777	0.9802	2.1482	0.3870	3.3549	4.1353
Tiffany	0.3020	0.5585	1.4366	1.5018	0.7773	0.0270	-1.5021	2.1900	2.5674
Melon	0.3515	0.2716	1.2461	1.1271	0.7737	0.7294	-0.7874	1.7390	2.1415
Average	0.3373	0.7807	1.3648	1.2002	0.7537	0.6920	-0.4956	2.0480	2.3490

(b)

Table 3.1. SNR Improvements of various algorithms in restoring images color-quantized with (a) median cut algorithm [Heckbert 82], (b) octree algorithm [Gervautz 90]

	Average of CIELAB difference ΔE									
	input **Y**	restored image								
		[Galatsanos 91a]	[Hunt 84]	[Altunbasak 01]-IND	[Altunbasak 01]-KL	[Kim 01]	[Mese 01]	[Chang 01]	Proposed Scheme A	Proposed Scheme B
Baboon	3.8271	3.7852	3.5682	3.4950	3.5002	3.7238	3.4496	3.6058	3.4851	3.3999
Boat	3.5413	3.5091	3.2857	3.1313	3.1432	3.4094	3.0860	3.3955	3.1223	3.0037
Couple	7.0907	6.8206	6.1153	6.3761	6.3850	6.7513	6.1148	6.4367	5.8507	5.4060
Cycles	4.7788	4.7711	4.1709	4.1886	4.2097	4.6213	4.2765	4.5756	4.0683	3.8300
Fruits	2.6968	2.6651	2.4459	2.4196	2.4363	2.5887	2.4093	2.5613	2.2958	2.1817
Girl	5.9561	6.0030	5.2304	5.1310	5.1466	5.6240	5.1419	5.4781	4.8609	4.5851
Lenna	2.6766	2.5915	2.2659	2.2882	2.2897	2.5531	2.3343	2.5359	2.1605	2.0748
Peppers	4.0172	4.0420	3.7577	3.6311	3.6303	3.8058	3.4798	3.8771	3.2641	3.0765
Tiffany	1.3160	1.2974	1.2468	1.1754	1.1778	1.2607	1.2386	1.2881	1.1620	1.1076
Melon	4.2157	4.2559	4.2600	3.8648	3.8620	4.0077	3.8012	3.9935	3.7326	3.5669
Average	4.0116	3.9741	3.6347	3.5701	3.5781	3.8346	3.5332	3.7748	3.4002	3.2232

(a)

	Average of CIELAB difference ΔE									
	input **Y**	restored image								
		[Galatsanos 91a]	[Hunt 84]	[Altunbasak 01]-IND	[Altunbasak 01]-KL	[Kim 01]	[Mese 01]	[Chang 01]	Proposed Scheme A	Proposed Scheme B
Baboon	3.9743	3.9381	3.6654	3.5961	3.6072	3.8698	3.5488	3.7527	3.6490	3.5460
Boat	3.3683	3.3237	3.1713	3.0294	3.0436	3.2626	2.9865	3.2690	3.0235	2.9491
Couple	6.0473	5.9530	5.9193	5.7894	5.8212	5.8641	5.4955	5.7416	5.4088	5.2330
Cycles	4.8500	4.8214	4.3412	4.3300	4.3638	4.7673	4.4801	4.7325	4.3141	4.1666
Fruits	2.8303	2.8084	2.5729	2.5466	2.5836	2.7366	2.5591	2.7388	2.4260	2.3138
Girl	5.3507	5.2992	5.1688	4.8127	4.8270	5.1397	4.7698	5.0560	4.5897	4.4641
Lenna	2.2740	2.2301	2.0712	2.0417	2.0563	2.1950	2.0617	2.2336	1.9685	1.9003
Peppers	4.3351	4.2590	4.1693	3.9921	4.0101	4.2318	3.9336	4.2999	3.7224	3.6219
Tiffany	1.6494	1.6091	1.5076	1.4414	1.4346	1.5854	1.5275	1.6187	1.4004	1.3319
Melon	3.9534	3.9165	3.8945	3.5780	3.5902	3.7899	3.5957	3.8389	3.4764	3.3332
Average	3.8633	3.8158	3.6482	3.5157	3.5338	3.7442	3.4958	3.7282	3.3979	3.2860

(b)

Table 3.2. CIELAB difference ΔE measurement of outputs of various algorithms in restoring images color-quantized with (a) median cut algorithm [Heckbert 82], (b) octree algorithm [Gervautz 90]

	% of pixels whose CIELAB ΔE <3									
	input **Y**	restored image								
		[Galatsanos 91a]	[Hunt 84]	[Altunbasak 01]-IND	[Altunbasak 01]-KL	[Kim 01]	[Mese 01]	[Chang 01]	Proposed Scheme A	Proposed Scheme B
Baboon	44.67	44.30	49.53	49.32	49.29	45.44	49.77	47.33	49.68	51.41
Boat	52.96	52.80	57.04	58.38	58.13	54.06	58.38	54.07	58.14	60.11
Couple	18.59	19.45	25.40	25.99	25.85	19.40	24.83	21.05	27.58	29.42
Cycles	37.78	37.45	44.14	45.52	45.19	38.53	42.66	38.88	46.27	49.99
Fruits	72.24	72.04	76.47	76.26	75.99	73.14	76.12	73.48	77.88	79.27
Girl	29.02	30.04	32.98	35.49	35.38	30.46	34.53	31.11	37.71	38.85
Lenna	66.65	67.58	75.55	74.90	74.86	68.68	73.86	69.44	77.81	79.48
Peppers	47.05	46.11	52.82	55.60	55.62	49.68	56.00	49.59	60.31	63.77
Tiffany	95.30	95.28	95.24	96.22	96.23	95.52	95.84	95.58	96.23	96.45
Melon	44.53	44.04	48.64	49.12	49.16	45.82	48.53	46.18	49.01	51.90
Average	50.88	50.91	55.78	56.68	56.57	52.07	56.05	52.67	58.06	60.07

(a)

	% of pixels whose CIELAB ΔE <3									
	input Y	restored image								
		[Galatsanos 91a]	[Hunt 84]	[Altunbasak 01]-IND	[Altunbasak 01]-KL	[Kim 01]	[Mese 01]	[Chang 01]	Proposed Scheme A	Proposed Scheme B
Baboon	38.42	38.10	45.37	43.98	43.84	39.17	44.61	40.87	42.90	30.31
Boat	54.96	54.77	58.58	59.92	59.72	55.68	60.02	55.59	59.51	60.79
Couple	20.07	20.65	24.26	25.09	24.69	20.95	24.51	21.96	25.66	26.97
Cycles	40.48	40.16	45.25	46.13	45.58	41.19	43.91	41.05	47.11	50.39
Fruits	65.11	64.60	71.99	71.41	70.63	66.19	69.87	66.16	73.43	76.29
Girl	26.68	26.79	31.18	33.62	33.43	28.45	32.89	28.80	36.02	37.64
Lenna	76.62	76.91	80.67	81.68	81.32	77.80	81.29	77.54	83.12	84.04
Peppers	42.75	42.74	47.41	50.03	49.77	44.04	48.29	44.45	53.14	55.58
Tiffany	92.73	93.27	93.66	95.33	95.62	93.62	94.35	93.47	95.95	96.23
Melon	49.03	48.74	48.73	51.08	50.98	49.48	51.51	48.85	52.48	55.07
Average	50.69	50.67	54.71	55.83	55.56	51.66	55.13	51.87	56.93	57.33

(b)

Table 3.3. Percentage of pixels whose CIELAB difference is not detectable (ΔE <3) after restoration when testing images were color-quantized with (a) median cut algorithm [Heckbert 82], (b) octree algorithm [Gervautz 90]

Figure 3.2 shows the restoration results of different algorithms for visual evaluation. Figure 3.2b is the color-quantized version of Figure 3.2a. The palette was generated with median-cut algorithm. One can see the false contour and the color shift in the bikes. After restoration, the proposed algorithm removes most of the artefacts while some of the others cannot do the job. Scheme B of the proposed algorithm provides a better visual result as compared with scheme A and is the best among the algorithms.

In the proposed algorithm, $\vec{\sigma}_k$ is estimated with the blurred **Y** instead of the original **X**. A study was performed to study how well this estimation could be done. In our study, the used measure was *normalized weighted error* **(NWE)** and it is defined as

$$\mathbf{NWE} = \frac{\sum_{k=1}^{N_c} w_k \left\| \vec{\sigma}''_k - \vec{\sigma}'_k \right\|^2}{\sum_{k=1}^{N_c} w_k \left\| \vec{\sigma}''_k \right\|^2} \times 100\% \qquad (3.21)$$

where $\vec{\sigma}''_k$ and $\vec{\sigma}'_k$ are, respectively, the $\vec{\sigma}_k$'s obtained with **X** and the blurred **Y**, and w_k is the number of $\hat{v}_k$ found in **Y**. Table 4 shows the **NWEs** when different testing images and their corresponding color-quantized **Y's** were used. On average, it is around 0.08. Besides, it was found that the impact of the difference between $\vec{\sigma}''_k$ and $\vec{\sigma}'_k$ to the final restoration result was little. The corresponding difference of the results is less than 0.1dB in SNRI on average.

(a) Original (b) Color-quantized

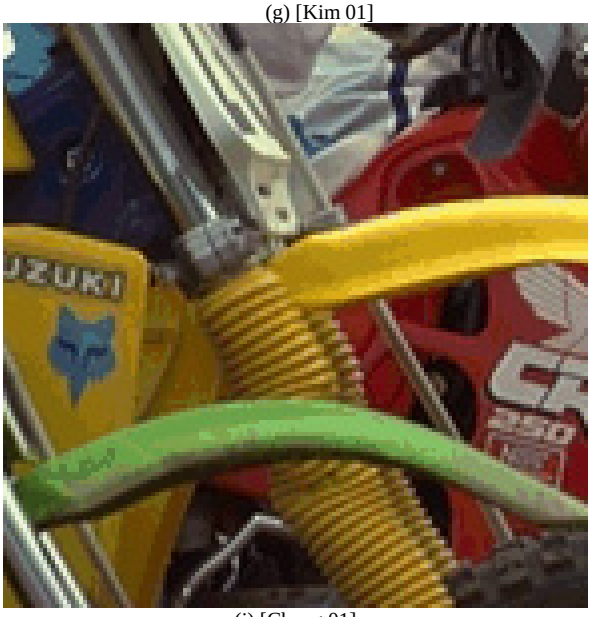
(c) [Galatsanos 91a] (d) [Hunt 84]

(e) [Altunbasak 01]-IND (f) [Altunbasak 01]-KL

Figure 3.2. Zoomed outputs of various approaches in restoring color-quantized *Cycles*

(g) [Kim 01]

(h) [Mese 01]



(i) [Chang 01]

(j) Proposed (Scheme A)

(k) Proposed (Scheme B)

Figure 3.2 (Continue). Zoomed outputs of various approaches in restoring color-quantized *Cycles*

	NWE (%)	
	Median-cut [Heckbert 82]	Octree [Gervautz 90]
Baboon	3.145	3.591
Boat	9.827	6.136
Couple	8.489	7.778
Cycles	5.902	6.636
Fruits	8.564	10.497
Girl	7.129	6.454
Lenna	5.149	6.641
Peppers	8.166	7.316
Tiffany	13.628	7.027
Melon	8.422	4.880
Average	7.842	6.695

Table 3.4. Discrepancy measured when $\vec{\sigma}_k$ is evaluated with the blurred observed image instead of the original image.

3.6 Summary

By far, very little research has been carried out to address the restoration of color-quantized images. Though there are some restoration algorithms for restoring blurred and noisy color images, they are generally not adequate to handle color-quantized images. In this Chapter, we proposed a dedicated restoration algorithm for restoring color-quantized images. This algorithm makes good use of the color palette to derive useful *a priori* information for restoration. It has been demonstrated by simulation results that the proposed algorithm can achieve a remarkable improvement in the quality of a color-quantized image in terms of both SNR and CIELAB color difference (ΔE) metric. Its performance is obviously better than other existing restoration approaches such as [Altunbasak 01, Galatsanos 91a, Hunt 84, Kim 01].

The proposed restoration algorithm requires no extra information other than the color palette to carry out the restoration. Besides, it is not tailor-made for a particular color quantization scheme and hence no *a priori* knowledge about the

construction of the color palette is required during the restoration. This makes it always able to provide a reasonable restoration performance whatever color quantization scheme with which it works. Simulation results verify that this is true.

Chapter 4

A POCS-based Algorithm for Restoring Color-quantized Images

4.1 Introduction

This Chapter presents another restoration algorithm for restoring color-quantized image by making use of POCS theory [Youla 82].

The organization of this Chapter is as follows. In Section 4.2, we formulate the *a priori* information we can have about the degradation process of color quantization and about the original image. In Section 4.3, we formulate the constraints for image restoration. In Section 4.4, we formulate the proposed algorithms for restoring color-quantized images based on the theory of projection onto convex sets. Simulation results for comparative study are provided in Section 4.5. Finally, a brief summary is given in Section 4.6.

4.2 Formulation of *A Priori* Information

A color image $\mathbf{X}$ generally consists of three color planes, say, $\mathbf{X}_r$, $\mathbf{X}_g$, and $\mathbf{X}_b$, which represent the red, green and blue color planes of the image, respectively.

The $(i, j)^{th}$ pixel of the image is hence a 3D vector represented by $\vec{\mathbf{X}}_{(i,j)} = (\mathbf{X}_{(i,j)r}, \mathbf{X}_{(i,j)g}, \mathbf{X}_{(i,j)b})^t$ where $\mathbf{X}_{(i,j)c} \in [0,1]$ is the intensity value of the c^{th} color component of the $(i, j)^{th}$ pixel and t denotes the transpose of a matrix. Here, we assume that the maximum and the minimum intensity values of a pixel are, respectively, 1 and 0.

In this Chapter, all vectors in a 3D space are represented in a column vector form and the *a priori* information about the color quantization without error diffusion formulated in Chapter 3.2 is used.

4.3 Formulation of Constraints for Restoration

Let $\vec{\mathbf{X}}_{(i,j)}$ and $\vec{\mathbf{Y}}_{(i,j)}$ be the 3-dimensional vectors representing the $(i, j)^{th}$ pixels of the original image $\mathbf{X}$ and the encoded image $\mathbf{Y}$, respectively. If $\hat{\mathbf{v}}_k \in C$ is the color used to represent $\vec{\mathbf{X}}_{(i,j)}$, then we will have $\vec{\mathbf{Y}}_{(i,j)} = \hat{\mathbf{v}}_k$, and the distance between $\vec{\mathbf{Y}}_{(i,j)}$ and $\vec{\mathbf{X}}_{(i,j)}$ will be bounded by the boundary of the Voronoi region R_k. More specifically, any particular principal component of $\vec{\mathbf{X}}_{(i,j)} - \vec{\mathbf{Y}}_{(i,j)}$, say, $[T_k(\vec{\mathbf{X}}_{(i,j)} - \vec{\mathbf{Y}}_{(i,j)})]_s$, is also bounded. In formulation, we have

$$([T_k(\vec{\mathbf{X}}_{(i,j)} - \vec{\mathbf{Y}}_{(i,j)})]_s)^2 \le \varepsilon_{(i,j)s} \tag{4.1}$$

where $\varepsilon_{(i,j)s}$ is the corresponding bound.

Since we have $\vec{\mathbf{Y}}_{(i,j)} = \hat{\mathbf{v}}_k$, the bound $\varepsilon_{(i,j)s}$ can be estimated to be a function of $\sigma_{k,s}^2$. By assuming that $\varepsilon_{(i,j)s}$ is proportional to $\sigma_{k,s}^2$, we have $\left|[T_k(\vec{\mathbf{X}}_{(i,j)} - \vec{\mathbf{Y}}_{(i,j)})]_s\right| \le \beta\sigma_{k,s}$, where β is a scaling parameter. This forms a constraint

set $S_1 = \{\mathbf{I} \mid \vec{\mathbf{I}}_{(i,j)} \in \Gamma$ and $\left|[T_k(\vec{\mathbf{X}}_{(i,j)} - \vec{\mathbf{Y}}_{(i,j)})]_s\right| \le \beta\sigma_{k,s}$ for all $i, j\}$, where k is the index of the codeword $\hat{\mathbf{v}}_k = \vec{\mathbf{Y}}_{(i,j)}$ to restore $\mathbf{X}$.

Typical images would generally have weak high frequency components as the intensity of neighboring pixels is highly correlated. This feature can be exploited as an additional *a priori* information in the restoration of color-quantized images. The conventional approach to incorporate this information into restoration is to assume that the energy of the high frequency components of image $\mathbf{X}$ is bounded.

In our approach, we assume that the energy of each high frequency component of $\mathbf{X}$ is bounded and the bound of each component is estimated with the low-pass filtered $\mathbf{Y}$. In particular, we have $\left|[T(\mathbf{X})]_{(u,w)}\right| \le \left|[T(F(\mathbf{Y})]_{(u,w)}\right|$ for $(u,w) \in \Omega_H$, where F and T are, respectively, a linear low-pass filtering operator and a 2D DCT operator, $[\bullet]_{(u,w)}$ denotes the $(u,w)^{th}$ element in the transform domain and Ω_H defines the set of high frequency components which should be bounded. This forms a smoothness constraint set $S_0 = \{\mathbf{I} \mid \vec{\mathbf{I}}_{(i,j)} \in \Gamma$ and $\left|[T(\mathbf{I})]_{(u,w)}\right| \le \left|[T(F(\mathbf{Y})]_{(u,w)}\right|$ for $(u,w) \in \Omega_H\}$ for restoring image $\mathbf{X}$.

Two more constraint sets can be used to restore $\mathbf{X}$. One is that which confines the intensity value of a particular pixel to be valid. In formulation, we have $S_2 = \{\mathbf{I} \mid \vec{\mathbf{I}}_{(i,j)} \in \Gamma$ and $0 \le \mathbf{I}_{(i,j)c} \le 1$ for all $i, j\}$. The other is $S_3 = \{\mathbf{I} \mid \vec{\mathbf{I}}_{(i,j)} \in \Gamma$ and $Q(\vec{\mathbf{I}}_{(i,j)}) = \vec{\mathbf{Y}}_{(i,j)}$ for all $i, j\}$, where Q is the color quantization operator. This set makes sure that the color-quantized output of the restored $\mathbf{X}$ is $\mathbf{Y}$.

4.4 Formulation of the POCS Algorithm

Based on the *a priori* information concerning the color quantization process and the original image, four constraint sets are defined in Section 4.3. All of these constraint sets are convex sets. Based on the theory of projection onto convex sets (POCS) [Youla 82], which states that a good estimate of the original image can be obtained by successively projecting an estimate onto each of convex constraint sets until it converges to an element in the intersection of all these sets, an iterative algorithm can be defined as

$$\mathbf{X}^{(n+1)} = P_3 P_2 P_1 P_0 \mathbf{X}^{(n)} \tag{4.2}$$

where $\mathbf{X}^{(n)}$ is the estimate of $\mathbf{X}$ at iteration n and P_i is a projection operator projecting a given image $\mathbf{I}$ onto S_i, where $i \in \{0,1,2,3\}$. In particular, they are defined as

$$P_0 : [T(\mathbf{I})]_{(u,w)} = [T(F(\mathbf{Y}))]_{(u,w)} \qquad if \ \left|[T(\mathbf{I})]_{(u,w)}\right| > \left|[T(F(\mathbf{Y}))]_{(u,w)}\right| \ \ for \ \ (u,w) \in \Omega_H \tag{4.3}$$

$$P_1 : [T_k \vec{\mathbf{I}}_{(i,j)}]_s = \begin{cases} [T_k \vec{\mathbf{Y}}_{(i,j)}]_s + \beta\sigma_{k,s} & if \ \ [T_k \vec{\mathbf{I}}_{(i,j)}]_s > [T_k \vec{\mathbf{Y}}_{(i,j)}]_s + \beta\sigma_{k,s} \\ [T_k \vec{\mathbf{Y}}_{(i,j)}]_s - \beta\sigma_{k,s} & if \ \ [T_k \vec{\mathbf{I}}_{(i,j)}]_s < [T_k \vec{\mathbf{Y}}_{(i,j)}]_s - \beta\sigma_{k,s} \end{cases}$$

$$for \ \ s \in \{1,2,3\}, \ \forall (i,j) \tag{4.4}$$

where k is the index of the codeword $\hat{\mathbf{v}}_k = \vec{\mathbf{Y}}_{(i,j)}$.

$$P_2 : \mathbf{I}_{(i,j)c} = \begin{cases} 1 & if \ \ \mathbf{I}_{(i,j)c} > 1 \\ 0 & if \ \ \mathbf{I}_{(i,j)c} < 0 \end{cases} \qquad for \ \ c \in \{r,g,b\}, \quad \forall (i,j) \tag{4.5}$$

$$P_3 : \vec{\mathbf{I}}_{(i,j)} = \vec{\mathbf{Y}}_{(i,j)} \quad if \ \ Q(\vec{\mathbf{I}}_{(i,j)}) \neq \vec{\mathbf{Y}}_{(i,j)} \qquad \forall (i,j) \tag{4.6}$$

The initial estimate $\mathbf{X}^{(0)}$ is set to be $F(\mathbf{Y})$, although theoretically no restriction is imposed on the initial estimate.

Based on a similar idea presented in [Kim 01], it is desirable to keep S_1 to be a subset of S_3. At the same time, the size of S_1 should be as big as possible under the constraint so as not to exclude the original $\mathbf{X}$ in the set of potential candidates. In our proposed algorithm, in order to achieve both goals, we first select an appropriate β such that S_1 and S_3 are of similar size at the beginning and then reduce the size of S_1 by adjusting the value of β at each iteration until $S_1 \subset S_3$ is achieved. Specifically, projection P_1 is modified as

$$P_1 : [T_k \vec{\mathbf{I}}_{(i,j)}]_s = \begin{cases} [T_k \vec{\mathbf{Y}}_{(i,j)}]_s + \beta_k \sigma_{k,s} & \text{if } [T_k \vec{\mathbf{I}}_{(i,j)}]_s > [T_k \vec{\mathbf{Y}}_{(i,j)}]_s + \beta_k \sigma_{k,s} \\ [T_k \vec{\mathbf{Y}}_{(i,j)}]_s - \beta_k \sigma_{k,s} & \text{if } [T_k \vec{\mathbf{I}}_{(i,j)}]_s < [T_k \vec{\mathbf{Y}}_{(i,j)}]_s - \beta_k \sigma_{k,s} \end{cases} \tag{4.7}$$

During the realization of projection P_3, if $Q(\vec{\mathbf{I}}_{(i,j)}) \neq \vec{\mathbf{Y}}_{(i,j)}$ happens, the β_k associated with codeword $\hat{\mathbf{v}}_k = \vec{\mathbf{Y}}_{(i,j)}$ is adjusted by $\beta_k = \alpha\beta_k$, where $\alpha(<1)$ is a scaling parameter which controls the rate of adjustment.

Note that this adjustment does not affect the convergence. The adjustment reduces the size of S_1. The adjusted S_1 is a convex set and hence, at any time, all constraint sets are convex sets. Eventually, S_1 becomes a subset of S_3 and there will be no more adjustment. When this point is reached, it becomes a typical POCS algorithm and the convergence can be guaranteed.

4.5 Simulation and Comparative Study

In this Section, the performace evaluation of the proposed algorithm is evaluated and a comparative study on the performance of different algorithm is made.

First, the details of the simulations and comparative study are given in Section 4.5.1. Simulation results and their implications are discussed in Section 4.5.2.

4.5.1 Realization Details of the Proposed Algorithm

Simulation has been carried out to evaluate the performance of the proposed algorithm on a set of color-quantized images. In our simulation, a number of 24-bit full color test images were first color-quantized with a color palette of 256 colors. In our study, two color palettes were generated with median cut algorithm [Heckbert 82] and octree algorithm [Gervautz 90], respectively. They were used to investigate if the algorithm worked with inputs obtained with different color palettes. No halftoning was performed during the quantization. The test images applied were a set of *de facto* standard 24-bit full color images of size 256×256 each.

The proposed restoration algorithm was then applied to restore the quantized images. In the realization of the proposed algorithm, the observed color image $\mathbf{Y}$ was blurred with a 3×3 average filter to generate a training set to estimate $\vec{\sigma}_k$. The 3×3 Gaussian filter was used as filter F and Ω_H was defined as $\{(u, w) \mid (256-u)^2 + (256-w)^2 < 80000$ and $0 \le u, w < 256\}$. The initial value of β_k was assigned to be $\beta_k = 0.05/\sigma$ for all k, where σ is the maximum of $\sigma_{k,s}$s for all k and s, and α was assigned to be 0.8. The threshold value 80000 was obtained empirically.

By considering that the extreme case happens when the whole color space is uniformly partitioned into slices of equal width and that colors are uniformly

distributed in the color space, the lower bound of $\sigma_{k,s}^2$ is set to be $\frac{1}{12}\left(\frac{1}{N_c}\right)^2$, where N_c is the total number of colors in the palette.

	SNR Improvement (dB)									
	[Galatsanos 91a]	[Hunt 84]	[Altunbasak 01]-IND	[Altunbasak 01]-KL	[Kim 01]	[Mese 01]	[Chang 01]	[Chan 05] (Proposed Schemes in Chapter 3)		Proposed
								A	B	
Baboon	0.2082	0.2381	0.6678	0.6302	0.5558	-1.1673	0.1737	1.0313	1.0588	0.8340
Boat	0.1985	0.3990	1.0932	1.0244	0.6423	0.6611	-0.6937	1.6678	1.8548	1.1627
Couple	0.4932	1.1115	1.5100	1.4709	0.6700	1.7842	0.0949	2.2979	2.3840	2.1252
Cycles	0.1333	1.2146	1.4514	1.3206	0.5573	-0.9133	-1.6107	1.5446	2.5069	1.5171
Fruits	0.3048	0.8707	0.8959	0.6666	0.8438	0.7930	-0.0569	2.2209	3.0551	2.0592
Girl	0.4993	1.0614	1.9525	1.9037	0.8186	2.1615	0.6053	2.7451	2.6816	2.7180
Lenna	0.3881	1.4077	1.6075	1.6334	0.9283	1.1651	-0.0913	2.5159	3.1592	2.7183
Peppers	0.1881	1.5510	2.1667	2.1799	0.9787	2.1976	0.4693	3.2713	3.9301	2.7374
Tiffany	0.1744	0.2217	0.9405	0.8640	0.7539	-0.3670	-1.3948	1.4691	2.0945	1.3421
Melon	0.0516	0.1304	0.9403	0.9846	0.7118	0.6324	-0.6235	1.6176	2.0908	1.3631
Average	0.2640	0.8206	1.3226	1.2678	0.7460	0.6947	-0.3128	2.0382	2.4816	1.8577

(a)

	SNR Improvement (dB)									
	[Galatsanos 91a]	[Hunt 84]	[Altunbasak 01]-IND	[Altunbasak 01]-KL	[Kim 01]	[Mese 01]	[Chang 01]	[Chan 05] (Proposed Schemes in Chapter 3)		Proposed
								A	B	
Baboon	0.1592	0.6033	0.9306	0.8715	0.5567	-0.6193	0.3205	1.0728	1.0100	1.1023
Boat	0.2956	0.1443	1.0104	0.8662	0.6533	0.3747	-1.1013	1.4712	1.5766	0.8919
Couple	0.5472	0.8953	1.3941	1.1468	0.7507	1.7055	0.0233	2.1436	2.1747	1.5417
Cycles	0.2109	0.9742	1.1289	0.9894	0.4232	-1.0389	-1.8684	1.2543	1.4674	1.2023
Fruits	0.1269	0.9274	0.8008	0.3400	0.8560	0.8228	-0.1856	2.2235	3.0747	2.0337
Girl	0.6852	1.2057	2.1050	2.0058	0.8837	2.3444	0.7137	2.8174	2.7059	2.8007
Lenna	0.3493	0.8224	1.2142	1.0759	0.8824	0.4260	-0.9558	2.1136	2.6365	2.0956
Peppers	0.3453	1.4042	2.1813	2.0777	0.9802	2.1482	0.3870	3.3549	4.1353	2.2682
Tiffany	0.3020	0.5585	1.4366	1.5018	0.7773	0.0270	-1.5021	2.1900	2.5674	2.2297
Melon	0.3515	0.2716	1.2461	1.1271	0.7737	0.7294	-0.7874	1.7390	2.1415	1.6020
Average	0.3373	0.7807	1.3448	1.2002	0.7537	0.6920	-0.4956	2.0380	2.3490	1.7768

(b)

Table 4.1. SNR Improvements of various algorithms in restoring images color-quantized with (a) median cut algorithm [Heckbert 82], (b) octree algorithm [Gervautz 90]

	Average of CIELAB difference ΔE										
		restored image									
	input **Y**	[Galatsanos 91a]	[Hunt 84]	[Altunbasak 01]-IND	[Altunbasak 01]-KL	[Kim 01]	[Mese 01]	[Chang 01]	[Chan 05] (Proposed Schemes in Chapter 3)		Proposed
									A	B	
Baboon	3.8271	3.7852	3.5682	3.4950	3.5002	3.7238	3.4496	3.6058	3.4851	3.3999	3.3882
Boat	3.5413	3.5091	3.2857	3.1313	3.1432	3.4094	3.0860	3.3955	3.1223	3.0037	3.0048
Couple	7.0907	6.8206	6.1153	6.3761	6.3850	6.7513	6.1148	6.4367	5.8507	5.4060	5.9214
Cycles	4.7788	4.7711	4.1709	4.1886	4.2097	4.6213	4.2765	4.5756	4.0683	3.8300	4.1047
Fruits	2.6968	2.6651	2.4459	2.4196	2.4363	2.5887	2.4093	2.5613	2.2958	2.1817	2.3520
Girl	5.9561	6.0030	5.2304	5.1310	5.1466	5.6240	5.1419	5.4781	4.8609	4.5851	4.8354
Lenna	2.6766	2.5915	2.2659	2.2882	2.2897	2.5531	2.3343	2.5359	2.1605	2.0748	2.1521
Peppers	4.0172	4.0420	3.7577	3.6311	3.6303	3.8058	3.4798	3.8771	3.2641	3.0765	3.4618
Tiffany	1.3160	1.2974	1.2468	1.1754	1.1778	1.2607	1.2386	1.2881	1.1620	1.1076	1.1480
Melon	4.2157	4.2559	4.2600	3.8648	3.8620	4.0077	3.8012	3.9935	3.7326	3.5669	3.7724
Average	4.0116	3.9741	3.6347	3.5701	3.5781	3.8346	3.5332	3.7748	3.4002	3.2232	3.4141

(a)

	Average of CIELAB difference ΔE										
		restored image									
	input **Y**	[Galatsanos 91a]	[Hunt 84]	[Altunbasak 01]-IND	[Altunbasak 01]-KL	[Kim 01]	[Mese 01]	[Chang 01]	[Chan 05] (Proposed Schemes in Chapter 3)		Proposed
									A	B	
Baboon	3.9743	3.9381	3.6654	3.5961	3.6072	3.8698	3.5488	3.7527	3.6490	3.5460	3.4836
Boat	3.3683	3.3237	3.1713	3.0294	3.0436	3.2626	2.9865	3.2690	3.0235	2.9491	3.0084
Couple	6.0473	5.9530	5.9193	5.7894	5.8212	5.8641	5.4955	5.7416	5.4088	5.2330	5.4259
Cycles	4.8500	4.8214	4.3412	4.3300	4.3638	4.7673	4.4801	4.7325	4.3141	4.1666	4.3217
Fruits	2.8303	2.8084	2.5729	2.5466	2.5836	2.7366	2.5591	2.7388	2.4260	2.3138	2.4688
Girl	5.3507	5.2992	5.1688	4.8127	4.8270	5.1397	4.7698	5.0560	4.5897	4.4641	4.5825
Lenna	2.2740	2.2301	2.0712	2.0417	2.0563	2.1950	2.0617	2.2336	1.9685	1.9003	1.9937
Peppers	4.3351	4.2590	4.1693	3.9921	4.0101	4.2318	3.9336	4.2999	3.7224	3.6219	3.9439
Tiffany	1.6494	1.6091	1.5076	1.4414	1.4346	1.5854	1.5275	1.6187	1.4004	1.3319	1.4203
Melon	3.9534	3.9165	3.8945	3.5780	3.5902	3.7899	3.5957	3.8389	3.4764	3.3332	3.5012
Average	3.8633	3.8158	3.6482	3.5157	3.5338	3.7442	3.4958	3.7282	3.3979	3.2860	3.4150

(b)

Table 4.2. CIELAB difference ΔE measurement of outputs of various algorithms in restoring images color-quantized with (a) median cut algoroithm [Heckbert 82], (b) octree algorithm [Gervautz 90]

	% of pixels whose CIELAB $\Delta E < 3$										
		restored image									
	input **Y**	[Galatsanos 91a]	[Hunt 84]	[Altunbasak 01]-IND	[Altunbasak 01]-KL	[Kim 01]	[Mese 01]	[Chang 01]	[Chan 05] (Proposed Schemes in Chapter 3)		Proposed
									A	B	
Baboon	44.67	44.30	49.53	49.32	49.29	45.44	49.77	47.33	49.68	51.41	52.04
Boat	52.96	52.80	57.04	58.38	58.13	54.06	58.38	54.07	58.14	60.11	60.66
Couple	18.59	19.45	25.40	25.99	25.85	19.40	24.83	21.05	27.58	29.42	25.66
Cycles	37.78	37.45	44.14	45.52	45.19	38.53	42.66	38.88	46.27	49.99	45.82
Fruits	72.24	72.04	76.47	76.26	75.99	73.14	76.12	73.48	77.88	79.27	77.19
Girl	29.02	30.04	32.98	35.49	35.38	30.46	34.53	31.11	37.71	38.85	37.64
Lenna	66.65	67.58	75.55	74.90	74.86	68.68	73.86	69.44	77.81	79.48	78.53
Peppers	47.05	46.11	52.82	55.60	55.62	49.68	56.00	49.59	60.31	63.77	57.69
Tiffany	95.30	95.28	95.24	96.22	96.23	95.52	95.84	95.58	96.23	96.45	96.25
Melon	44.53	44.04	48.64	49.12	49.16	45.82	48.53	46.18	49.01	51.90	48.58
Average	50.88	50.91	55.78	56.68	56.57	52.07	56.05	52.67	58.06	60.07	58.01

(a)

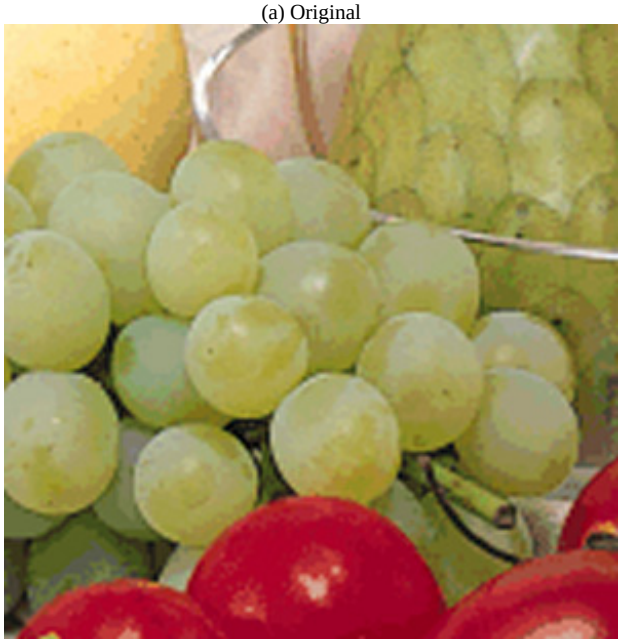
	% of pixels whose CIELAB $\Delta E < 3$										
		restored image									
	input **Y**	[Galatsanos 91a]	[Hunt 84]	[Altunbasak 01]-IND	[Altunbasak 01]-KL	[Kim 01]	[Mese 01]	[Chang 01]	[Chan 05] (Proposed Schemes in Chapter 3)		Proposed
									A	B	
Baboon	38.42	38.10	45.37	43.98	43.84	39.17	44.61	40.87	42.90	30.31	46.95
Boat	54.96	54.77	58.58	59.92	59.72	55.68	60.02	55.59	59.51	60.79	60.59
Couple	20.07	20.65	24.26	25.09	24.69	20.95	24.51	21.96	25.66	26.97	26.15
Cycles	40.48	40.16	45.25	46.13	45.58	41.19	43.91	41.05	47.11	50.39	46.69
Fruits	65.11	64.60	71.99	71.41	70.63	66.19	69.87	66.16	73.43	76.29	72.32
Girl	26.68	26.79	31.18	33.62	33.43	28.45	32.89	28.80	36.02	37.64	36.89
Lenna	76.62	76.91	80.67	81.68	81.32	77.80	81.29	77.54	83.12	84.04	82.11
Peppers	42.75	42.74	47.41	50.03	49.77	44.04	48.29	44.45	53.14	55.58	48.63
Tiffany	92.73	93.27	93.66	95.33	95.62	93.62	94.35	93.47	95.95	96.23	95.75
Melon	49.03	48.74	48.73	51.08	50.98	49.48	51.51	48.85	52.48	55.07	52.11
Average	50.69	50.67	54.71	55.83	55.56	51.66	55.13	51.87	56.93	57.33	56.82

(b)

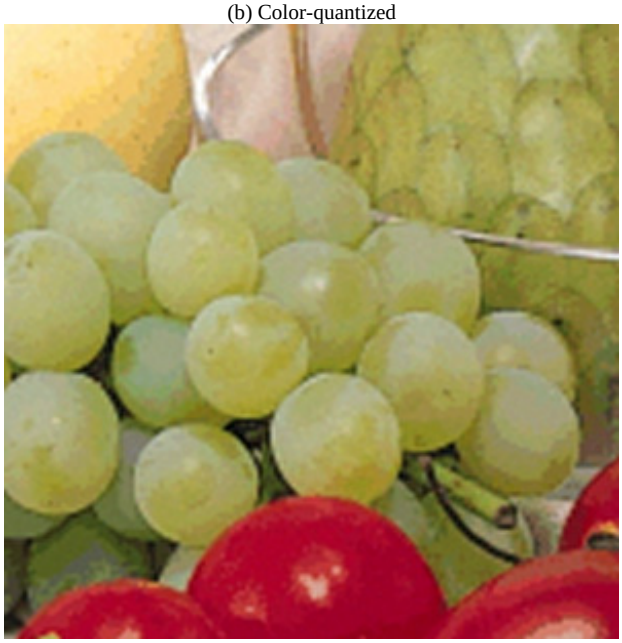
Table 4.3. Percentage of pixels whose CIELAB difference is not detectable (ΔE <3) after restoration when testing images were color-quantized with (a) median cut algorithm [Heckbert 82], (b) octree algorithm [Gervautz 90]

(a) Original

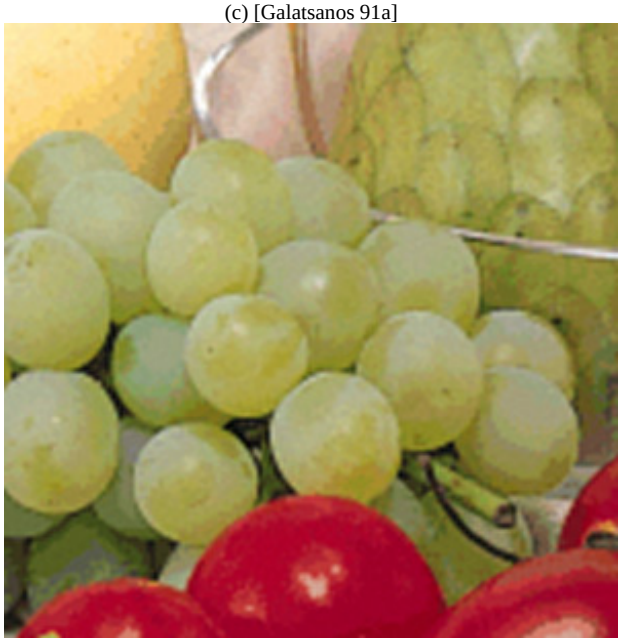
(b) Color-quantized



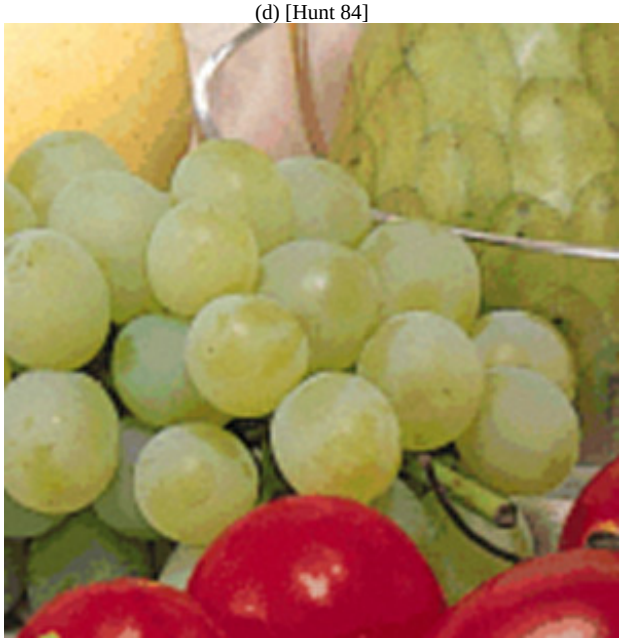
(c) [Galatsanos 91a]



(d) [Hunt 84]



(e) [Altunbasak 01]-IND

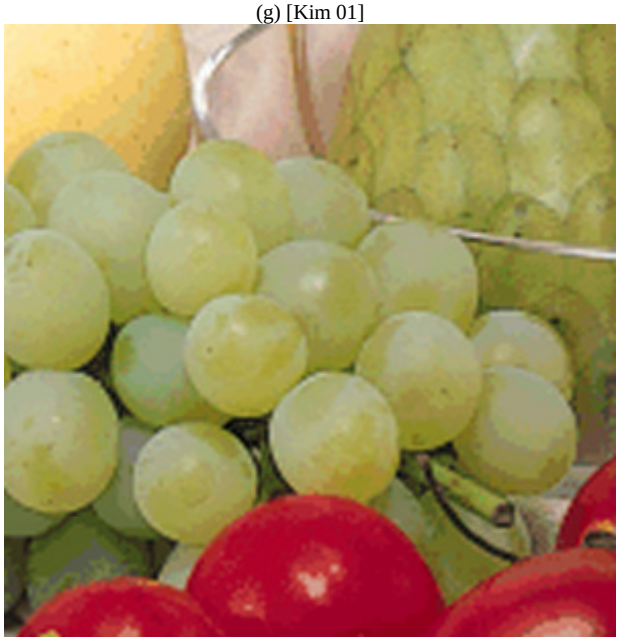


(e) [Altunbasak 01]-KL

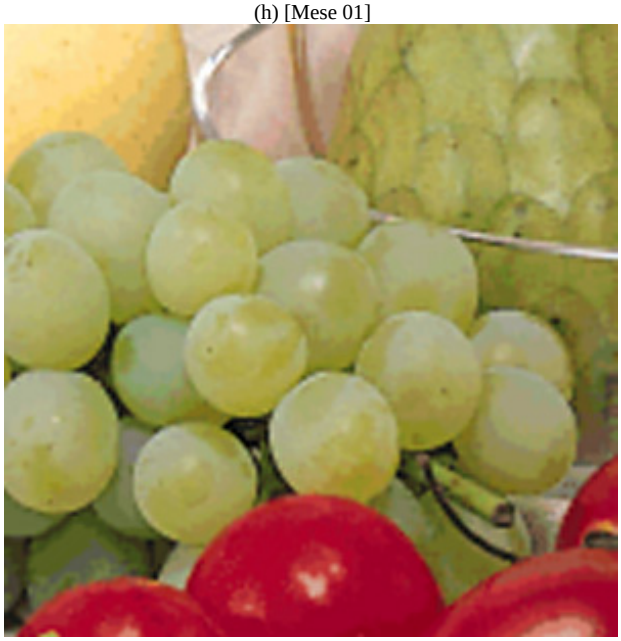
Figure 4.1. Zoomed outputs of various approaches in restoring color-quantized Fruits.

(g) [Kim 01]

(h) [Mese 01]



(i) [Chang 01]



(j) Proposed Scheme in Chapter 3 (Scheme A)

(k) Proposed Scheme in Chapter 3 (Scheme B)

(l) Proposed

Figure 4.1(continue). Zoomed outputs of various approaches in restoring color-quantized Fruits.

4.5.2 Algorithms for Comparative Study

For comparison, the same set of evaluated restoration algorithms and measurements used in Chapter 3, Section 3.5.2 and Section 3.5.3 were used. Tables 4.1-4.3 show the performance of various algorithms in restoring images color-quantized with a 256-color palette. The palette was generated with a median cut algorithm [Heckbert 82] and octree algorithm [Gervautz 90].

For easier comparison, we duplicate the result from the Chapter 3 here. From Tables 4.1, one can see that the performance of the proposed algorithm is better as compared with the others. On average, by applying the proposed algorithm to restore the color-quantized images, a SNRI of 1.9dB, a SNRI of 1.8dB in image quality were, respectively, achieved for inputs color-quantized with median-cut [Heckbert 82], octree [Gervautz 90].

Tables 4.2 and 4.3 show the performance of the evaluated algorithms in terms of the CIELAB color difference (ΔE) metric. Table 4.2 shows the average of the ΔE values of all pixels in a restoration output and Table 4.3 shows the percentage of pixels whose color error is visually undetectable in a restoration output.

Figure 4.1 shows the restoration results of different algorithms for visual evaluation. Figure 4.1b is the color-quantized version of Figure 4.1a. The palette was generated with median-cut algorithm. One can see the false contour and the color shift in the Fruits. After restoration, the proposed algorithm removes most of the artefacts while some of the others cannot do the job.

4.5.3 Simulation Results and Implications

As compare with the restoration algorithm proposed in Chapter 3, the restoration results proposed in this Chapter are a little bit lower in various aspects. Based on the simulation results, we observed that the regularization approach could work better when images contain more low frequency contents. For images contain more high frequency contents such as testing image “Baboon”, the improvement of the regulatization approach over the POCS approach becomes less. In the regularization approach, spatial high pass filter was used as a smoothness measure to capture the local variation of the images to form the smoothness constraint. However, in POCS approach, sometimes the definition of certain types of constraint sets used to project onto is not straightforward. When smoothness constraint is used in POCS approach, spatial filtering cannot be used because it cannot be formed as a closed convex set, the projection cannot be defined and the convergence cannot be guaranteed. Instead, in the proposed POCS approach, frequency projection is used as the smoothness constraint set. However, the spatial information to capture the local variation is omitted.

It is observed that the restoration problem becomes less ill-conditioned when more *a priori* knowledge about the original image is incorporated into the restoration algorithm, better solution will be obtained when more constraints are used or when the constraints are made tighter. The intersection is made smaller, thus reducing the deviation between the elements in the set.

In image restoration problems, the more the *a priori* information available, the more the chance that the restoration output closed to the original. When the number of *a priori* knowledge is increased, regularization approach will get more complicated

and it may not be expressed in the form of a stabilizing linear objective functional, so it is not easy to formulate the problem with regularization approach. By using POCS, large number of *a priori* information can be incorporated into an image restoration algorithm easily only if the formulated constraint sets are closed convex sets. A nonexpansive mapping can then be defined to assocate with each of these constraint sets. Though the method of POCS is not really an optimization method, but it does allow for finding solutions to inverse problems that are not simply minima of some quadratic Tikhonov functional. Obviously, the intersection should be defined as tightly as possible.

For tackling more complicated problem such as restoration of halftoned color-quantized image, POCS approach can be used when the number of constraints is increased and this will be addressed in the following Chapter.

4.6 Summary

In this Chapter, a restoration algorithm for restoring color-quantized images is presented which is based on the POCS approach. This algorithm makes good use of the color palette to derive useful *a priori* information for restoration. It has been demonstrated by simulation results that the proposed restoration algorithm is capable of improving the quality of a color-quantized image. Besides, it provides a better restoration performance as compared with other existing restoration approaches such as in [Altunbasak 01, Galatsanos 91a, Hunt 84, Kim 01].

The proposed restoration algorithm requires no extra information other than the available color quantized image **Y** and the color palette to carry out the restoration. This makes it always able to provide a reasonable restoration performance irrespective of the color quantization scheme with which it works.

Chapter 5.

A POCS-based Restoration Algorithm for Restoring Halftoned Color-quantized Images

5.1 Introduction

In Chapters 3 and 4, we proposed two restoration algorithms for restoring color-quantized images. These two algorithms [Chan 05, Fung 04a] do not take account of the case when error diffusion is involved in the quantization process. Obviously, the degradation model for color quantization without error diffusion is different from that for color quantization with halftoning.

Inverse halftoning can be used to restore binary halftones to its original [Chang 01, Hein 95, Kite 00, Mese 01, Wong 95]. In printing applications, color images are decomposed into 3 or 4 color components (CMY or CMYK). Each color plane is then considered as an individual gray scale image and they are separately halftoned with a conventional binary halftoning algorithm. In such cases, a straightforward extension of a conventional inverse halftoning algorithm can do the job as one can restore each color plane with an inverse halftoning algorithm directly.

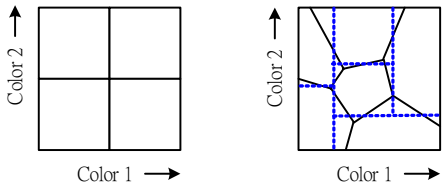
However, this straightforward approach only works for handling color prints in which the colors are composed of a few of bi-level fixed color components.

Color quantization is actually a vector quantization instead of a bi-level uniform scalar quantization as in the case of binary halftoning. It is not a combination of several independent bi-level uniform scalar quantization processes either. In general, when a low-end display unit such as a VGA monitor is involved, the palette colors are not uniformly distributed in the color space. By considering this, restoring color halftones generated for printing applications is only a special case of the problem concerned in this Chapter. In particular, it is equivalent to the case that a palette {(C,M,Y)|C,M,Y=0,1}, where 0 and 1 denote the minimum and maximum intensity values respectively, is used in color quantization.

Figure 5.1 shows how a color palette clusters a cross-section of a color space. As shown in Figure 5.1a, in the restoration of color prints generated for printing applications, one can easily define the bounds of a cluster in all dimensions and derive a simple constraint set for color restoration. However, in the general case with which we are tackling, since the involved palette can be of arbitrary size and contain arbitrary colors, clusters can be of arbitrary shape and size. Hence, it is not easy to precisely define their boundaries. Figure 5.1b shows an example of this general case. The solid lines in Figure 5.1b are the cluster boundaries which partition the cross-section in this example.

Even though an inverse halftoning algorithm is extended to handle halftones involving multi-level quantization, processing color planes separately does not work effectively. It is because, due to the implicit assumptions that this approach has, one has to approximate the clusters with rectangular column in restoration. The dotted lines in Figure 5.1b show the approximation result of the presented example. One can

see that the approximation error can be very large and hence processing color planes separately with inverse halftoning algorithms cannot provide a good restoration result in general. In other words, further extension of existing binary inverse halftoning algorithms for handling the general case is not as straightforward.



(a) (b)

Figure 5.1. How a cross-section of a color space is partitioned by a color palette: (a) in a color halftone generated for printing applications; (b) in general color quantization.

This Chapter is devoted to formulating the process of color quantization in which error diffusion is involved and developing a POCS-based restoration algorithm to restore corresponding degraded images. As mentioned in Chapter 4, a POCS algorithm is a good image restoration algorithm when the degradation model gets complicated and a linear objective function cannot be formulated. The proposed restoration algorithm is able to restore any images which are color-quantized with an arbitrary color palette and an error diffusion process.

The orgainization of this Chapter is as follows. In Section 5.2, we formulate the *a priori* information about the degradation process and the original image. In Section 5.3, we formulate our proposed restoration algorithm based on the theory of projection onto convex sets. Its performance is reported in Section 5.4. A summary is given in Section 5.5.

5.2 Formulation of *A Priori* Information

The proposed algorithm is a POCS-based algorithm which makes an estimation of the original image $\mathbf{X}$ with the observed $\mathbf{Y}$ by projecting intermediate estimates among convex constraint sets iteratively. The constraint sets used in the algorithm are formulated in this Section.

Section 2.4.1.1.2 presents the model of a system which realizes color quantization with error diffusion. In this Chapter, this model is used in the formulation of the *a priori* information of the degradation process. Suppose we have already made an estimation of $\mathbf{X}$ to get an intermediate estimate $\mathbf{X'}$ and are going to refine our estimate. The pixels of $\mathbf{X'}$ are adjusted one by one in the refinement. The order in that the pixels are adjusted follows the order in that the pixels were color-quantized. Here, we assume that the degradation process Q_{ch} is known.

Without loss of generality, consider we are now processing pixel (m,n). As we have mentioned, pixels are adjusted one by one as they were processed in color quantization. All previously adjusted pixels and pixel (m,n) of $\mathbf{X'}$ form a partial image and can be color-quantized with operation Q_{ch} (eqns.(2.16-2.18)) until pixel (m,n) is reached. For the sake of reference, this partial image is referred to as $\mathbf{I}_p$ and the set of the coordinates of the adjusted pixels and (m,n) is referred to as Ω_p hereafter. When $\mathbf{I}_p$ is color-quantized with eqns.(2.16-2.18), intermediate state vectors $\vec{\mathbf{U}}_{(k,l)}$ and error vectors $\vec{\mathbf{E}}_{(k,l)}$, where $(k,l) \in \Omega_p$, are generated. They are referred to as $\vec{\mathbf{U}}'_{(k,l)}$ and $\vec{\mathbf{E}}'_{(k,l)}$.

Obviously, since the observed image $\mathbf{Y}$ is the color-quantized output of the original image $\mathbf{X}$, the color-quantized output of our estimate of $\mathbf{X}$ should also be $\mathbf{Y}$.

In other words, we should have $Q_{ch}[\vec{\mathbf{X}}'_{(m,n)}] = \vec{\mathbf{Y}}_{(m,n)}$, where $Q_{ch}[\vec{\mathbf{I}}_{(m,n)}]$ denotes the $(m,n)^{th}$ pixel of the output when image $\mathbf{I}$ is processed with eqns.(2.16-2.18) from pixel (1,1) to pixel (m,n). Accordingly, after adjusting pixel (m,n), a convex constraint set in which the desirable output image should be can be formed as follows.

$$S_{2,(m,n)} = \{\ \mathbf{I} \mid \vec{\mathbf{I}}_{(i,j)} = \vec{\mathbf{X}}''_{(i,j)} \qquad \forall (i,j) \in \Omega_p \setminus (m,n),$$

$$\vec{\mathbf{I}}_{(i,j)} = \vec{\mathbf{X}}'_{(i,j)} \qquad \forall (i,j) \notin \Omega_p \quad \text{and}$$

$$Q_{ch}[\vec{\mathbf{I}}_{(m,n)}] = \vec{\mathbf{Y}}_{(m,n)}\} \tag{5.1}$$

where $\vec{\mathbf{X}}''_{(i,j)}$ is the adjusted $\vec{\mathbf{X}}'_{(i,j)}$ in the refinement and $\mathbf{I}$ is an image whose size is the same as $\mathbf{X}$.

Recall that $\vec{\mathbf{X}}'_{(m,n)}$ is the estimate of $\vec{\mathbf{X}}_{(m,n)}$ before adjustment. If $Q_{ch}[\vec{\mathbf{X}}'_{(m,n)}] = \vec{\mathbf{Y}}_{(m,n)}$ happens, the current state of the refined $\mathbf{X}'$ will be a member of $S_{2,(m,n)}$ and no adjustment of $\vec{\mathbf{X}}'_{(m,n)}$ will be required. However, it is possible that $Q_{ch}[\vec{\mathbf{X}}'_{(m,n)}] \neq \vec{\mathbf{Y}}_{(m,n)}$ as the $\vec{\mathbf{U}}'_{(m,n)}$ obtained with eqn.(2.16) is out of R_k as shown in Figure 5.2. Here, R_k is defined as the Voronoi region associated with palette color $\hat{\mathbf{v}}_k\ (= \vec{\mathbf{Y}}_{(m,n)})$. In other words, we have $Q_c[\vec{\mathbf{U}}'_{(m,n)}] \neq \vec{\mathbf{Y}}_{(m,n)}$ in formulation. In such a case, a projection is necessary to project $\vec{\mathbf{U}}'_{(m,n)}$ onto the boundary of R_k.

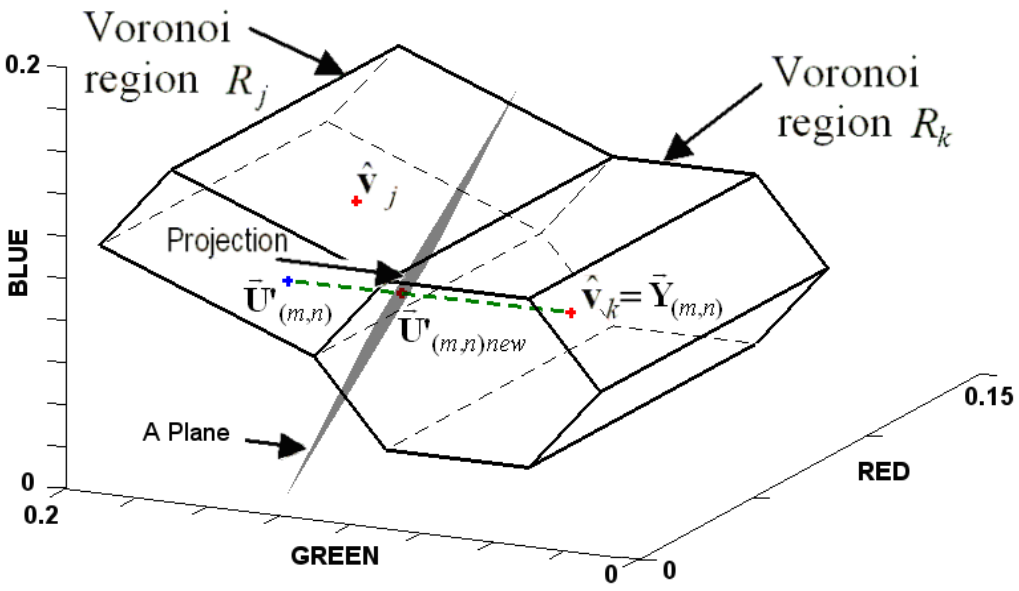


Figure 5.2. Projection of $\vec{\mathbf{U}}'_{(m,n)}$ onto R_k'

The projection is carried out as follows. Starting from $\vec{\mathbf{U}}'_{(m,n)}$, we search along the straight line connecting $\vec{\mathbf{U}}'_{(m,n)}$ and $\vec{\mathbf{Y}}_{(m,n)}$ to seek a new point $\vec{\mathbf{U}}'_{(m,n)new}$ such that $Q_c[\vec{\mathbf{U}}'_{(m,n)new}] = \vec{\mathbf{Y}}_{(m,n)}$ is satisfied. The search is conducted with estimates generated iteratively with

$$\vec{\mathbf{U}}'_{(m,n)new} = \vec{\mathbf{Y}}_{(m,n)} + \lambda^n (\vec{\mathbf{U}}'_{(m,n)} - \vec{\mathbf{Y}}_{(m,n)}) \tag{5.2}$$

where λ^n is a relaxation parameter at iteration n. The convergence of the estimate can be guaranteed as long as $0 \le \lambda < 1$. After $\vec{\mathbf{U}}'_{(m,n)new}$ is found, $\vec{\mathbf{U}}'_{(m,n)}$ is updated to be $\vec{\mathbf{U}}'_{(m,n)new}$.

Note that this point-to-point projection may not be perpendicular to the surface of R_k. However, this does not matter. As shown in Figure 5.2, one can define a plane which passes $\vec{\mathbf{U}}'_{(m,n)new}$ and is perpendicular to the line connecting $\vec{\mathbf{U}}'_{(m,n)}$ and $\vec{\mathbf{Y}}_{(m,n)}$. In formulation, the equation of the plane is given as

$$\left(\vec{\mathbf{p}} - \vec{\mathbf{U}}'_{(m,n)new}\right) \bullet \left(\vec{\mathbf{U}}'_{(m,n)} - \vec{\mathbf{Y}}_{(m,n)}\right) = 0 \qquad (5.3)$$

where $\vec{\mathbf{p}} \in R_k$ is a pixel vector in the color space and $\bullet$ denotes the dot product operator of two vectors. This plane cuts through Voronoi region R_k and splits it into two. The one containing point $\vec{\mathbf{Y}}_{(m,n)}$ forms a new constraint set R_k'. By doing so, projection from $\vec{\mathbf{U}}'_{(m,n)}$ to $\vec{\mathbf{U}}'_{(m,n)new}$ is equivalent to a projection onto a convex set R_k'.

After determining $\vec{\mathbf{U}}'_{(m,n)}$, the adjusted $\vec{\mathbf{X}}'_{(m,n)}$ can be obtained by

$$\mathbf{X}''_{(m,n)c} = \mathbf{U}'_{(m,n)c} + \sum_{(k,l)\in S} \mathbf{H}_{(k,l)c} \mathbf{E}'_{(m-k,n-l)c} \qquad \text{for } c \in \{r, g, b\} \quad (5.4)$$

The adjusted image is then a member of $S_{2,(m,n)}$. The constraint set $S_{2,(m,n)}$ and its associated projection presented above is dedicated for handling halftoned color-quantized images with POCS.

Figure 5.3 shows two examples of how the pixels of the current estimate $\mathbf{X}'$ are handled during the adjustment. Without loss of generality, in these examples, we assume gray-level images and use a two-color palette to make the examples simple enough to illustrate the approach clearly. Pixels (2,2) and (2,3) are, respectively, the pixels being handled in the first and the second examples. The shaded positions mark the pixels that were handled at the moment.

In the first example, $\vec{\mathbf{U}}'_{(2,2)}$ is determined with eqn. (5.1). Since we have $Q_c[\vec{\mathbf{U}}'_{(2,2)}] = \vec{\mathbf{Y}}_{(2,2)}$ =(0.2, 0.2, 0.2), no adjustment of $\vec{\mathbf{X}}'_{(2,2)}$ is required. $\vec{\mathbf{E}}'_{(2,2)}$ is updated with eqn. (2.18) then.

In the second example, we have $Q_c[\vec{\mathbf{U}}'_{(2,3)}] \neq \vec{\mathbf{Y}}_{(2,3)}$. Adjustment of $\vec{\mathbf{X}}'_{(2,3)}$ is hence necessary. The adjustment is carried out by iteratively adjusting $\vec{\mathbf{U}}'_{(2,3)}$ with eqn. (5.2). At iteration n=3, $\vec{\mathbf{U}}'_{(2,3)}$ is adjusted to be (0.51, 0.51, 0.51), which makes $Q_c[\vec{\mathbf{U}}'_{(2,3)}] = \vec{\mathbf{Y}}_{(2,3)}$ =(0.8, 0.8, 0.8). The corresponding $\vec{\mathbf{X}}'_{(2,3)}$ and $\vec{\mathbf{E}}'_{(2,3)}$ can then be obtained with eqns. (5.4) and (2.18) respectively.

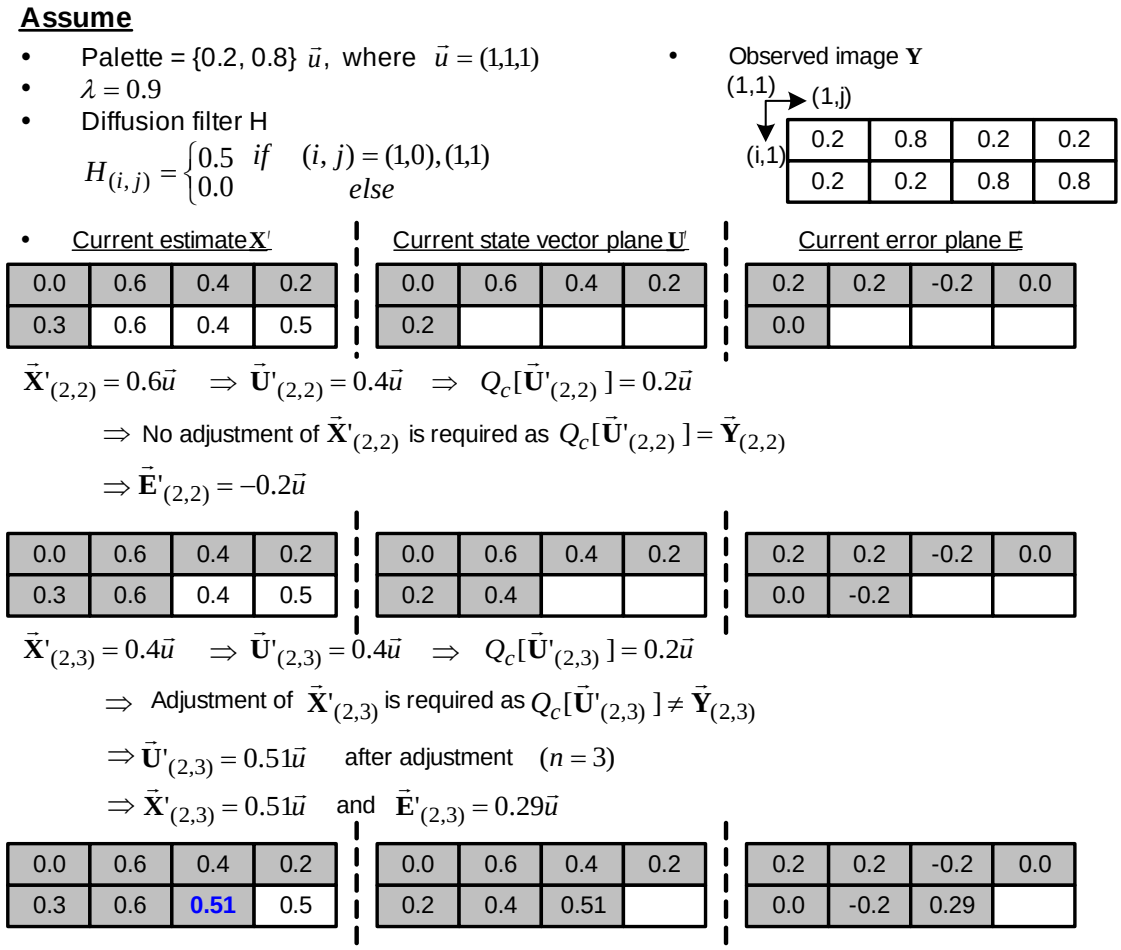


Figure 5.3. Examples of how pixels of an estimate of the original image are updated

Typical images would generally have weak high frequency components as the intensity of neighboring pixels is highly correlated. This feature can be exploited as *a priori* information in the restoration of halftoned color-quantized images. In our approach, we assume that the energy of each high frequency component of **x** is

bounded as given by $\left|[T(\mathbf{X})]_{(u,w)}\right| \le \left|[T(F(\mathbf{Y}))]_{(u,w)}\right|$ for $(u,w) \in \Omega_H$, where F and T are, respectively, a linear low-pass filtering operator and a 2D DCT operator, $[\bullet]_{(u,w)}$ denotes the $(u,w)^{th}$ element in the transform domain and Ω_H defines the set of high frequency components which should be bounded. This forms a smoothness constraint set

$$S_1 = \{\mathbf{I} \mid \left|[T(\mathbf{X})]_{(u,w)}\right| \le \left|[T(F(\mathbf{Y}))]_{(u,w)}\right| \quad \text{for } (u,w) \in \Omega_H\} \tag{5.5}$$

Another constraint set for restoring $\mathbf{X}$ is the one that confines the intensity value of a particular pixel to be valid. In formulation, we have

$$S_3 = \{\mathbf{I} \mid \vec{\mathbf{I}}_{(i,j)} \in \Gamma, \forall (i,j)\} \tag{5.6}$$

where $\Gamma = \{(r,g,b) \mid 0 \le r,g,b \le 1\}$.

5.3 Formulation of POCS Algorithm

A POCS-based iterative algorithm can be defined based on the convex constraint sets defined in the previous Section. In formulation, we have

$$\mathbf{X}^{(m+1)} = P_3 P_{2,(N,N)} \cdots P_{2,(i,j)} \cdots P_{2,(1,1)} P_1 \mathbf{X}^{(m)} \tag{5.7}$$

where $\mathbf{X}^{(m)}$ is the estimate of $\mathbf{X}$ at iteration m, P_1, $P_{2,(i,j)}$ and P_3 are the projection operators to project a given image $\mathbf{I}$ onto S_1, $S_{2,(i,j)}$ and S_3, respectively. In particular, they are defined as

$$P_1 : [T(\mathbf{I})]_{(u,w)} = [T(F(\mathbf{Y}))]_{(u,w)} \text{ if } \left|[T(\mathbf{I})]_{(u,w)}\right| > \left|[T(F(\mathbf{Y}))]_{(u,w)}\right| \text{ for } (u,w) \in \Omega_H \tag{5.8}$$

$P_{2,(i,j)}$:

$$\mathbf{I}_{(i,j)c} = \begin{cases} \mathbf{I}_{(i,j)c} & \text{if } Q_c[\vec{\mathbf{U}}'_{(i,j)}] = \vec{\mathbf{Y}}_{(i,j)} \\ \mathbf{Y}_{(i,j)c} + \beta_{(i,j)}(\mathbf{U}'_{(i,j)c} - \mathbf{Y}_{(i,j)c}) + \sum_{(k,l)\in S} \mathbf{H}_{(k,l)c}\mathbf{E}'_{(i-k,j-l)c} & \text{if } Q_c[\vec{\mathbf{U}}'_{(i,j)}] \neq \vec{\mathbf{Y}}_{(i,j)} \end{cases}$$

$$\text{for } c \in \{r, g, b\},\ \forall (i, j) \quad (5.9)$$

where $\beta_{(i,j)}$ is the corresponding λ^n for pixel (i, j) to adjust $\vec{\mathbf{U}}'_{(i,j)}$ with eqn.(5.5) such that $Q_c[\vec{\mathbf{U}}'_{(i,j)}] = \vec{\mathbf{Y}}_{(i,j)}$ can be satisfied.

$$P_3 : \mathbf{I}_{(i,j)c} = \begin{cases} 1 & \text{if } \mathbf{I}_{(i,j)c} > 1 \\ 0 & \text{if } \mathbf{I}_{(i,j)c} < 0 \end{cases} \text{ for } c \in \{r, g, b\},\ \forall (i, j) \quad (5.10)$$

Note that the pixels of the image were processed one by one from position (1,1) to (N, N) in a raster scanning order when projections $P_{2,(i,j)}$'s are performed. This order corresponds to the order that the pixels were processed during its color quantization. The examples shown in Figure 5.3 correspond to projections $P_{2,(2,2)}$ and $P_{2,(2,3)}$. The initial estimate $\mathbf{X}^{(0)}$ is set to be $F(\mathbf{Y})$ which is a filtered version of the observed image. Since all involved constraint sets are convex sets, the convergence of POCS algorithm can be guaranteed.

The physical meaning of the projections is as follows. Projection P_1 guarantees that the energy of the high frequency components of the restored image is less than the energy of the high frequency components of the low-pass filtering result of the observed image. Projection $P_{2,(i,j)}$ makes sure that the color-quantized result of the restored image is exactly the same as the observed image from the first processing pixel to the $(i, j)^{th}$ pixel. With the use of projection P_3, all pixels of the restored image are displayable.

In this Chapter, the proposed algorithm is formulated in R-G-B color space. This is based on the observation that a palette for a video color display is usually defined in R-G-B domain. With an approach similar to the one presented in this Chapter, the proposed algorithm can be reformulated in other color spaces.

5.4 Simulation and Comparative Study

In this Section, the performance of the proposed algorithm is evaluated. The details of the realization of the proposed algorithm in the evaluation are described in Section 5.4.1. Some other restoration algorithms are used for comparative study in the evaluation. They are introduced in Section 5.4.2. Finally, simulation results are given and some discussions on the results are provided.

5.4.1 Realization Details of the Proposed Algorithm

Simulation was carried out to evaluate the performance of the proposed algorithm. In our simulation, a number of *de facto* standard 24-bit full-color images including *Couple*, *Window*, *Peppers*, *Fruits*, *Lenna*, *House*, *Girl*, *Parrots*, *Pool*, *Caps*, *Baboon* and *Melon* were used. Each of them is of size 256×256. The images were color-quantized to produce $\mathbf{Y}$'s. Color palettes of different size were used in color quantization and they were generated with different palette generation algorithms.

In color quantization, halftoning was performed with error diffusion and Floyd-Steinberg diffusion filter [Floyd 76] was used. In the realization of the proposed algorithm, a 3×3 Gaussian filter was used as filter F. Ω_H and λ were, respectively, selected to be $\{(u, w) \mid (256-u)^2 + (256-w)^2 < 100000$ and $0 \le u, w < 256\}$ and 0.9. The threshold value 100000 was obtained empirically.

5.4.2 Algorithms for Comparative Study

To our best knowledge, there is no reported restoration algorithm directly proposed for restoring halftoned color-quantized images. Fung's algorithms [Chan 05, Fung 04a] were proposed for restoring color-quantized images, but they assume that there is no error diffusion involved in color quantization. Mese's algorithm [Mese 01] was originally proposed for inverse halftoning. It makes use of some training images to pre-train a linear prediction filter for filtering binary halftones. For comparison, it was modified here to handle halftoned color-quantized images. Some other restoration algorithms were also evaluated for comparison which including Galatsanos's algorithm [Galatsanos 91a], Hunt's algorithm [Hunt 84] and Altunbasak's work [Altunbasak 01]. They were realized in the way presented in Section 3.4.2.

Mese's algorithm trains a linear prediction filter with best linear estimator [Mese 01]. In the training phase, *Baboon*, *Melon*, *House*, *Lenna* and their corresponding halftoned color-quantized images were used to train the filter. Note that in Fung's algorithms [Chan 05, Fung 04a] and the proposed algorithm, no original full-color image is required to extract information and no training images are required to pre-train a linear prediction filter. Hence, additional credit should be added to the simulation results of these two algorithms.

5.4.3 Simulation Results

Tables 5.1, 5.2 and 5.3 show the restoration performance achieved with various algorithms when the involved palettes were obtained with octree algorithm [Gervautz 90]. Table 5.1 shows the SNRI performance which is defined as

$$\mathbf{SNRI} = 10\log\frac{\sum_{(i,j)}\left\|\vec{\mathbf{X}}_{(i,j)} - \vec{\mathbf{Y}}_{(i,j)}\right\|^2}{\sum_{(i,j)}\left\|\vec{\mathbf{X}}_{(i,j)} - \vec{\mathbf{X}}'_{(i,j)}\right\|^2} \tag{5.11}$$

where $\vec{\mathbf{X}}_{(i,j)}$, $\vec{\mathbf{Y}}_{(i,j)}$ and $\vec{\mathbf{X}}'_{(i,j)}$ are, respectively, the $(i, j)^{th}$ pixel of the original, the halftoned color-quantized and the restored images. One can see that the proposed algorithm is superior to any other algorithms whatever color palette size is concerned. On average, the proposed algorithm achieved, respectively, a SNRI of 7.12, 8.16, 9.88 and 9.25 dB when the involved palette is of size 256, 128, 64 and 32.

In Chapter 3, we showed that, as compared with conventional image restoration algorithms such as [Altunbasak 01, Galatsanos 91a, Hunt 84], Fung's algorithms [Chan 05, Fung 04a] worked well on restoring images that were color quantized without error diffusion. However, when error diffusion is involved in the quantization process, their restoration performance is lower than the algorithm proposed in this Chapter. This is because, similar to other conventional restoration algorithms, the degradation models assumed by Fung's algorithms [Chan 05, Fung 04a] does not take the error diffusion process into account as the proposed algorithm does.

	SNR Improvement (dB)							
	Proposed	[Chan 05] (Proposed Scheme in Chapter 3)	[Fung 04a] (Proposed Scheme in Chapter 4)	[Altunbasak 01]-IND	[Altunbasak 01]-KL	[Mese 01]	[Galatsanos 91a]	[Hunt 84]
palette size: 256								
Lenna	6.421	5.703	4.972	3.630	3.313	3.790	1.547	2.701
Baboon	0.902	3.274	2.720	2.162	2.010	2.757	0.474	1.671
Peppers	9.187	6.856	4.226	5.791	5.518	4.404	1.891	4.034
Fruits	9.424	6.663	4.745	4.507	3.976	4.320	1.521	3.834
Couple	4.083	3.755	3.466	3.890	3.509	3.583	2.028	2.765
Girl	6.168	5.170	4.671	4.672	4.472	3.866	2.409	3.149
Parrots	11.930	7.324	6.099	5.043	4.435	4.400	2.006	4.474
Pool	8.698	3.105	3.637	6.015	5.696	3.584	2.140	3.996
Caps	9.253	5.270	4.129	5.303	4.741	3.996	1.669	4.304
Window	5.110	4.909	4.218	4.005	3.237	3.890	2.120	3.352
Average	**7.118**	**5.203**	**4.288**	**4.502**	**4.091**	**3.859**	**1.781**	**3.428**
palette size: 128								
Lenna	9.933	6.230	5.638	5.101	4.725	5.201	2.852	3.934
Baboon	3.009	3.851	2.669	2.698	2.456	3.548	0.618	2.193
Peppers	8.773	6.771	2.866	5.968	5.727	5.365	2.148	4.392
Fruits	9.694	6.552	4.361	5.049	4.587	5.699	1.883	4.139
Couple	5.927	5.166	4.281	4.888	4.469	4.963	3.043	3.694
Girl	7.281	5.900	4.829	5.192	4.811	4.749	3.520	3.832
Parrots	8.835	5.788	4.595	4.421	3.657	4.729	2.187	3.853
Pool	10.581	6.426	1.800	6.812	6.307	5.004	3.229	4.283
Caps	10.329	5.459	4.080	6.450	6.000	5.484	2.133	5.239
Window	7.199	5.899	4.664	4.978	4.239	5.644	3.073	4.063
Average	**8.156**	**5.804**	**3.978**	**5.156**	**4.698**	**5.039**	**2.469**	**3.962**
palette size: 64								
Lenna	9.488	6.917	6.062	5.562	5.156	7.210	3.277	4.383
Baboon	5.099	4.226	3.160	3.839	3.462	5.490	0.851	3.294
Peppers	11.435	7.227	2.452	7.296	7.077	7.948	2.586	5.619
Fruits	11.990	6.700	4.375	6.083	5.632	7.664	3.143	5.069
Couple	5.978	4.414	2.537	4.288	3.751	5.313	3.561	3.509
Girl	8.347	6.357	4.262	5.814	5.418	6.494	4.203	4.468
Parrots	12.812	6.242	3.144	6.567	6.031	7.504	2.898	5.475
Pool	10.800	6.499	2.710	6.945	6.539	6.619	3.451	4.562
Caps	14.203	6.734	3.037	8.077	7.984	7.529	3.333	6.582
Window	8.652	6.404	4.387	5.627	5.016	7.165	4.554	4.492
Average	**9.880**	**6.172**	**3.613**	**6.010**	**5.607**	**6.894**	**3.186**	**4.745**
palette size: 32								
Lenna	12.172	7.076	3.581	6.807	6.695	10.260	4.547	5.591
Baboon	3.690	2.668	2.093	2.676	1.977	4.641	1.246	2.483
Peppers	10.721	6.011	2.330	7.257	6.977	8.943	2.765	5.790
Fruits	10.952	5.323	3.918	5.785	5.344	8.720	3.029	4.871
Couple	7.001	5.026	3.863	4.921	4.473	6.818	5.281	4.170
Girl	7.044	4.728	2.669	4.614	4.341	5.860	3.867	3.961
Parrots	11.202	5.223	3.178	6.132	5.740	7.929	2.632	5.087
Pool	7.665	4.682	2.085	5.197	5.085	5.915	2.860	4.221
Caps	15.005	6.235	2.412	9.199	9.124	9.633	3.498	7.609
Window	7.070	4.967	4.051	5.345	4.793	7.301	5.475	4.472
Average	**9.252**	**5.194**	**3.018**	**5.793**	**5.455**	**7.602**	**3.520**	**4.826**

Table 5.1. SNR Improvements of various algorithms in restoring halftoned color-quantized images

	Average of CIELAB difference ΔE								
	Observed **Y**	Proposed	[Chan 05] (Proposed Scheme in Chapter 3)	[Fung 04a] (Proposed Scheme in Chapter 4)	[Altunbasak 01]-IND	[Altunbasak 01]-KL	[Mese 01]	[Galatsanos 91a]	[Hunt 84]
palette size: 256									
Lenna	2.730	1.822	1.959	2.110	2.210	2.260	2.304	2.559	2.306
Baboon	4.625	3.458	3.617	3.802	3.922	3.956	3.911	4.511	4.071
Peppers	5.104	3.446	3.722	4.214	4.161	4.236	4.367	4.799	4.562
Fruits	3.539	2.084	2.436	2.717	2.755	2.829	2.926	3.375	2.931
Couple	6.927	5.583	5.714	5.943	6.033	6.112	6.128	6.525	6.542
Girl	6.161	4.346	4.654	4.898	5.032	5.086	5.316	5.810	5.665
Parrots	4.365	2.096	2.775	3.058	3.196	3.306	3.566	4.061	3.387
Pool	4.042	2.963	3.447	3.397	3.487	3.645	3.611	3.884	4.286
Caps	2.845	1.856	2.145	2.301	2.280	2.330	2.447	2.771	2.506
Window	2.292	1.630	1.741	1.881	1.831	1.931	1.968	2.190	1.914
Average	**4.263**	**2.928**	**3.221**	**3.432**	**3.491**	**3.569**	**3.654**	**4.049**	**3.817**
palette size: 128									
Lenna	3.760	2.167	2.472	2.633	2.888	2.988	2.790	3.337	3.074
Baboon	5.829	4.109	4.550	4.929	4.740	4.810	4.717	5.626	4.970
Peppers	5.988	4.263	4.471	5.253	4.883	4.979	4.959	5.546	5.322
Fruits	4.143	2.423	2.881	3.258	3.146	3.231	3.226	3.876	3.351
Couple	7.586	6.017	6.150	6.412	6.631	6.755	6.489	7.002	7.233
Girl	8.507	6.085	6.572	6.996	6.761	6.910	7.252	7.656	7.350
Parrots	5.091	2.570	3.338	3.713	3.568	3.740	3.937	4.689	3.875
Pool	5.818	4.175	4.451	5.302	4.750	5.010	4.952	5.319	6.062
Caps	3.820	2.325	2.809	3.092	2.872	2.947	3.086	3.666	3.229
Window	3.065	2.022	2.205	2.483	2.337	2.469	2.459	2.842	2.472
Average	**5.361**	**3.616**	**3.990**	**4.407**	**4.258**	**4.384**	**4.387**	**4.956**	**4.694**
palette size: 64									
Lenna	4.190	2.507	2.860	3.055	3.153	3.296	3.063	3.661	3.353
Baboon	7.347	4.900	5.790	6.109	5.719	5.895	5.522	7.044	5.961
Peppers	7.805	5.247	5.890	7.014	6.312	6.403	6.032	7.272	6.894
Fruits	5.882	3.429	4.318	4.759	4.420	4.538	4.340	5.301	4.723
Couple	11.070	8.752	9.395	10.060	9.314	9.552	9.517	10.130	9.699
Girl	9.024	6.383	6.928	7.595	7.164	7.336	7.314	8.031	7.889
Parrots	6.820	3.246	4.736	5.637	4.706	4.897	4.882	6.221	5.254
Pool	6.241	4.424	4.656	5.493	5.161	5.429	5.075	5.750	6.600
Caps	4.890	2.884	3.554	4.168	3.624	3.810	3.767	4.707	4.171
Window	3.691	2.427	2.664	3.056	2.796	2.956	2.891	3.254	3.010
Average	**6.696**	**4.420**	**5.079**	**5.695**	**5.237**	**5.411**	**5.240**	**6.137**	**5.756**
palette size: 32									
Lenna	6.189	3.358	4.258	4.994	4.437	4.626	3.920	5.141	4.694
Baboon	9.055	6.175	7.363	7.642	6.895	7.262	6.618	8.623	7.296
Peppers	9.075	6.312	7.247	8.246	7.406	7.547	6.879	8.526	8.006
Fruits	7.458	4.146	5.637	6.018	5.270	5.494	5.040	6.677	5.704
Couple	11.670	9.408	9.937	10.303	9.982	10.237	9.800	10.519	10.415
Girl	14.264	10.530	11.669	12.739	11.313	11.639	11.675	12.779	11.906
Parrots	8.093	4.189	6.074	6.799	5.617	5.817	5.622	7.413	6.222
Pool	9.048	6.864	7.090	8.607	7.913	7.990	7.323	9.046	9.148
Caps	6.337	3.969	4.999	5.682	4.823	5.032	4.729	6.178	5.473
Window	5.495	4.024	4.398	4.619	4.288	4.515	4.264	4.887	4.540
Average	**8.668**	**5.898**	**6.867**	**7.565**	**6.794**	**7.016**	**6.587**	**7.979**	**7.340**

Table 5.2. CIELAB difference ΔE measurement of the outputs of various algorithms in restoring halftoned color-quantized images

	% of pixels whose CIELAB ΔE <3								
	Observed **Y**	Proposed	[Chan 05] (Proposed Scheme in Chapter 3)	[Fung 04a] (Proposed Scheme in Chapter 4)	[Altunbasak 01]-IND	[Altunbasak 01]-KL	[Mese 01]	[Galatsanos 91a]	[Hunt 84]
palette size: 256									
Lenna	65.242	84.030	81.467	78.764	77.094	75.774	74.466	68.141	75.061
Baboon	32.175	48.209	46.223	42.702	39.848	39.365	40.320	32.669	39.020
Peppers	35.840	63.052	57.280	47.939	50.362	49.396	44.299	38.222	43.388
Fruits	55.803	82.997	75.800	69.995	68.971	67.807	65.100	57.716	66.568
Couple	16.960	28.906	25.909	24.478	23.773	23.155	21.799	19.757	21.315
Girl	22.423	40.277	37.058	34.949	31.149	30.699	28.796	25.214	27.176
Parrots	42.917	80.273	67.841	62.196	55.959	53.972	51.640	45.317	51.947
Pool	59.547	71.956	64.302	68.803	62.608	59.546	64.955	61.588	45.754
Caps	71.916	84.808	79.982	79.036	78.549	77.803	76.718	72.197	74.263
Window	76.498	86.749	84.554	82.753	84.401	82.553	81.105	77.873	83.594
Average	**47.932**	**67.126**	**62.045**	**59.162**	**57.271**	**56.007**	**54.920**	**49.869**	**52.809**
palette size: 128									
Lenna	44.666	77.856	71.841	68.423	60.689	58.819	63.997	50.713	56.793
Baboon	24.685	40.737	36.627	32.131	32.349	31.624	33.487	25.275	30.542
Peppers	24.790	50.957	44.939	32.763	39.548	38.362	35.255	27.105	33.269
Fruits	49.148	77.227	69.200	62.616	63.028	62.056	60.991	52.089	60.266
Couple	12.017	24.491	21.484	19.580	18.918	18.319	17.842	15.876	16.509
Girl	15.514	32.123	27.808	24.787	23.827	23.028	21.570	19.820	20.349
Parrots	31.705	70.021	55.780	49.901	47.922	45.493	43.604	34.984	42.433
Pool	49.875	54.128	55.388	54.216	46.271	43.204	53.833	54.480	27.611
Caps	60.262	77.037	71.436	69.666	68.306	67.679	67.679	60.960	62.082
Window	67.217	81.619	78.719	75.182	77.025	75.275	74.715	70.641	74.770
Average	**37.988**	**58.620**	**53.322**	**48.927**	**47.788**	**46.386**	**47.297**	**41.194**	**42.462**
palette size: 64									
Lenna	38.791	71.770	64.101	59.491	55.985	53.632	57.278	45.557	51.787
Baboon	15.416	32.446	25.221	22.269	22.051	21.211	25.084	16.180	20.406
Peppers	15.533	40.479	31.726	20.789	25.822	25.307	27.052	16.440	20.334
Fruits	39.839	67.514	55.742	50.711	50.221	49.542	52.138	43.216	46.478
Couple	7.062	17.006	14.139	10.320	12.891	12.050	12.268	10.338	11.456
Girl	13.445	29.186	25.209	21.196	20.738	20.056	21.123	18.063	17.209
Parrots	18.955	58.690	39.369	31.004	30.760	29.744	32.440	22.157	24.637
Pool	47.453	51.090	56.319	54.215	40.054	37.802	49.390	51.912	23.117
Caps	49.161	68.636	63.083	57.643	53.322	52.113	57.100	48.041	44.424
Window	60.097	76.592	73.067	68.224	69.693	67.995	69.525	66.052	65.103
Average	**30.575**	**51.341**	**44.798**	**39.586**	**38.154**	**36.945**	**40.340**	**33.796**	**32.495**
palette size: 32									
Lenna	20.183	56.981	41.634	30.269	34.897	33.077	41.145	26.172	31.113
Baboon	5.714	17.087	11.856	9.892	11.177	9.784	12.691	6.772	9.564
Peppers	11.470	29.569	22.385	15.674	19.302	18.605	22.220	11.731	14.442
Fruits	16.261	53.395	35.657	29.390	33.232	31.328	36.087	19.965	28.351
Couple	4.994	14.166	10.788	8.391	9.828	9.160	10.436	8.522	8.608
Girl	4.733	19.981	12.311	8.499	11.903	11.256	11.493	7.695	9.508
Parrots	15.419	47.014	30.048	24.864	24.966	24.109	26.218	17.999	19.821
Pool	42.888	35.233	43.378	40.375	23.879	23.216	43.753	20.995	13.985
Caps	43.073	55.789	49.933	48.920	41.521	40.753	49.390	41.200	32.968
Window	40.736	56.862	50.359	49.876	49.411	47.699	52.416	45.419	44.013
Average	**20.547**	**38.608**	**30.835**	**26.615**	**26.012**	**24.899**	**30.585**	**20.647**	**21.237**

Table 5.3. Percentage of pixels whose CIELAB difference is not detectable ($\Delta E < 3$) after restoration

Tables 5.2 and 5.3 show the performance of the evaluated algorithms in terms of the CIELAB color difference (ΔE) metric. Table 5.2 shows the average of the ΔE values of all pixels in a restoration output and Table 5.3 shows the percentage of pixels whose color error is visually undetectable in a restoration output. Again, one can see that the proposed algorithm is superior to the others.

Figures 5.4 and 5.5 show the restoration results of different algorithms for visual evaluation. Figures 5.4b and 5.5b were, respectively, obtained with a palette of size 64 and a palette of size 128. Though error diffusion can remove false contour and color shift to a certain extent, these artefacts still appear in the color-quantized outputs as the palette size is too small. Besides, pepper noise was introduced by error diffusion. After restoration, one can see that the proposed algorithm can remove most of the artefacts while the others are comparatively inferior in this aspect.

5.4.4 Robustness Study

In color quantization, an image may be quantized with any given palette. In this section, we explore if the performance of the proposed algorithm is sensitive to the palette used in the color quantization process.

Table 5.4 shows the average performance of various algorithms when the palettes were generated with median-cut algorithm [Heckbert 82]. The same set of standard testing images used to obtain the results presented in Tables 5.1 to 5.3 were used for evaluation. The figures in the Tables are the average values obtained with the restoration results of the color-quantized images. One can see that the proposed algorithm is still superior to the others. This simulation result shows that the proposed algorithm works with different palettes generated with different palette generation algorithms.

	Palette size	Observed (**Y**)	Restored (**X'**)							
			Proposed	[Chan 05] (Proposed Scheme in Chapter 3)	[Fung 04a] (Proposed Scheme in Chapter 4)	[Altunbasak 01]-IND	[Altunbasak 01]-KL	[Mese 01]	[Galatsanos 91a]	[Hunt 84]
a		Average of (SNR Improvement (dB))								
	256	-	7.258	5.661	4.543	4.460	4.195	3.946	1.494	3.377
	128	-	8.426	5.996	4.687	5.169	4.853	5.293	2.232	4.046
	64	-	9.847	6.218	4.512	5.788	5.448	6.628	3.127	4.636
	32	-	10.058	5.527	3.904	5.883	5.635	7.705	3.951	4.867
b		Average of (Average of CIELAB difference ΔE)								
	256	4.876	2.615	3.083	3.406	3.627	3.692	3.945	4.496	3.903
	128	5.821	3.099	3.743	4.184	4.237	4.346	4.404	5.224	4.647
	64	7.285	3.730	4.708	5.418	5.174	5.349	5.169	6.287	5.751
	32	8.951	4.979	6.399	7.031	6.473	6.718	6.268	7.698	7.115
c		Average of (% of pixels whose CIELAB $\Delta E < 3$)								
	256	42.044	70.220	63.48	58.664	56.007	55.048	51.430	45.028	51.860
	128	32.286	63.899	55.13	48.864	47.250	45.939	45.020	36.983	42.160
	64	24.169	56.583	44.88	36.989	37.012	35.571	37.685	29.837	31.771
	32	15.247	44.498	31.61	25.659	26.329	24.920	28.894	21.796	21.540

Table 5.4. Average performance of various algorithms in restoring halftoned color-quantized images in various aspects (Palette was generated with median-cut algorithm [Heckbert 82])

(a) Original (b) Color-quantized (with error diffusion)

(c) [Galatsanos 91a] (d) [Hunt 84]

(e) [Altunbasak 01]-IND (f) [Altunbasak 01]-KL

Figure 5.4. Restoration results of color-quantized “Caps” (palette is generated by median-cut algorithm [Heckbert 82] with size 64)

(g) [Chan 05] (Proposed Scheme in Chapter 3)

(h) [Fung 04a] (Proposed Scheme in Chapter 4)

(i) [Mese 01]

(j) Proposed

Figure 5.4(continue). Restoration results of color-quantized “Caps” (palette is generated by median-cut algorithm [Heckbert 82] with size 64)

(a) Original (b) Halftoned Color-quantized

(c) [Galatsanos 91a] (d) [Hunt 84]

(e) [Altunbasak 01]-IND (f) [Altunbasak 01]-KL

Figure 5.5. Restoration results of color-quantized “Parrots” (palette is generated by Octree algorithm [Gervautz 90] with size 128)

(g) [Chan 05] (Proposed Scheme in Chapter 3)

(h) [Fung 04a] (Proposed Scheme in Chapter 4)

(i) [Mese 01]

(j) Proposed

Figure 5.5(continue). Restoration results of color-quantized “Parrots” (palette is generated by Octree algorithm [Gervautz 90] with size 128)

	Palette size	Observed ($\mathbf{Y}$)	Restored ($\mathbf{X}'$)		
			Floyd- Steinberg [Floyd 76]	Jarvis-Judice- Ninke [Jarvis 76]	Stucki [Stucki 81]
a		Average of (SNR Improvement (dB))			
	256	-	7.258	6.621	6.750
	128	-	8.426	7.716	7.850
	64	-	9.847	9.022	9.159
	32	-	10.058	9.142	9.269
b		Average of (Average of CIELAB difference ΔE)			
	256	4.876	2.615	2.703	2.686
	128	5.821	3.099	3.206	3.185
	64	7.285	3.730	3.870	3.843
	32	8.951	4.979	5.154	5.125
c		Average of (% of pixels whose CIELAB $\Delta E < 3$)			
	256	42.044	70.220	68.908	69.187
	128	32.286	63.899	62.304	62.570
	64	24.169	56.583	54.371	54.729
	32	15.247	44.498	42.295	42.637

Observed $\mathbf{Y}$ is color-quantized with a palette generated with median-cut algorithm and a Floyd- Steinberg filter.

Table 5.5. Average performance of the proposed algorithm when different error diffusion filters are assumed in restoration (Palette was generated with median-cut algorithm [Heckbert 82])

In the proposed algorithm, we assumed that the error diffusion filter used in color quantization is known in restoration. Sometimes this information may not be available and one has to estimate it from the observed images. The robustness of the proposed algorithm was studied. Table 5.5 shows the average restoration performance of the proposed algorithm when different error diffusion filters were exploited in the restoration of a color-quantized image which was obtained with a Floyd-Steinberg filter [Floyd 76]. Note that the three error diffusion filters [Floyd 76], [Jarvis 76] and [Stucki 81] involved in the study are all popular filters used in practice. It was found that a wrong assumption resulted in less than 1 dB drop in SNRI on average. Even so, as compared with the figures reported in Table 5.4, the proposed algorithm is still superior to the others.

5.5 Summary

By far very little research has been carried out to address the restoration of halftoned color-quantized images. Although there are many restoration algorithms for restoring blurred and noisy color images and inverse halftoning, they are not adequate to handle halftoned color-quantized images. The noise introduced by color quantization with error diffusion is basically signal dependent and is not white, which violates the assumptions adopted in most current multichannel restoration algorithms. Though Fung's algorithms [Chan 05, Fung 04a] was proposed to restore color-quantized images, they do not take error diffusion into account and hence cannot handle the case well either.

In this Chapter, we proposed a dedicated restoration algorithm for restoring halftoned color-quantized images. This algorithm makes use of the available color palette and the *a priori* knowledge about the halftoning process to derive useful information for restoration. Unlike some other conventional restoration algorithms, it requires no estimation of parameters describing the nature of the original full-color image and no training image to pre-train prediction filters. Simulation results demonstrated that the proposed algorithm can achieve a remarkable improvement in the quality of a halftoned color-quantized image in terms of both SNR and CIELAB color difference ΔE metric irrespective of the size and the production of the color palette exploited in the color quantization.

Chapter 6.

A Simulated Annealing Restoration Algorithm for Restoring Halftoned Color-quantized Images

6.1 Introduction

Color quantization with error diffusion is a nonlinear process. Consequently, conventional gradient-oriented optimization algorithms may not be the best tools to solve the problem. Simulated annealing (SA) [Kirkpatrick 83] is an adaptive searching algorithm that works very well on discrete optimization problem. The basic idea of SA is to simulate an annealing process in a system. As compared with some other conventional methods, it accepts solution with deteriorated cost to a limited extent. This feature gives the heuristic the capability to escape from the local minimum. This Chapter is devoted to develop a simulated annealing restoration algorithm to a halftoned color-quantized image.

The organization of this Chapter is as follows. Section 6.2 presents the derivation of the proposed SA-based restoration algorithm. In Section 6.3, simulation

results for comparative study are provided to evaluate the performance of the proposed algorithm. Finally, a summary is given in Section 6.4.

6.2 Proposed Simulated Annealing Restoration Algorithm

The model discussed in Section 2.4.1.1.2 is used again in this Chapter. Let **S** be the output image of the restoration. Obviously, when the restored image **S** is color-quantized with error diffusion, the output should be equal to **Y**. In formulation, we should have

$$\mathbf{Y} = Q_{ch}[\mathbf{S}] \tag{6.1}$$

where $Q_{ch}[\bullet]$ denotes the operator which performs color quantization with error diffusion as shown in Figure 2.5. Based on this criterion, the cost function of a restored image is defined as

$$E = \Sigma\left[\mathbf{Y} - Q_{ch}[\mathbf{S}]\right] \tag{6.2}$$

where $\Sigma[\mathbf{I}]$ denotes the total number of nonzero elements in image **I**.

In our approach, **S** is searched with simulated annealing to minimize cost function *E*. Without loss of generality, simulated annealing is a double-loop iterative algorithm that simulates an annealing process at a given temperature T. During simulated annealing, temperature T is reduced in a controlled manner as given by

$$T_{k+1} = \alpha T_k \tag{6.3}$$

where T_k is the temperature at stage k and α is a constant used to achieve cooling. At a particular temperature T_k, the amount of time spent in annealing is gradually adjusted by

$$M_{k+1} = \beta M_k \tag{6.4}$$

where M_k is actually the number of iterations performed at temperature T_k and β is a constant used to do the adjustment. The algorithm is terminated at temperature T_m when $\sum_{k=0}^{m} M_k$ is larger than a predefined threshold $t_{\max}$. Here, we assume that the simulated annealing process starts at its initial stage $k = 0$.

Let $\mathbf{S}_{cur}$ be the current estimate of the restored image at a particular iteration at temperature T_k and E_{cur} be its corresponding cost. The new estimate of the restored image is made with $\mathbf{S}_{cur}$ by

$$\mathbf{S}_{new} = \mathbf{S}_{cur} + \gamma\left(\mathbf{Y} - Q_{ch}[\mathbf{S}_{cur}]\right) \tag{6.5}$$

where γ is a controlling parameter used to control the amount of perturbation applied to the $\mathbf{S}_{cur}$. It is based on the idea that, if any pixel of $Q_{ch}[\mathbf{S}_{cur}]$ does not equal to that of $\mathbf{Y}$ at a particular pixel location, the corresponding pixel of $\mathbf{S}_{cur}$ should be adjusted.

The cost of $\mathbf{S}_{new}$, say, E_{new}, is then evaluated with eqn. (6.2). When $E_{new} < E_{cur}$ happens, $\mathbf{S}_{cur}$ is updated to be $\mathbf{S}_{new}$. Furthermore, if $E_{new} < E_{best}$ happens, where E_{best} is the cost of the best estimate so far, say $\mathbf{S}_{best}$, then $\mathbf{S}_{best}$ will be replaced by $\mathbf{S}_{new}$. In formulation, we have

$$\mathbf{S}_{cur} = \begin{cases} \mathbf{S}_{new} & \text{if } E_{new} < E_{cur} \\ \mathbf{S}_{cur} & \text{otherwise} \end{cases} \tag{6.6}$$

and

$$\mathbf{S}_{best} = \begin{cases} \mathbf{S}_{new} & \text{if } E_{new} < E_{best} \\ \mathbf{S}_{best} & \text{otherwise} \end{cases} \tag{6.7}$$

When $E_{new} \geq E_{cur}$ happens, $\mathbf{S}_{cur}$ will be updated to be $\mathbf{S}_{new}$ only if $r < e^{(E_{cur} - E_{new})/K_B T}$, where *r* is a randomly generated value which is uniformly distributed between 0 and 1,

T denotes the current temperature and K_B is the Boltzmann constant. This criterion for accepting the new solution is known as the *Metropolis criterion*.

At the beginning, temperature T is high. This permits many uphill moves and provides chances for the solution to leave a local minimum. As temperature T is reduced gradually, fewer and fewer uphill moves are permitted and only downhill moves are allowed eventually.

6.3 Performance Evaluation and Comparative Study

The performance of the proposed algorithm was evaluated and the evaluation results are provided in this Section. The details of the realization of the proposed algorithm are first described in Section 6.3.1. Some other restoration algorithms were also evaluated and a study of the impact of the relevant parameters to the restoration performance are given. Simulation results and a brief discussion on the simulation results is given in Section 6.3.3.

6.3.1 Realization Details of the Proposed Algorithm

Simulation has been carried out to evaluate the performance of the proposed algorithm. In our simulation, a number of *de facto* standard 24-bit full-color images of size 256×256 each were used. These testing images were color-quantized to produce $\mathbf{Y}$'s. Color palettes of different size were used for quantization and they were generated with different palette generation algorithms such as the median-cut algorithm [Heckbert 82] and octree algorithm [Gervautz 90]. In color quantization, error diffusion was performed with error diffusion and the Floyd-Steinberg diffusion filter [Floyd 76] was used. The proposed restoration algorithm was used to restore the halftoned color-quantized images ($\mathbf{Y}$'s).

In the realization of the proposed algorithm, both the initial estimate of $\mathbf{S}$ and $\mathbf{S}_{best}$ were initialized to be the filtered output of the observed image $\mathbf{Y}$. Specifically, a 3×3 Gaussian filter was used to generate the initial estimate of $\mathbf{S}$ and $\mathbf{S}_{best}$. Initial temperature T_0 was selected to be $|E_{\mathbf{S}_0} - E_{\mathbf{S'}_0}|/(K_B \log_e(0.95))$, where $E_{\mathbf{S}_0}$ and $E_{\mathbf{S'}_0}$ were, respectively, the cost of $\mathbf{S}_0$ and $\mathbf{S'}_0$. Here, $\mathbf{S}_0$ denotes the initial estimate of $\mathbf{S}$ and $\mathbf{S'}_0$ is the first estimate obtained with eqn. (6.8) based on $\mathbf{S}_0$. In formulation, we have $\mathbf{S'}_0 = \mathbf{S}_0 + \gamma(\mathbf{Y} - Q_{ch}[\mathbf{S}_0])$. This allows reasonable amount of uphill move at the beginning. Parameter α was selected to be 0.9, the middle value of the selection range suggested in [Kirkpatrick 83]. To simplify the algorithm, we selected β to be 1 such that we had $M_k = M$ for all $k \geq 0$.

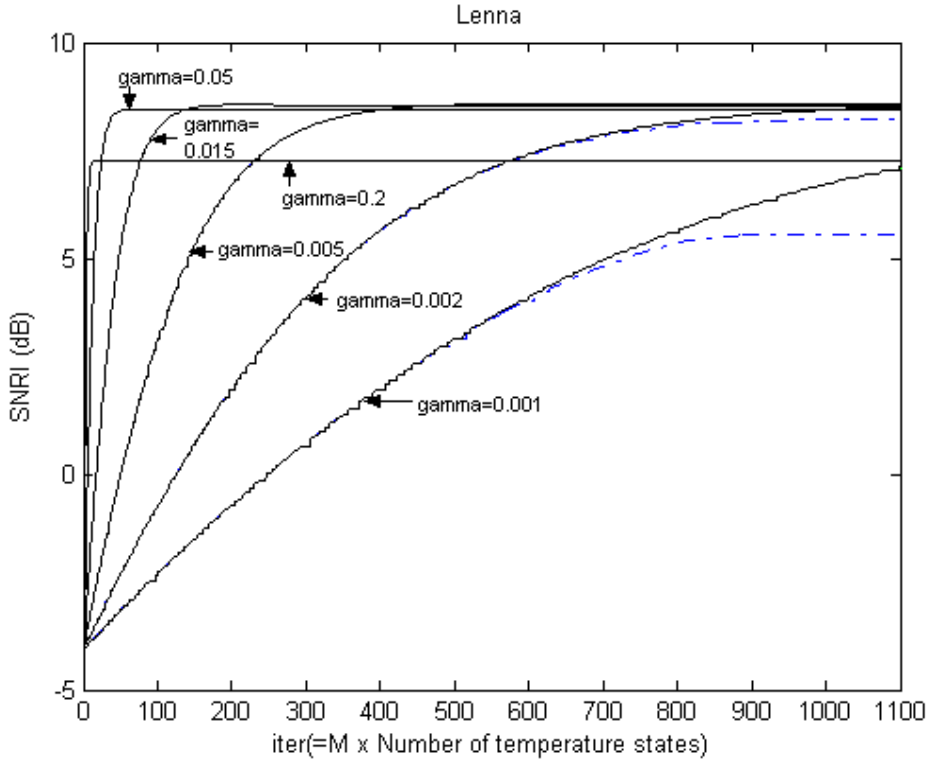
Different combinations of M and γ were evaluated to study their impact to the restoration performance. Figure 6.1 shows some typical results of the study. Specifically, it shows the Signal-to-Noise Ratio Improvement (SNRI) that was achieved when the proposed algorithm was used to restore color-quantized "Lenna", "Peppers" and "Couple". The color quantization was realized with a 128-color palette generated with median-cut algorithm [Heckbert 82]. Here, SNRI is defined as

$$SNRI = 10\log\frac{\sum_{(i,j)}\left\|\vec{\mathbf{X}}_{(i,j)} - \vec{\mathbf{Y}}_{(i,j)}\right\|^2}{\sum_{(i,j)}\left\|\vec{\mathbf{X}}_{(i,j)} - \vec{\mathbf{S}}_{best(i,j)}\right\|^2} \tag{6.8}$$

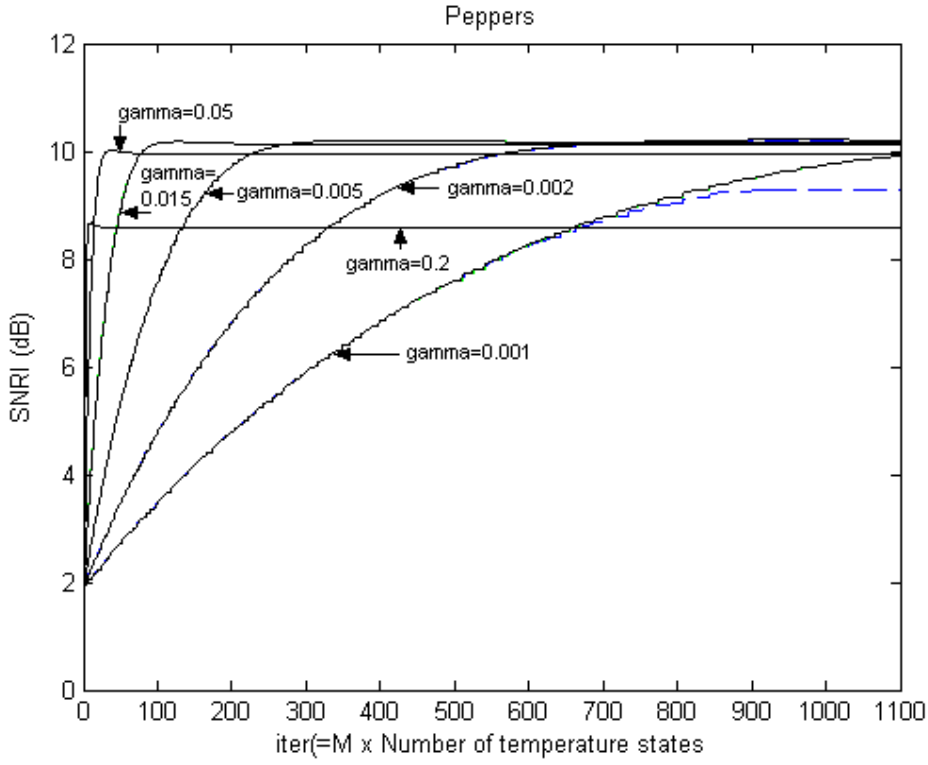
where $\vec{\mathbf{X}}_{(i,j)}$, $\vec{\mathbf{Y}}_{(i,j)}$ and $\vec{\mathbf{S}}_{best(i,j)}$ are, respectively, the $(i, j)^{th}$ pixels of the original, the halftoned color-quantized and the restored images. In Figure 6.1, the broken lines correspond to the cases of $M = 10$ while the solid lines correspond to the cases of $M = 50$ and 100. Note the lines for $M = 50$ and $M = 100$ overlap with each other for any particular γ. For any γ the value of which is larger than 0.002, it is difficult

to discriminate the line for $M = 10$ and the lines for $M = 50$ and 100. It was also found that, when $M \geq 50$, the smaller the value of γ, the more iterations were required for the estimate to converge but the better the converged output could be achieved in terms of SNRI.

Based on the evaluation results of the study, it can be found that, the combination of $M = 50$ and $\gamma = 0.002$ is a reasonable choice for restoring halftoned color-quantized images. With this combination, the best SNRI performance can be achieved in around 1000 iterations. It is good enough for the estimates of the iterative algorithm to converge to its restoration result. If complexity is a critical concern, one may select a combination of $M = 50$ and $\gamma = 0.05$, which can achieve a SNRI close to the one obtained with the previous selection in less than 100 iterations. In this Chapter, the presented results of the proposed algorithm were obtained with $\alpha = 0.9$, $\beta = 1$, $T_0 = | E_{\mathbf{S}_0} - E_{\mathbf{S}'_0} | / (K_B \log_e(0.95))$, $t_{\max} = 1000$, $M = 50$ and $\gamma = 0.002$.

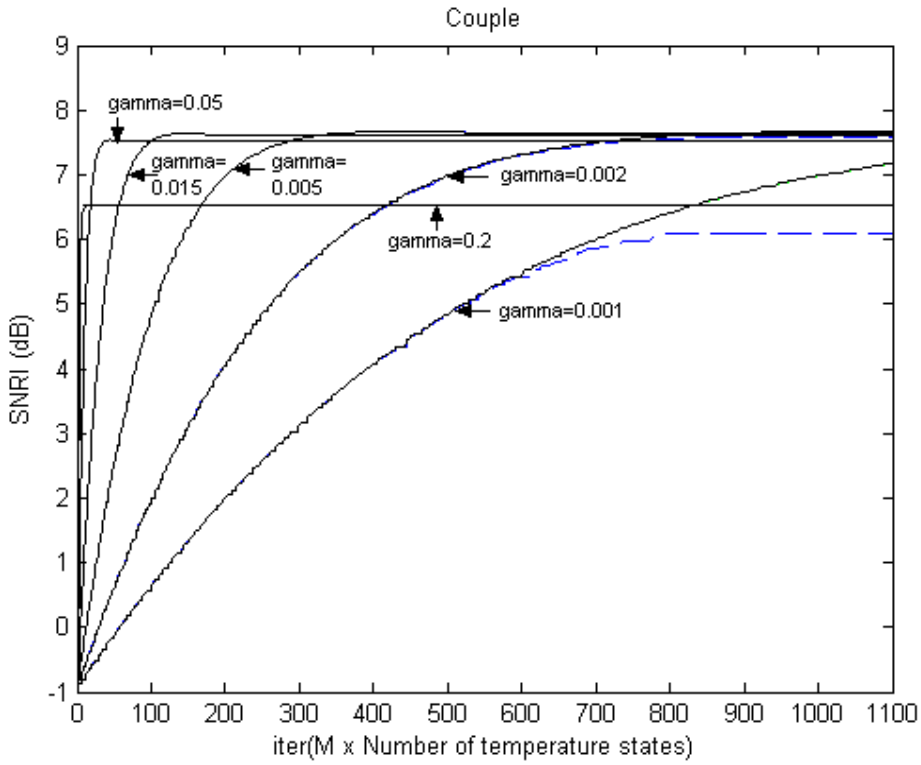


(a)



(b)

Figure 6.1: SNR Improvements achieved with various combinations of γ and M in restoring halftoned color-quantized (a) “Lenna “, (b) “Peppers” and (c) “Couple”



(c)

Figure 6.1(continue). SNR Improvements achieved with various combinations of γ and M in restoring halftoned color-quantized (a) "Lenna ", (b) "Peppers" and (c) "Couple"

6.3.2 Simulation Results

Some other restoration algorithms were also evaluated for comparison. They are, respectively, Galatsanos's algorithm [Galatsanos 91a], Hunt's algorithm [Hunt 84], Altunbasak's work [Altunbasak 01] and Mese's algorithm [Mese 01]. Again, they were realized in the way presented in Section 3.4.2. Futuremore, restoration algorithm presented in Chapter 3, 4 and 5 are also presented for comparison.

	SNR Improvement (dB)								
	Proposed	[Fung 04b] (Proposed Scheme in Chapter 5)	[Chan 05] (Proposed Scheme in Chapter 3)	[Fung 04a] (Proposed Scheme in Chapter 4)	[Altunbasak 01]-IND	[Altunbasak 01]-KL	[Mese 01]	[Galatsanos 91]	[Hunt 84]
Palette size: 256									
Lenna	7.2292	7.1522	6.1891	5.4038	4.0755	3.9394	4.1646	1.6259	3.1495
Baboon	2.7570	1.4065	3.5355	2.6836	1.9198	1.7546	2.5360	0.5088	1.2670
Boat	3.9082	3.5373	3.4822	3.4746	3.2372	3.1287	3.4953	0.8375	2.0733
Peppers	8.5542	8.4536	6.0067	5.0088	5.3847	5.2902	4.1263	1.1409	3.8154
Fruits	7.9054	7.9554	5.6862	4.4207	4.0514	3.6858	3.9884	1.4248	3.0521
Couple	5.9121	5.2629	5.0616	4.6150	4.0953	3.9070	3.7473	2.1920	2.8439
Girl	6.8524	5.9905	5.1438	4.4647	4.4029	4.2232	3.8023	2.0865	2.8951
Average	**6.1598**	**5.6798**	**5.0150**	**4.2959**	**3.8810**	**3.7041**	**3.6943**	**1.4023**	**2.7280**
Palette size: 128									
Lenna	8.5110	8.9303	6.5340	5.8254	4.9392	4.5433	5.7123	2.4357	3.8346
Baboon	4.3958	2.9715	4.2906	3.7212	2.6872	2.4976	3.6064	0.8323	1.9535
Boat	5.6739	5.0513	4.2341	3.7710	4.0782	3.8174	4.9277	1.4246	2.7164
Peppers	10.206	10.134	6.9744	6.1409	6.2363	6.1193	5.6587	1.7904	4.5946
Fruits	10.136	9.9116	5.8511	3.8925	5.1414	4.6937	5.6356	2.2694	3.8296
Couple	7.6694	7.2605	5.9278	5.2696	4.9311	4.5735	5.3854	3.4066	3.8429
Girl	8.6239	7.9857	5.7216	4.4373	5.3435	5.0748	5.1745	3.2783	3.8128
Average	**7.8880**	**7.4636**	**5.6477**	**4.7226**	**4.7653**	**4.4742**	**5.1572**	**2.2053**	**3.5121**
Palette size: 64									
Lenna	9.8515	10.342	6.6298	6.0192	5.6204	5.2254	7.0742	3.2720	4.5439
Baboon	5.6862	4.3182	4.4730	4.2475	3.4326	3.1365	4.7706	0.9225	2.9029
Boat	6.6932	6.5459	4.7235	4.4319	4.8928	4.5457	6.0329	2.3850	3.4323
Peppers	11.285	11.388	7.1605	5.6258	6.7896	6.6543	7.0382	2.5847	5.2137
Fruits	11.017	9.9350	5.5337	3.9601	5.6140	5.1302	6.8376	2.9100	4.3907
Couple	9.8891	9.9129	6.9779	4.7026	5.8604	5.4856	7.0296	4.9702	4.5849
Girl	10.079	9.8046	6.8762	4.9431	5.9710	5.7091	6.4191	4.1484	4.5656
Average	**9.2144**	**8.8924**	**6.0535**	**4.8472**	**5.4544**	**5.1267**	**6.4575**	**3.0275**	**4.2334**
Palette size: 32									
Lenna	11.741	11.732	6.6416	5.8347	5.9938	5.7719	8.9195	5.1395	4.9865
Baboon	5.8750	5.0720	4.1535	3.5332	3.9324	3.5293	5.5499	1.2983	3.1871
Boat	8.3006	8.3584	4.8256	4.4603	5.7094	5.3401	7.6022	3.3472	4.0502
Peppers	10.985	10.911	5.8319	4.7372	6.6413	6.5218	7.9197	3.4330	5.2593
Fruits	11.942	10.778	4.9044	3.0951	5.9704	5.6623	8.4185	4.2259	4.9070
Couple	9.6600	9.2897	5.5657	4.8138	5.5600	5.2710	8.1417	6.2369	4.7835
Girl	10.545	10.355	6.4679	4.7718	6.2691	6.0062	7.3840	5.6861	5.0606
Average	**9.8641**	**9.4994**	**5.4844**	**4.4637**	**5.7252**	**5.4432**	**7.7051**	**4.1953**	**4.6049**

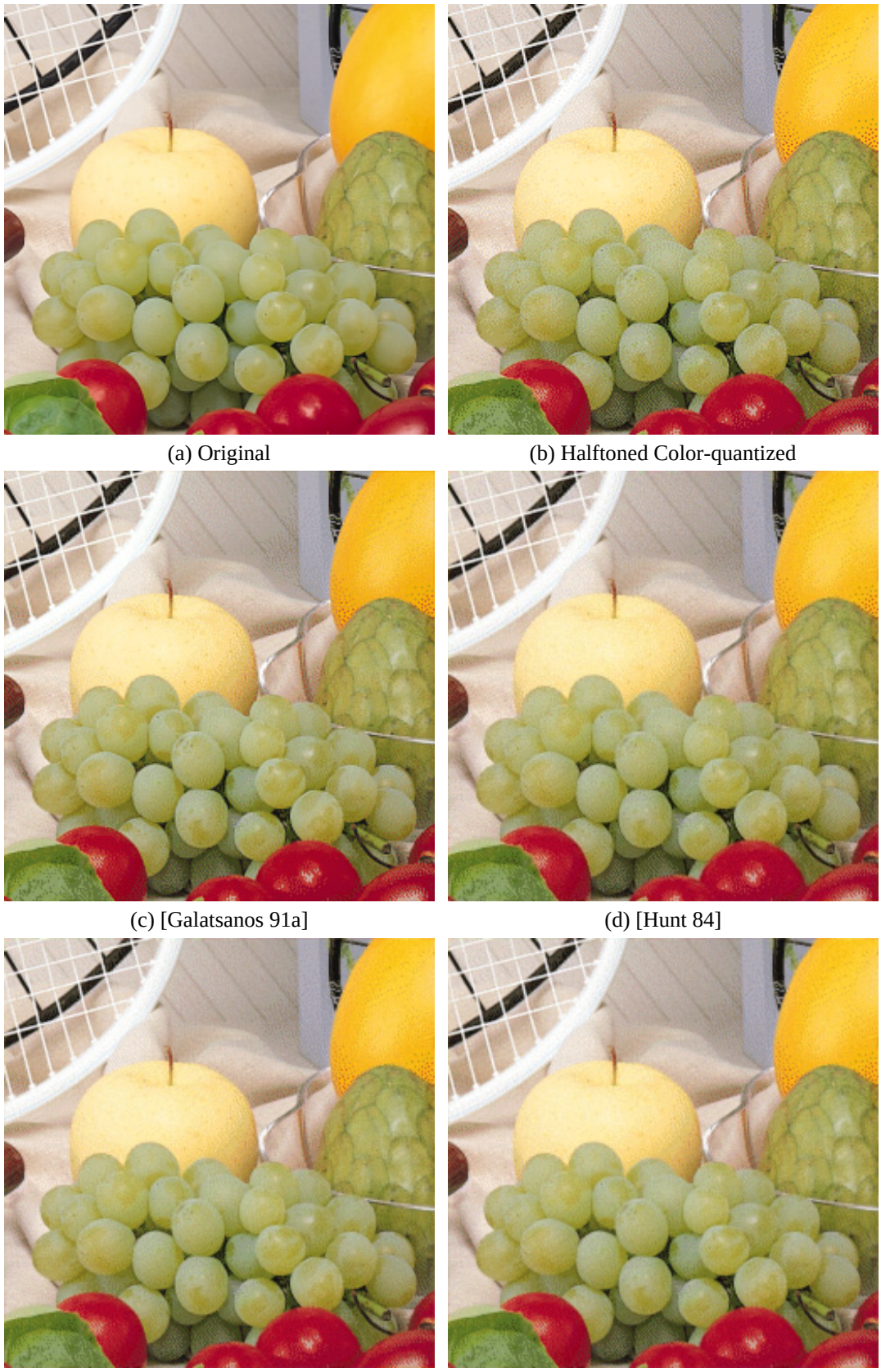
Table 6.1. SNR Improvements of various algorithms in restoring halftoned color-quantized images with the palette generated by median-cut algorithm [Heckbert 82].

	Average of CIELAB difference ΔE									
	Observed **Y**	Proposed	[Fung 04b] (Proposed Scheme in Chapter 5)	[Chan 05] (Proposed Scheme in Chapter 3)	[Fung 04a] (Proposed Scheme in Chapter 4)	[Altunbasak 01]-IND	[Altunbasak 01]-KL	[Mese 01]	[Galatsanos 91a]	[Hunt 84]
Palette size: 256										
Lenna	3.2825	2.0043	1.881	2.1557	2.2942	2.4992	2.5408	2.6592	2.9921	2.6074
Baboon	4.5157	3.4646	3.3987	3.5192	3.7660	3.8570	3.8893	3.8170	4.4015	4.0216
Boat	4.3246	2.9612	2.9305	3.2147	3.3129	3.4187	3.4453	3.5574	4.1808	3.6966
Peppers	4.8757	3.0295	2.9401	3.3548	3.7024	3.9070	3.9413	4.0404	4.7539	4.2931
Fruits	3.4039	2.1027	2.0106	2.3463	2.6086	2.6524	2.7085	2.8116	3.2263	2.8190
Couple	9.3832	4.7604	4.6214	5.3117	5.5639	6.8130	6.9349	7.5452	7.7575	7.0757
Girl	7.1182	3.9498	3.9228	4.4710	4.8083	5.1771	5.2650	5.6876	6.6614	5.7108
Average	**5.2720**	**3.1818**	**3.1007**	**3.4819**	**3.7223**	**4.0463**	**4.1036**	**4.3026**	**4.8534**	**4.3177**
Palette size: 128										
Lenna	3.9581	2.3189	2.1498	2.5545	2.7375	2.8777	3.0124	2.9825	3.4767	3.0619
Baboon	5.7313	3.9406	3.8648	4.2715	4.3301	4.5950	4.6499	4.4859	5.4871	4.7857
Boat	5.1483	3.3552	3.3321	3.9222	4.0071	3.9664	4.0272	3.9847	4.8933	4.3653
Peppers	6.2261	3.7152	3.6134	4.2171	4.5155	4.7408	4.8001	4.8953	5.8101	5.2532
Fruits	4.4127	2.5316	2.4532	3.0595	3.4507	3.2907	3.3936	3.3813	4.0053	3.5722
Couple	10.005	5.0681	5.012	5.7066	6.3255	7.3146	7.5623	7.4200	7.8051	7.7401
Girl	8.4566	4.5556	4.5524	5.4591	6.2587	6.0465	6.2002	6.3329	7.5471	6.8269
Average	**6.2769**	**3.6407**	**3.5682**	**4.1707**	**4.5179**	**4.6902**	**4.8065**	**4.7832**	**5.5750**	**5.0865**
Palette size: 64										
Lenna	4.6383	2.7927	2.5514	3.129	3.3420	3.3728	3.5928	3.3759	4.0210	3.6044
Baboon	7.2079	4.5986	4.4502	5.358	5.3470	5.4557	5.5802	5.2686	6.8561	5.6664
Boat	6.3857	3.9120	3.8253	4.6258	4.8146	4.6909	4.7850	4.6646	5.7469	5.2097
Peppers	7.2352	4.2868	4.0826	4.9982	5.3879	5.5308	5.6176	5.4084	6.5681	6.0995
Fruits	5.4861	3.1426	3.1138	3.9798	4.3304	4.0054	4.1481	4.0384	4.8940	4.3324
Couple	11.882	5.6279	5.5418	6.678	8.3154	8.5876	9.0331	8.0450	8.4878	9.3496
Girl	11.157	5.6124	5.4433	6.7925	7.8426	7.4886	7.7466	7.7247	9.4510	8.5264
Average	**7.7132**	**4.2819**	**4.1441**	**5.0802**	**5.6257**	**5.5903**	**5.7862**	**5.5037**	**6.5750**	**6.1126**
Palette size: 32										
Lenna	5.7174	3.2258	3.0769	3.9978	4.2656	4.1548	4.3888	3.8653	4.5761	4.4620
Baboon	8.1388	5.5644	5.3107	6.3545	6.4859	6.1560	6.3694	6.0041	7.7539	6.4359
Boat	8.0848	4.6100	4.413	5.6254	6.0734	5.6439	5.8113	5.4462	7.0438	6.1993
Peppers	10.180	6.3679	6.1713	7.7449	8.0440	7.7957	7.9312	7.5899	8.9396	8.4388
Fruits	7.5845	4.1855	4.1967	5.8532	6.2373	5.3853	5.5915	5.2303	6.4358	5.8271
Couple	10.961	6.3518	6.4344	7.3557	7.8506	8.6510	9.1970	7.5988	8.5580	8.9274
Girl	13.329	6.8493	6.7354	8.7243	9.4531	8.9461	9.3961	8.8682	10.728	10.122
Average	**9.1422**	**5.3078**	**5.1912**	**6.5223**	**6.9157**	**6.6761**	**6.9550**	**6.3718**	**7.7193**	**7.2018**

Table 6.2. CIELAB difference ΔE measurement of the outputs of various algorithms in restoring haftoned color-quantized images with the palette generated by median-cut algorithm [Heckbert 82] of size

	% of pixels whose CIELAB ΔE <3									
	Observed **Y**	Proposed	[Fung 04b] (Proposed Scheme in Chapter 5)	[Chan 05] (Proposed Scheme in Chapter 3)	[Fung 04a] (Proposed Scheme in Chapter 4)	[Altunbasak 01]-IND	[Altunbasak 01]-KL	[Mese 01]	[Galatsanos 91a]	[Hunt 84]
Palette size: 256										
Lenna	53.71	81.50	83.24	77.08	74.58	69.98	69.05	65.76	58.15	68.02
Baboon	36.55	51.01	51.73	50.15	45.92	43.96	43.44	44.90	37.07	42.19
Boat	45.43	60.81	61.03	55.26	56.17	55.08	54.75	53.26	46.84	52.29
Peppers	37.36	68.47	68.41	61.13	54.90	53.51	52.94	47.67	38.28	45.78
Fruits	63.48	81.08	82.64	77.42	73.81	73.60	72.72	70.84	65.67	71.34
Couple	14.79	35.54	35.29	31.33	29.21	24.33	23.78	21.01	19.26	20.65
Girl	21.72	45.58	44.55	39.71	36.69	33.02	32.30	29.14	26.45	28.25
Average	**39.01**	**60.57**	**60.98**	**56.01**	**53.04**	**50.50**	**49.85**	**47.51**	**41.67**	**46.93**
Palette size: 128										
Lenna	41.76	76.32	78.58	69.57	65.04	62.18	59.46	58.85	48.48	58.32
Baboon	24.37	42.48	44.02	38.54	37.18	32.81	32.29	34.77	25.58	31.75
Boat	37.33	56.05	56.24	48.11	48.12	48.33	47.57	47.86	39.72	44.61
Peppers	26.59	60.97	61.63	52.16	46.38	43.92	43.11	39.97	30.13	35.93
Fruits	50.42	76.20	78.17	69.66	63.20	64.79	63.17	63.38	55.27	60.46
Couple	8.57	30.74	30.22	26.57	22.22	17.38	16.41	16.57	14.51	14.60
Girl	13.39	37.71	36.94	29.90	24.62	24.72	23.72	22.40	19.35	19.81
Average	**28.92**	**54.35**	**55.11**	**47.79**	**43.82**	**42.02**	**40.82**	**40.54**	**33.29**	**37.93**
Palette size: 64										
Lenna	31.41	69.27	71.82	57.93	53.55	51.52	48.11	50.32	38.41	47.00
Baboon	15.15	34.89	36.84	27.89	27.17	23.50	22.89	26.36	16.40	22.90
Boat	29.25	50.01	51.33	41.12	40.69	41.18	40.22	41.50	34.55	37.03
Peppers	16.78	52.18	54.19	39.87	34.07	33.82	32.74	32.23	20.72	26.77
Fruits	38.88	70.80	72.78	60.42	53.87	56.28	54.02	56.94	43.95	51.01
Couple	3.89	24.28	24.92	18.11	10.10	10.10	9.21	10.55	8.58	7.78
Girl	4.95	30.32	32.11	20.50	13.95	15.26	14.52	13.44	9.60	12.13
Average	**20.04**	**47.39**	**49.14**	**37.98**	**33.34**	**33.10**	**31.67**	**33.05**	**24.60**	**29.23**
Palette size: 32										
Lenna	19.26	60.10	62.48	43.48	39.65	38.51	35.06	41.94	28.42	32.62
Baboon	12.30	26.16	28.47	21.23	19.00	18.79	17.66	21.11	14.06	17.24
Boat	23.37	44.30	47.22	34.26	33.06	34.89	33.80	38.34	29.65	30.28
Peppers	8.51	37.56	40.52	24.13	20.04	21.42	20.30	21.88	11.90	16.11
Fruits	24.26	58.35	58.86	40.26	35.00	39.43	37.80	43.74	31.28	33.97
Couple	4.72	20.69	18.76	15.34	11.92	8.89	7.86	12.34	9.04	7.46
Girl	3.51	24.07	26.76	13.02	9.64	11.03	10.01	10.82	6.81	8.60
Average	**13.70**	**38.75**	**40.44**	**27.39**	**24.04**	**24.71**	**23.21**	**27.17**	**18.74**	**20.90**

Table 6.3. Percentage of pixels whose CIELAB difference is not detectable ($\Delta E < 3$) after restoration when testing images were color-quantized with the palette generated by median-cut algorithm [Heckbert 82] of size



(a) Original (b) Halftoned Color-quantized

(c) [Galatsanos 91a] (d) [Hunt 84]

(e) [Altunbasak 01]-IND (f) [Altunbasak 01]-KL

Figure 6.2. Restored halftoned color-quantized “Fruits” of various approaches (palette is generated by median-cut algorithm [Heckbert 82] with size 128)

(g) [Mese 01]

(h) Proposed Scheme in Chapter 3

(i) Proposed Scheme in Chapter 4

(j) Proposed Scheme in Chapter 5

(k) Proposed

Figure 6.2(continue). Restored halftoned color-quantized “Fruits” of various approaches (palette is generated by median-cut algorithm [Heckbert 82] with size 128)

	Palette size	Observed (**Y**)	Restored (**X'**)								
			Proposed	[Fung 04b] (Proposed Scheme in Chapter 5)	[Chan 05] (Proposed Scheme in Chapter 3)	[Fung 04a] (Proposed Scheme in Chapter 4)	[Altunbasak 01]-IND	[Altunbasak 01]-KL	[Mese 01]	[Galatsanos 91a]	[Hunt 84]
a		Average of (SNR Improvement (dB))									
	256	-	6.5653	5.6935	5.2368	4.0159	4.0097	3.6977	3.7508	1.5941	2.8789
	128	-	7.3868	6.9857	5.7450	4.0430	4.7119	4.3549	4.8697	2.2585	3.5493
	64	-	8.8699	7.1172	5.9735	3.6709	5.4454	5.0258	6.6426	2.9054	4.2723
	32	-	8.7839	8.5174	5.1387	3.0231	5.4257	5.0174	7.5855	3.4035	4.4434
b		Average of (Average of CIELAB difference ΔE)									
	256	4.7305	3.3756	3.3831	3.6837	3.8534	3.9139	3.9739	4.0520	4.4882	4.2346
	128	5.8678	4.1237	4.1036	4.5161	4.8119	4.7375	4.8371	4.8130	5.4207	5.1096
	64	7.3602	5.1048	5.0632	5.8635	6.2701	5.8456	5.9957	5.7979	6.7306	6.2729
	32	9.3714	6.5337	6.4828	7.6850	8.1267	7.3447	7.5776	7.1194	8.5231	7.8172
c		Average of (% of pixels whose CIELAB $\Delta E < 3$)									
	256	39.46	57.95	58.38	53.96	50.75	49.68	48.88	47.06	41.56	46.55
	128	29.91	49.97	50.96	45.32	41.08	40.93	39.96	40.02	33.10	37.40
	64	22.77	42.94	43.67	36.02	31.53	32.42	31.44	33.60	25.94	28.72
	32	12.79	32.06	33.25	22.44	19.04	22.15	20.99	24.54	15.60	18.67

Table 6.4. Average performance of various algorithms in restoring halftoned color-quantized images in various aspects (palette was generated by the octree algorithm [Gervautz 90])

Table 6.1 shows the SNRI performance achieved by various algorithms when the involved palettes were obtained with the median-cut algorithm [Heckbert 82]. One can see that the proposed algorithm is superior to any other algorithms whatever color palette size is concerned. On average, the proposed algorithm achieved, respectively, a SNRI of 6.16, 7.89, 9.21 and 9.86dB when the involved palette is of size 256, 128, 64 and 32.

Tables 6.2 and 6.3 show the performance of the evaluated algorithms in terms of the CIELAB color difference (ΔE) metric. The CIELAB color difference (ΔE) metric is defined as the Euclidean distance between the original color of a pixel and its reproduction in CIELAB color metric space [CIE 78]. Table 6.2 shows the average of the ΔE values of all pixels in a restoration output and Table 6.3 shows the percentage of pixels whose color error is visually undetectable in a restoration output. Again, one can see that the proposed algorithm is superior to the others.

Figures 6.2 shows the restoration results of different algorithms for visual evaluation. Figure 6.2b was obtained with a palette of size 128. Though error diffusion can remove false contour and color shift to a certain extent, these artefacts still appear in the color-quantized outputs as the palette size is too small. Besides, pepper noise was introduced by error diffusion. After restoration, one can see that the proposed algorithm can remove most of the artefacts while the others are comparatively inferior in this aspect.

Table 6.4 shows the average performance of various algorithms when the palettes were generated with the octree algorithm [Gervautz 90]. The same set of standard testing images used to obtain the results presented in Tables 6.1 to 6.3 were used for evaluation. The figures in the Table are the average values obtained with the restoration results of the color-quantized images. One can see that the proposed

algorithm is still superior to the others. This simulation result shows that the proposed algorithm works with different palettes generated with different palette generation algorithms.

The algorithm proposed in Chapters 3 and 4 do not take error diffusion process into account and hence their inferior performance in handling halftoned color-quantized image is expected.

As compare with the restoration algorithm presented in Chapter 5, the performance of the proposed algorithm is a little bit better in terms of the SNRI measurement. In the measurement of CIELAB color difference (ΔE) metric, the performance of the POCS approach is a little bit better. In the visual evaluation, the restoration outputs are more or less the same as shown in Figure 6.2i and 6.2j. Since the proposed SA restoration algorithm accepts solution with deteriorated cost to a limited extent, it gives the ability to approach the global optimial while POCS approach cannot.

The number of parameters required to be determined in SA approach is relatively more compare with the POCS approach and the time required in SA approach to reach the same performance of POCS approach is longer.

6.4 Summary

In this Chapter, we presented a dedicated restoration algorithm for restoring halftoned color-quantized images based on Simulated Annealing. Like the algorithm proposed in Chapter 5, this algorithm makes a good use of the available color palette and the halftoning process to derive useful *a priori* information for restoration. Simulation results for comparative study demonstrate that the proposed algorithm can achieve a remarkable improvement in the quality of a halftoned color-quantized image

in terms of both SNRI and CIELAB color difference ΔE metric. The proposed algorithm can remove most of the artefacts introduced by color quantization in which error diffusion is involved and hence can improve the restoration result subjectively.

Chapter 7.

On the Production of Scalable Color-quantized Images for Printing Purpose

7.1 Introduction

Color-quantized images are mainly produced for printing purpose and displaying purpose. When they are produced for being displayed with a low end display unit, the color of each pixel is treated as a vector and a vector quantization process is performed to each pixel with a predefined or given palette. Halftoning may be used to improve the visual quality of the color-quantized image when the palette size is too small.

However, the situation is different for printing application, when one wants to produce a hardcopy of a color image, the image is usually decomposed into 3 or 4 separate color planes and each plane is halftoned separately as if they were gray level images. In such a case, producing a halftoned color-quantized image is equivalent to producing three separate gray level images and the involved halftoning algorithm plays a significant role in the color quantization process.

When one produces hardcopies for different clients over heterogenous networks, clients may need or only support hardcopies of different spatial resolutions under various constraints. In that case, it is desirable to make the halftoned color-quantized images to be printed scalable such that they can be delivered efficiently over the network or stored efficiently if necessary. When one renders halftone images for different printers of different resolutions, the output should better be scalable in a way that it embeds low resolution halftone images into a full-scale halftone image and, through a very simple procedure such as down-sampling, the low resolution halftone images can be obtained from the high resolution halftone image directly.

In this Chapter, we address this issue and propose an algorithm for producing scalable halftoned color-quantized images for printing applications. Note that, since scalable halftoned color-quantized hardcopies are produced by halftoning individual planes with a binary halftoning algorithm, the addressed problem can be turned into a problem concerning how to generate a scalable binary halftone and this is the reason why we devote the effort to formulating a scalable binary halftoning algorithm thoughout this Chapter.

7.2 Proposed Framework for Generating Scalable Binary Halftones with MED Algorithms

As mentioned in Chapter 1.2.2.2, the production of a scalable binary halftone can be modelled as a constrained halftoning problem. Multiscale error diffusion (MED) is a recently proposed halftoning technique and it was proven to be superior to conventional error diffusion as it can eliminate directional hysteresis completely [Katsavounidis 97]. Due to its frame-based nature, it is modified to realize constrained halftoning easily and effectively in this Chapter.

Suppose one wants to halftone a continuous-tone image X. Without loss of generality, we assume that X is of size $N \times N$ and X^r, the downscaled version of X, is of size $(N/s_r) \times (N/s_r)$, where $s_r \in \{ 2^r \mid r = 1,2 \ldots R; 2^R < 2^L = N \}$ is a desirable scaling factor. Note that this combination of N and s_r is picked for illustration only. In general, N and s_r can be any integer values and the framework presented here can be easily modified to handle it.

The objective is to produce an output such that all B^r can be obtained by simply down-sampling B, where B and B^r are, respectively, the halftone results of X and X^r. Note X can be downscaled with any approach to obtain X^r, producing different results. In our approach, X^r is obtained by averaging X as follows.

$$X^r_{(i,j)} = \frac{1}{s_r \times s_r} \left(\sum_{m=0}^{s_r-1} \sum_{n=0}^{s_r-1} X_{(s_r i+m, s_r j+n)} \right) \text{ for } i, j = 0,1,\ldots,(N/s_r)-1 \qquad (7.1)$$

where $X^r_{(i,j)}$ and $X_{(i,j)}$ are, respectively, the $(i, j)^{th}$ pixels of X^r and X.

In the proposed framework, starting with $r = R$, we iteratively generate B^r with X^r and then use B^r as a constraint to produce B^{r-1} in the next iteration until B is eventually obtained. As selected by the user, B^R is of the lowest resolution to be supported in the scalable B. There is no constraint to generate it and one can exploit any MED algorithms [Chan 98, Chan 04, Katsavounidis 97] by setting the input image to be X^R.

To obtain B^r with X^r for $0 < r < R$, the same MED algorithm can be applied by applying a constraint in the initialization stage of generating B^r. Suppose one has already obtained B^r with X^r and starts to produce B^{r-1} with X^{r-1}. At the very beginning, we force the down-sampled elements of B^{r-1} to be

$$B^{r-1}_{(m,n)} = B^{r}_{(i,j)} \qquad if \;\; (m,n) = (2i,2j) \text{ for } i, j = 0,1,\ldots,(N/s_r) - 1 \qquad (7.2)$$

where $B^{r}_{(i,j)}$ is the $(i, j)^{th}$ element of B^r and $B^{r-1}_{(m,n)}$ is the $(m,n)^{th}$ element of B^{r-1}. Note assignment (7.2) guarantees that B^r can be obtained by simply down-sampling B^{r-1}.

X^{r-1} is then updated by

$$X^{r-1}_{(2i+p,2j+q)} = \begin{cases} 0 & if\,(p,q) = (0,0) \\ X^{r-1}_{(2i+p,2j+q)} + W_{(p,q)}(X^{r-1}_{(2i,2j)} - B^{r-1}_{(2i,2j)}) & if\,(p,q) \in \{(u,v) \mid u,v = 0,\pm 1\} \setminus \{(0,0)\} \end{cases}$$
$$\text{for each constrained pixel } (i, j) \qquad (7.3)$$

where $X^{r-1}_{(m,n)}$ is the $(m,n)^{th}$ pixel of X^{r-1} and W is defined as W = $[W_{(-1,-1)}\, W_{(-1,0)}\, W_{(-1,1)}\; ;\; W_{(0,-1)}\, W_{(0,0)}\, W_{(0,1)}\; ;\; W_{(1,-1)}\, W_{(1,0)}\, W_{(1,1)}]$ = [1,2,1;2,-12,2;1,2,1]/12.

To handle corner or boundary pixels, W is modified and filters such as [0,0,0;0,-5,2;0,2,1]/5 and [0,0,0;2,-8,2;1,2,1]/8 are used instead to avoid energy leakage. After updating X^{r-1}, the remaining unprocessed pixels are processed with the selected MED algorithm as usual to produce B^{r-1}.

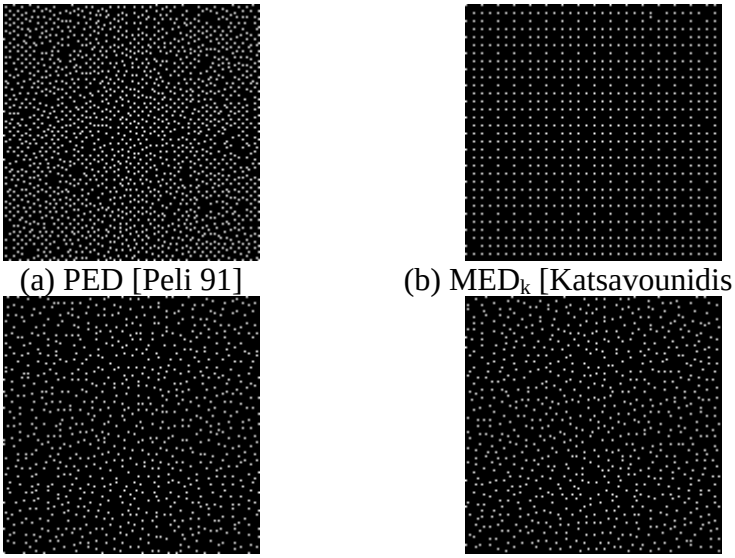
Note that this framework handles all constrained pixels before processing those non-constrained pixels. Besides, unlike CH, it does not confine the direction of error diffusion and hence provides more flexibility to compensate for the negative effect of assignment (7.2) in which a constrained pixel is assigned a value without concerning its original intensity. As a result, an output of higher image quality can be obtained.

7.3 Picking an Appropriate MED

Though the proposed framework works with any MED algorithms, we found that not every MED algorithm could provide a good performance in multi-resolution halftoning. To look for a good multi-resolution halftoning algorithm, two analyses were carried out to evaluate the impact of various error diffusion algorithms in constrained halftoning. The first analysis was carried out to evaluate the general halftoning performance of different MED algorithms. The second analysis was carried out to evaluate the performance of different MED algorithm when they were used in constrained halftoning. Based on the analysis results, we propose an appropriate MED algorithm to work with the framework proposed in Section 7.2 to produce scalable binary halftones.

7.3.1 Performance Analysis on MED Algorithms

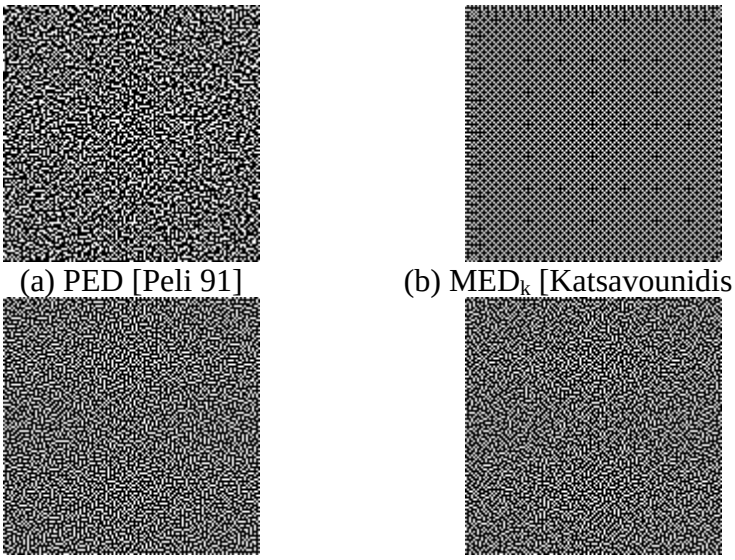
In our analysis, various error diffusion algorithms were applied to a constant gray-level image of size 128x128 and the dot distributions in their outputs were studied in terms of the statistics mentioned in Section 2.3.1. In this Section, we will focus on the results of multiscale-based algorithms including Peli's algorihm (PED) [Peli 91], Katsavounidis's algorithm (MED_k) [Katsavounidis 97] and Chan's algorithms (MEDc98) [Chan 98] and (MEDc04) [Chan 04]. Strictly speaking, PED is not a MED algorithm, but it distributes dots from a multiscale point of view.



(a) PED [Peli 91] (b) MED_k [Katsavounidis 97]

(c) MED_c98 [Chan 98] (d) MED_c04 [Chan 04]

Figure 7.1. Halftoning results of (a) PED, (b) MED_k, (c) MED_c98 and (d) MED_c04 for constant gray-level input (13/255) of size 128x128



(a) PED [Peli 91] (b) MED_k [Katsavounidis 97]

(c) MED_c98 [Chan 98] (d) MED_c04 [Chan 04]

Figure 7.2. Halftoning results of (a) PED, (b) MED_k, (c) MED_c98 and (d) MED_c04 for constant gray-level input (95/255) of size 128x128

Figure 7.1 shows the halftone results of a 128×128 image of constant gray level g=13/255. There are pattern artefacts in Figure 7.1b. Figures 7.1c and 7.1d are visually better than Figures 7.1a and 7.1b as they do not contain any directional ripples and pattern artefacts. As for Figure 7.1a, one can see that dots are denser than the other outputs. In fact, PED tends to introduce more minority dots than necessary,

which results in a brighter or darker output. Figure 7.2 shows another result when the constant gray level is 95/255.

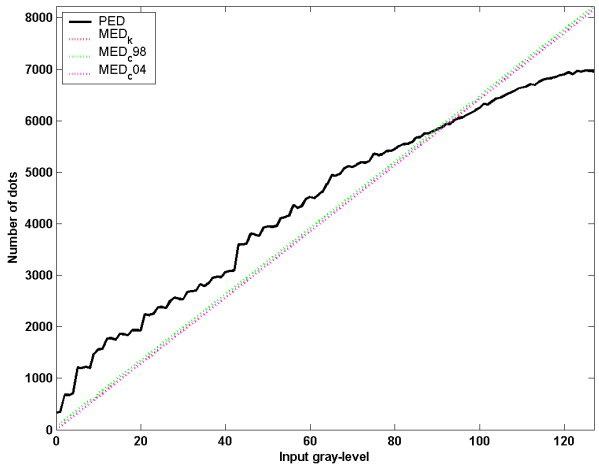
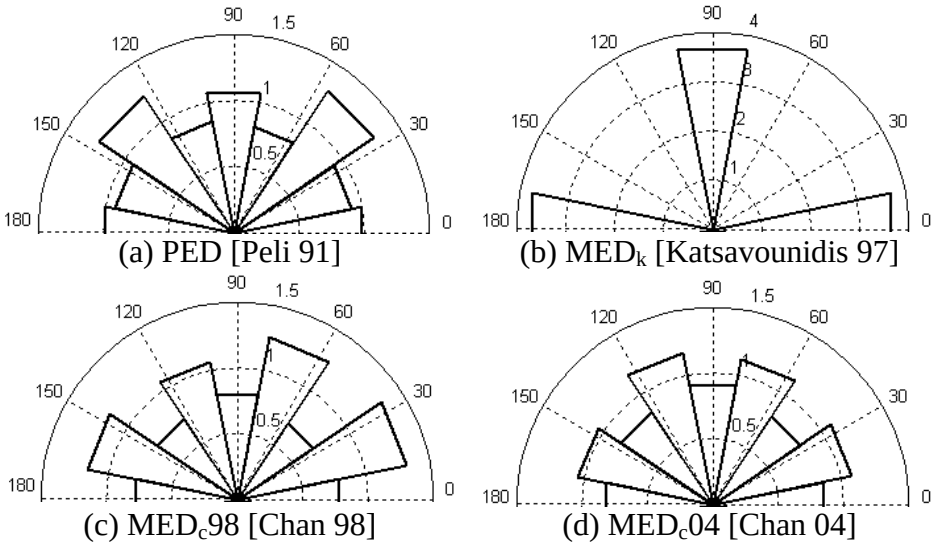


Figure 7.3. Performance in terms of Number of dots

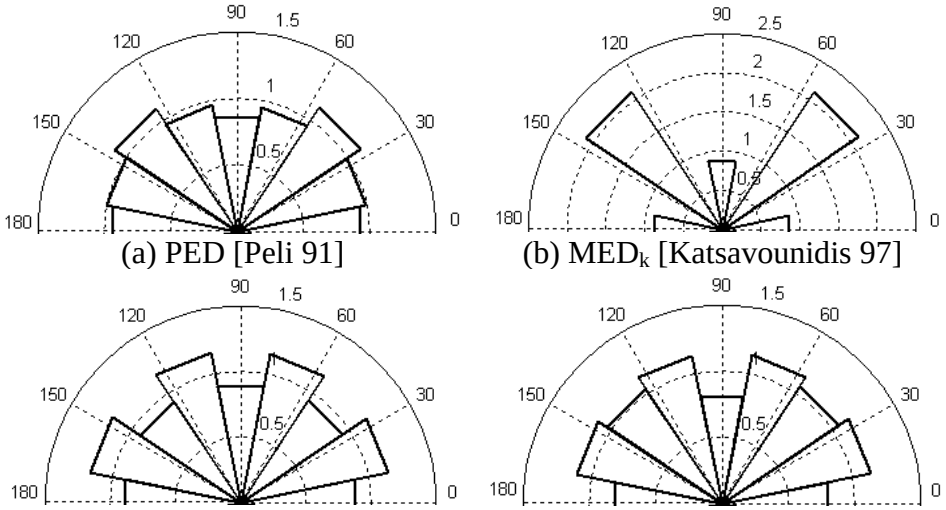
Figure 7.3 reports the deviation between the intensity of a constant input and the number of white dots in the corresponding halftone output. One can see that all presented algorithms can emulate the gray level except PED.



(a) PED [Peli 91] (b) MED_k [Katsavounidis 97]

(c) MED_c98 [Chan 98] (d) MED_c04 [Chan 04]

Figure 7.4. Corresponding directional distribution functions of Figure 7.1



(a) PED [Peli 91] (b) MED_k [Katsavounidis 97]

(c) MED_c98 [Chan 98] (d) MED_c04 [Chan 04]

Figure 7.5. Corresponding directional distribution functions of Figure 7.2

Figure 7.4 shows the directional distribution functions $D_{r_1,r_2}(\alpha)$ of the patterns shown in Figure 7.1. Only the upper halves of the plots are shown. The lower half is the mirror image of the upper half. Figure 7.5 shows another result of the directional distribution functions $D_{r_1,r_2}(\alpha)$ of the patterns shown in Figure 7.2.

Theoretically, MEDk, MEDc98 and MEDc04 can eliminate directional hysteresis as no causal filter and no predetermined scanning path is used in these algorithms. One can see that the plots in Figures 7.4b-d are symmetric in all four directions (East, North, West and South). This supports the theory. However, when the same issue is addressed at a finer direction resolution, MEDc98 and MEDc04 are better than MEDk in a way that their plots are symmetric in 8 directions while MEDk's one is not. In other words, MEDc98 and MEDc04 can eliminate directional hysteresis in more directions. PED's performance is comparable to MED_c's.

To have a better picture of the directional hysteresis introduced by a halftoning algorithm, a measure called *directional index function* is defined as

$$\sigma^2(g) = \frac{1}{16}\sum_{\alpha=1}^{16}(1 - D_{0,\max(\lambda,3)}(\alpha))^2 \qquad \forall g \tag{7.4}$$

where $D_{0,\max(\lambda,3)}(\alpha)$ is the directional distribution function values of the algorithm's halftoning output of a constant input, g is the gray level of the input and λ is the principal wavelength of the input. This measure does not carry any information about the direction of the directional hysteresis in the output. It simply reflects how severe the direction hysteresis is in the output. The greater the value, the more severe the direction hysteresis is. In ideal case, its value should be zero.

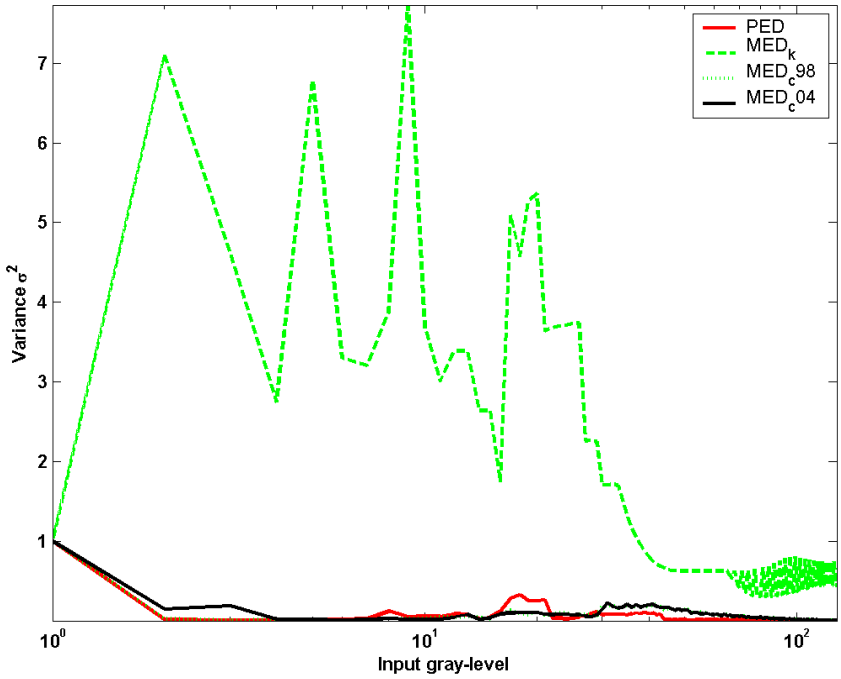
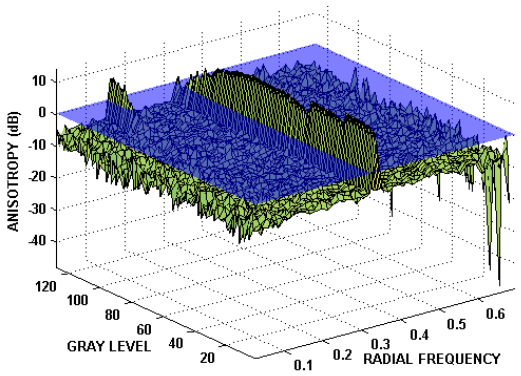
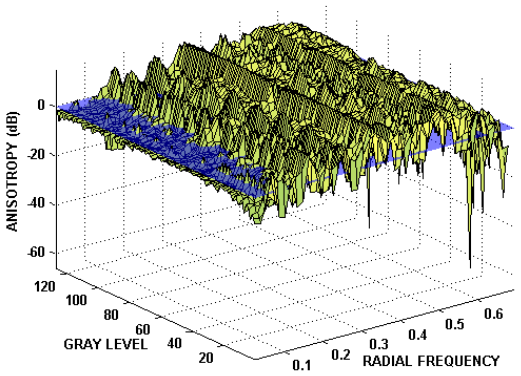


Figure 7.6. Performance in terms of directional distribution of dots

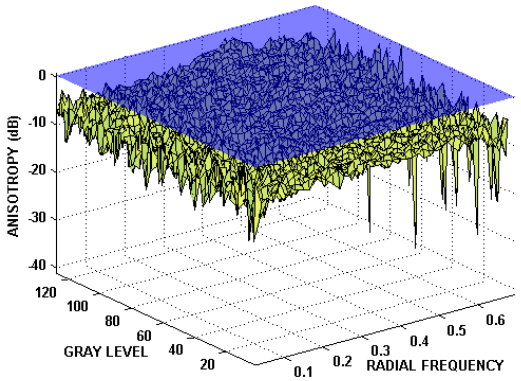
Figure 7.6 shows the directional index functions of the presented algorithms at different gray levels. Logarithmic scale is used for the abscissa in the plot. In Figure 7.6, one can see that PED, MEDc98 and MEDc04 can provide a good performance. Another interesting observation is that, though MED_k is symmetric in four directions, when the direction resolution is increased from 4 to 16, the directional index function values of MED_k are larger than those of the others in quite a number of gray level inputs.



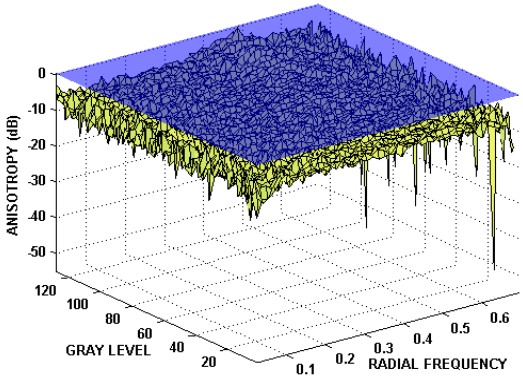
(a) PED [Peli 91]



(b) MED_k [Katsavounidis 97]



(c) MED_c98 [Chan 98]

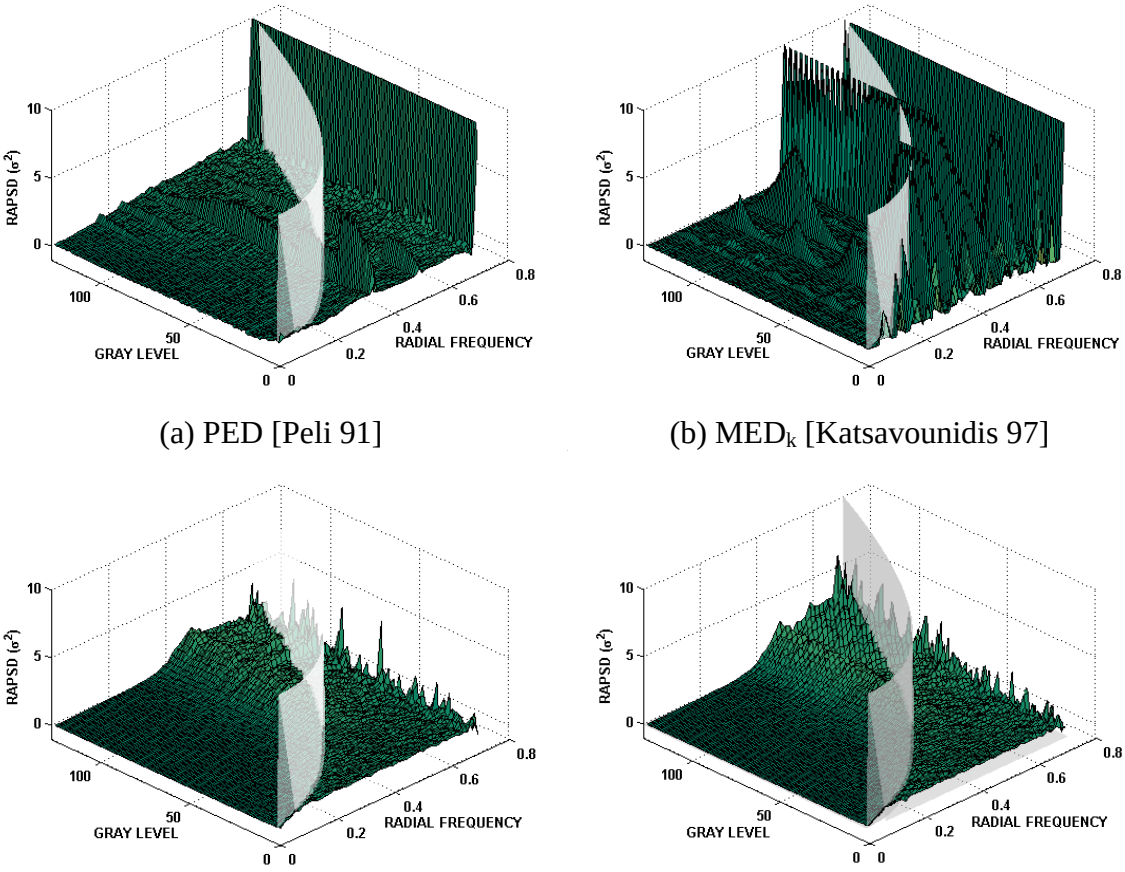


(d) MED_c04 [Chan 04]

Figure 7.7. Performance in terms of anisotropy (a) PED, (b) MED_k and (c) MED_c98 and (d) MED_c04

Figure 7.7 shows the performance of various algorithms in terms of the anisotropy of dots in their halftoning results of different constant gray-level inputs that $A(f_p)$ is larger than 0 db is considered strong in directional components and it is noticeable to the human eye [Ulichney 88]. To provide a reference to study the performance of the algorithms, a surface defined by $A(f_p)$=0 dB is added in each of the plots. The plots show that MEDc98 and MEDc04 are better than the other algorithms. Their corresponding anisotropy values are well below 0 dB in all combinations of gray levels and radial frequencies.

The output of blue noise halftoning is characterized by a distribution of binary pixels where the minority dots are spread as homogeneously as possible [Ulichney 88]. It is visually pleasant as it does not clash with the structure of an image. Pixels distributed in this way create an aperiodic and isotropic pattern and it does not contain any low-frequency spectral components.



(a) PED [Peli 91] (b) MED_k [Katsavounidis 97]

(c) MED_c98 [Chan 98] (d) MED_c04 [Chan 04]

Figure 7.8. Performance in terms of RAPSD (a) PED, (b) MED_k, (c) MED_c98 and (d) MED_c04

Figure 7.8 shows the performance of various algorithms in terms of RAPSD. For easier comparison, the range of RAPSD shown in all these plots is bounded to be less than 10. If a RAPSD value is larger than 10, it is clipped and the clipped value is displayed in the plots.

A good blue noise generator should produce a result which carries little or no low frequency spectral components, provides a flat high frequency spectral region and a spectral peak at blue noise principal frequency f_b. In order to provide a clear picture of the performance of the algorithm, a white surface which marks the principal frequency f_b for a particular gray level is added in each of the plots as a reference for comparison. Figure 7.8c-d show that the outputs of MEDc98 and MEDc04 have all these features. The harmonics appeared in the plot shown in Figure 7.8b explain why there are so many pattern artefacts in the outputs of MED_k.

7.3.2 Performance Analysis on Constrained MED Algorithms

In another analysis, multi-resolution halftoning was applied to a constant gray-level image of size 128x128 with various error diffusion algorithms and the dot distribution in their outputs was studied. Directional distribution function $D_{r_1,r_2}(\alpha)$ was used again to study the dot distribution in a halftone. In our analysis, the annular ring was defined by r_1=0 and r_2=3 and it was divided into 16 equal segments.

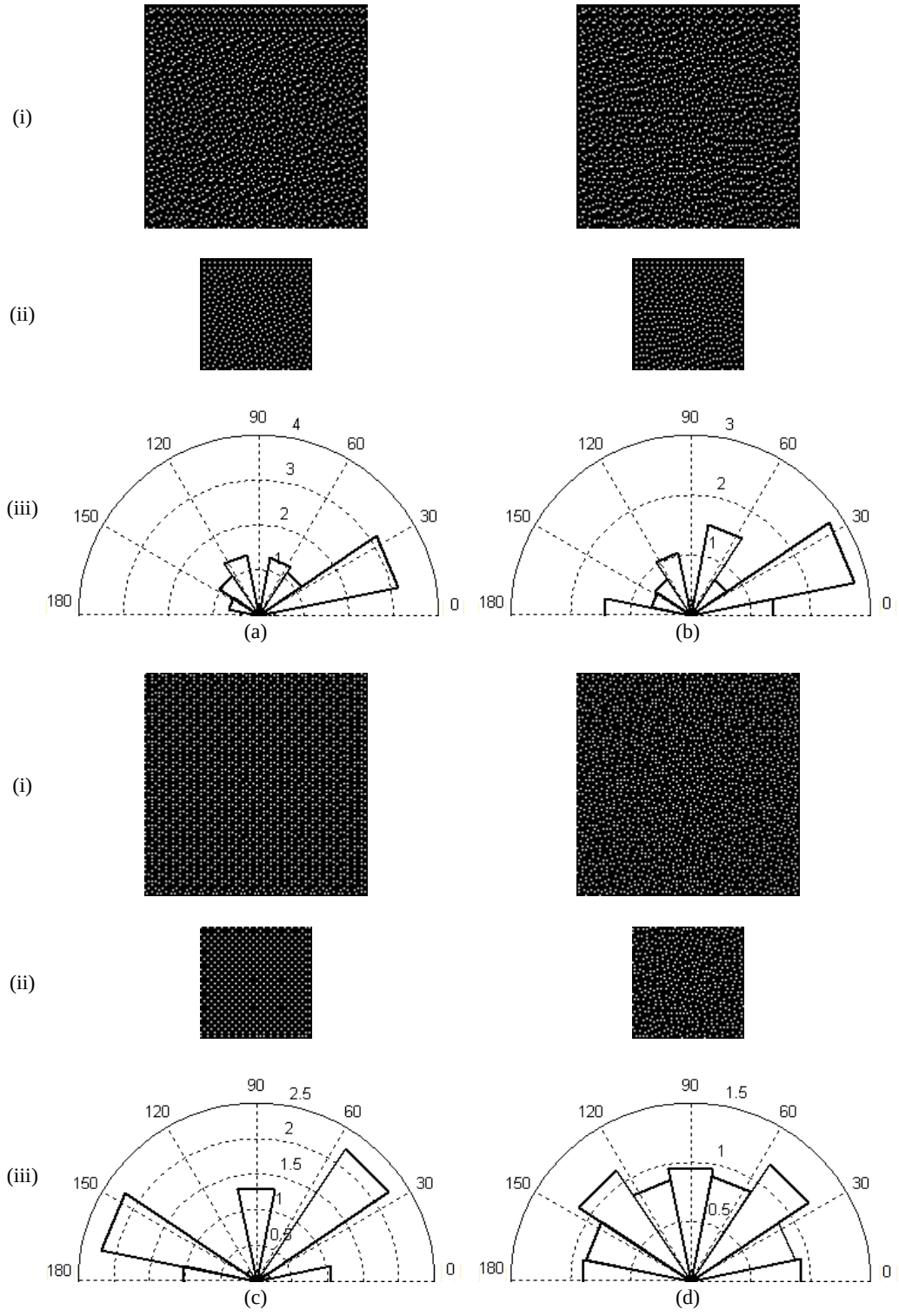


Figure 7.9. Halftoning results of (a) CH-SED, (b) CH-AED, (c) CH_p-MED and (d) the proposed algorithm for constant gray-level input (31/255) of size 128x128: (i) full-scale outputs, (ii) down-sampling results of (i) and (iii) directional distributions.

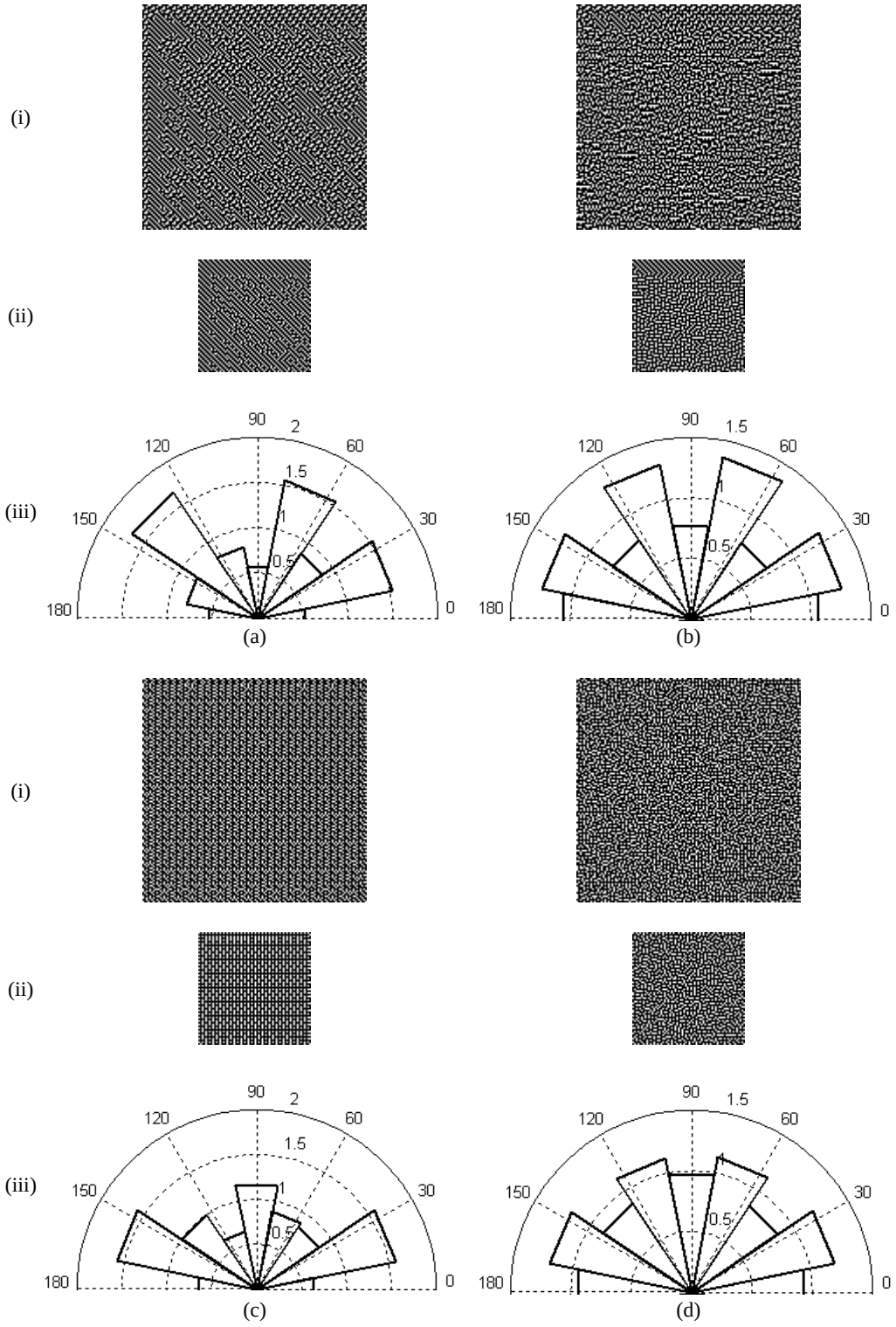


Figure 7.10. Halftoning results of (a) CH-SED, (b) CH-AED, (c) CH_p-MED and (d) the proposed algorithm for constant gray-level input (88/255) of size 128x128: (i) full-scale outputs, (ii) down-sampling results of (i) and (iii) directional distributions.

Figures 7.9(i)a-c show some multi-resolution halftoning results when the gray-level is 31/255 and Figures 7.9(ii)a-c show their corresponding down-sampling results. In particular, Figure 7.9(i)a shows the result of using framework CH and standard error diffusion [Floyd 76] (CH-SED) while Figure 7.9(i)b shows the result of [Wong 03] (CH-AED), which actually exploits framework CH and adaptive error diffusion. There exist directional ripples in Figures 7.9(i)a and 7.9(i)b. Figure 7.9(i)c shows the case of using the proposed constrained halftoning framework for MED algorithms with the MED algorithm proposed in [Katsavounidis 97] (CH_p-MED). It is visually smoother, but pattern artefacts can be observed.

Figures 7.9(iii)a-c show the directional distribution functions of Figures 7.9(i)a-c. One can see that all outputs suffer from directional hysteresis and dots are not uniformly distributed in all directions. For CH-SED and CH-AED, this is expected because of the reasons mentioned in Section 1.2.2.2. As for CH_p-MED, though theoretically directional hysteresis can be eliminated by MED, it appears again after constrained halftoning. The fixed error diffusion filter exploited in [Katsavounidis 97] diffuses quantization error to the constrained pixels and this amount of error is trapped there forever. CH_p-MED does not take care of the constrained pixels well in the course and error leakage happens in every constrained pixel.

To solve the problem encountered in CH_p-MED, an adaptive diffusion filter should be used to avoid diffusing error back to the constrained pixels. In this Section, we suggest using a modified version of the feature-preserving MED algorithm proposed in [Chan 04]. This algorithm detects whether white or black dots are minority dots in a local region and then assigns a minority dot to the region so as to preserve the local feature. In constrained halftoning, when generating the halftoning

result of higher resolution, constrained pixels are handled first without concerning any local feature. After that, the critical pixel positions in the region for displaying the local feature may have already been occupied by constrained pixels. Even though the feature-preserving mechanism of this algorithm is activated, there is not much gain in the quality and sometimes it may even make it worse. Hence, when this MED algorithm [Chan 04] works with constrained halftoning in our suggested multi-resolution halftoning algorithm (CH_p-MED_p), this feature-preserving mechanism is purposely off. Accordingly, no region-oriented detection of minority dots is performed and it always locates the minority dots of the whole image. Table 7.1 contrasts the differences among the algorithms evaluated in the analysis.

Algorithm	Scanning order	Error diffusion filter	Constrained halftoning
SED [Floyd 76]	Raster	Non-adaptive, causal	No
CH-SED	Raster	Non-adaptive, causal	Scheme in Section 1.2.2.2
CH-AED [Wong 03]	Raster	Adaptive, causal	Scheme in Section 1.2.2.2
CH_p-MED	Max intensity guidance [Katsavounidis 97]	Non-adaptive, non-causal	Scheme in Section 7.2
Proposed	Max intensity guidance[1]	Adaptive, non-causal	Scheme in Section 7.2

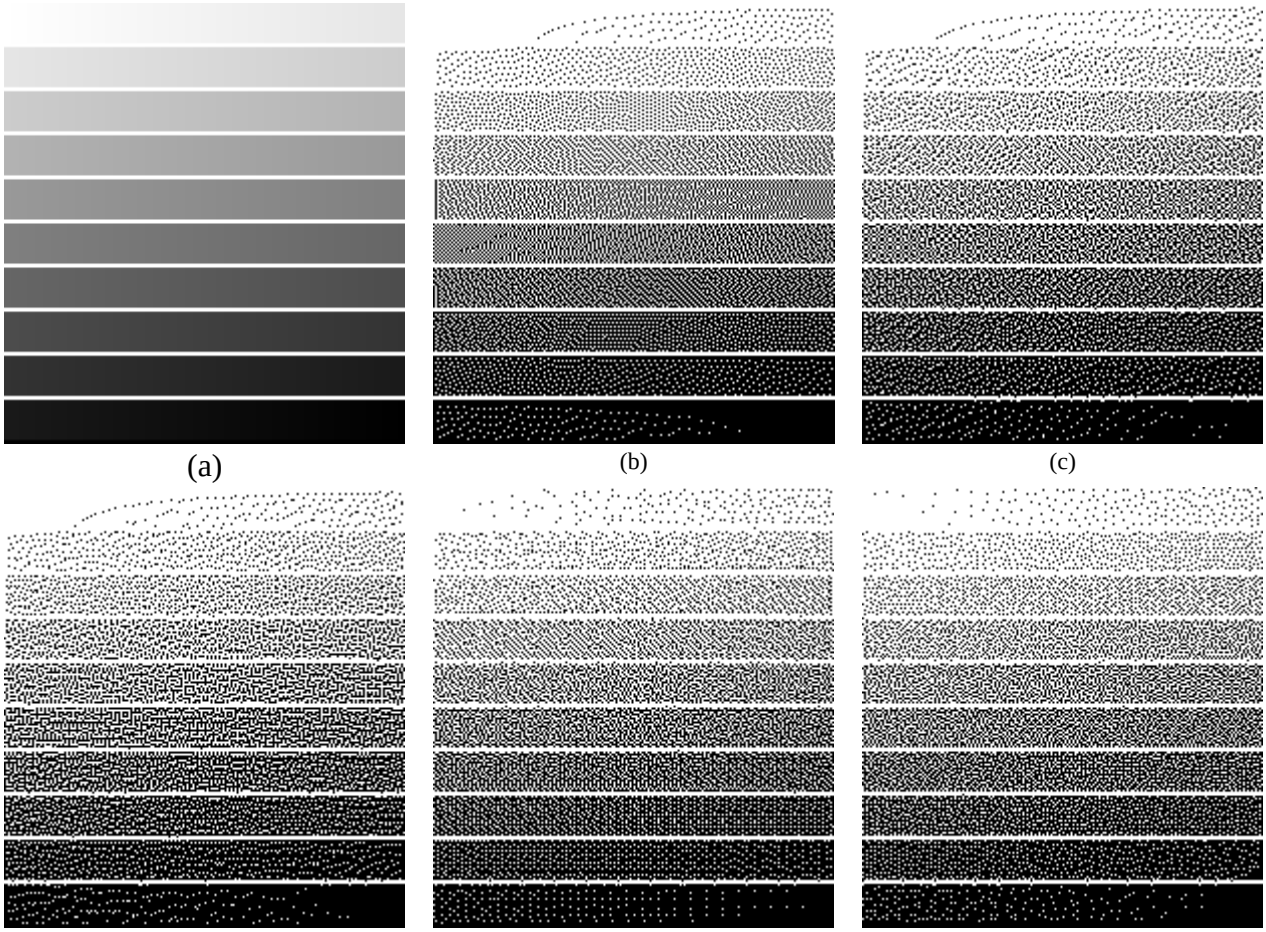
[1] The scheme used in [Chan 04] without turning the feature preserving mechanism on

Table 7.1 Summary of the algorithms evaluated for comparison

Figure 7.9(i)d shows the full-scale halftoning result of the proposed algorithm and Figure 7.9(ii)d shows its down-sampling result. There is no directional hysteresis and little, if any, pattern artefact in both images. The directional distribution function shown in Figure 7.9(iii)d also verifies that dots are evenly distributed in all directions. Figure 7.10 shows another result when the constant gray level is 88/255.

7.4 Comparative Study on the Performance of Different Constrainted Halftoning Algorithms

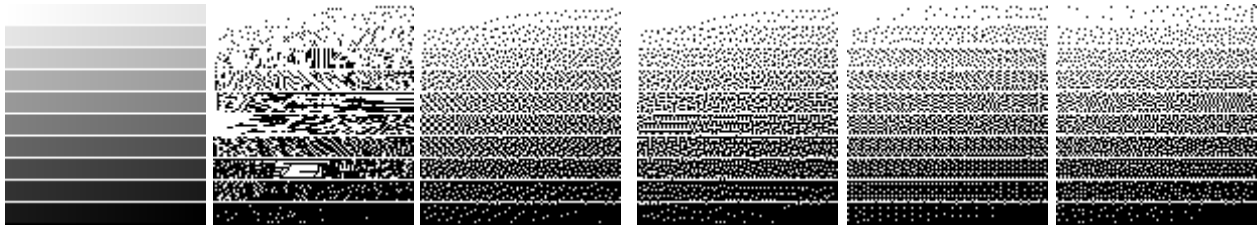
Simulation was carried out to evaluate the performance of various multi-resolution halftoning algorithms. This Section shows the performance in generating scalable binary halftones and scalable color prints.



(a) (b) (c)

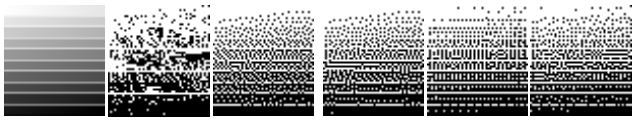
(d) (e) (f)

Figure 7.11. (a) Original ramp image and halftone results of (b) [Floyd 76], (c) CH-SED, (d) CH-AED, (e) CH_p-MED and (f) the proposed algorithm.



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

Figure 7.12. Down-sampling results of Figures 7.11a-7.11f.



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

Figure 7.13. Down-sampling results of Figures 7.11a-7.11f.

Figure 7.11a shows the original testing ramp image of size 200×218 and Figure 7.12a and 7.13a are its down-sampling results. As shown in Figures 7.11b, 7.12b and 7.13b, a plain conventional halftoning algorithm such as standard error diffusion [Floyd 76] does not embed a low-resolution halftoning result into its output and hence the down-sampling results of its output can be very poor. Figures 7.11c-7.11f show, respectively, the corresponding multi-resolution halftoning results of using CH-SED, CH-AED, CH_p-MED and the proposed algorithm. Figures 7.12a-7.12f and Figure 7.13a-7.13f are the down-sampling results of Figures 7.11a-7.11f. A down-sampling factor of 2 was applied to each direction. In the realization of CH_p-MED and the proposed algorithm, R was selected to be 2. The performance of the proposed algorithm is the best in terms of both directional hysteresis and pattern noise.

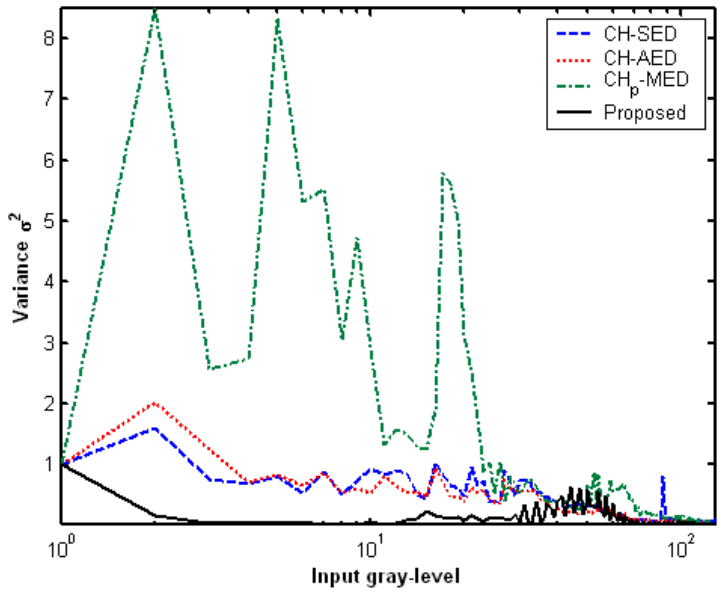


Figure 7.14. Performance in terms of directional distribution of dots of full-scale halftones

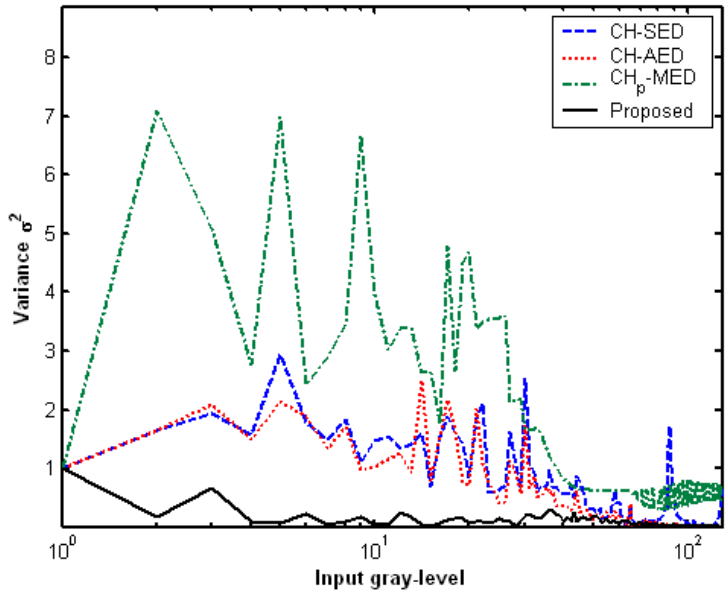
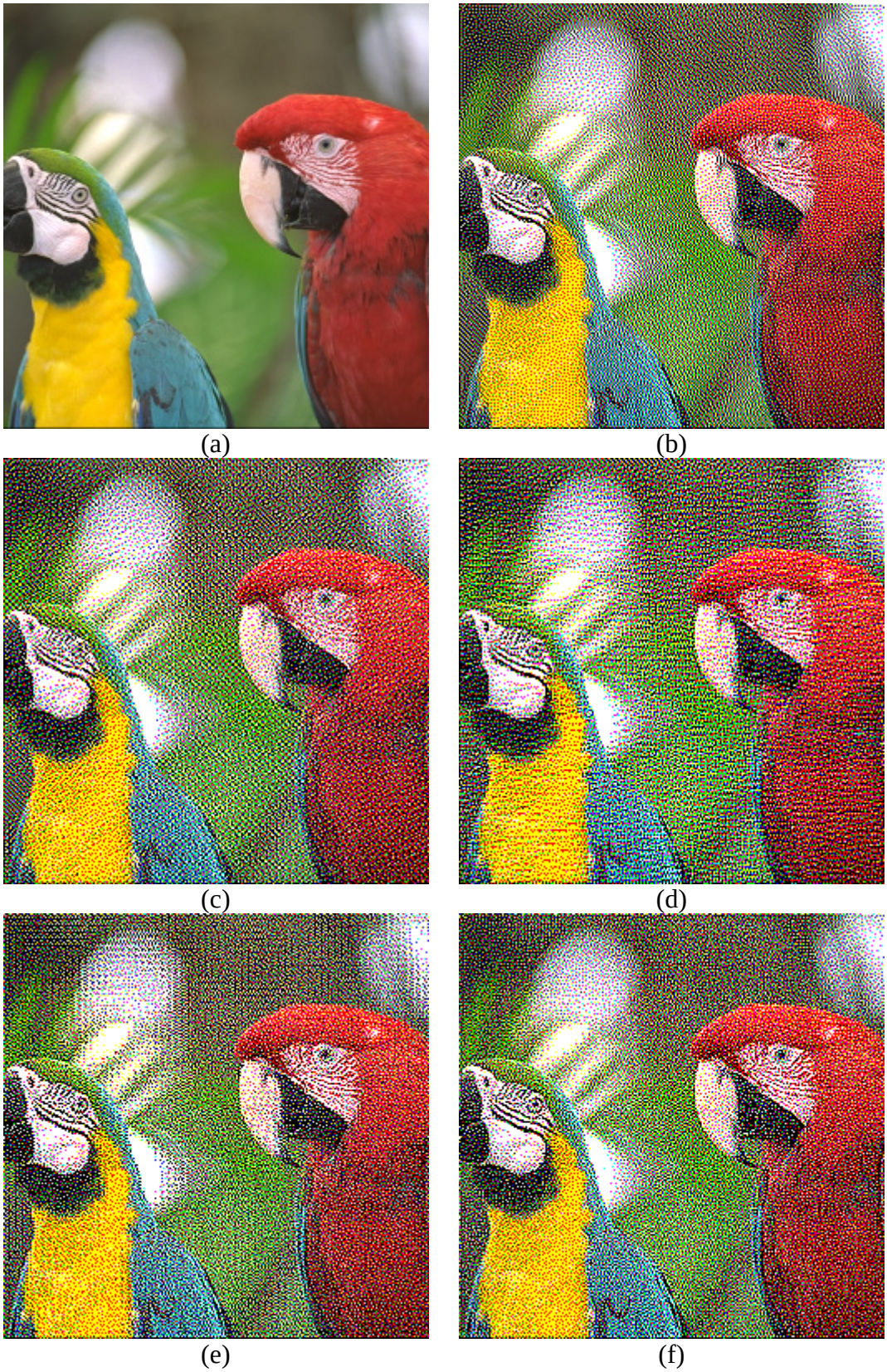


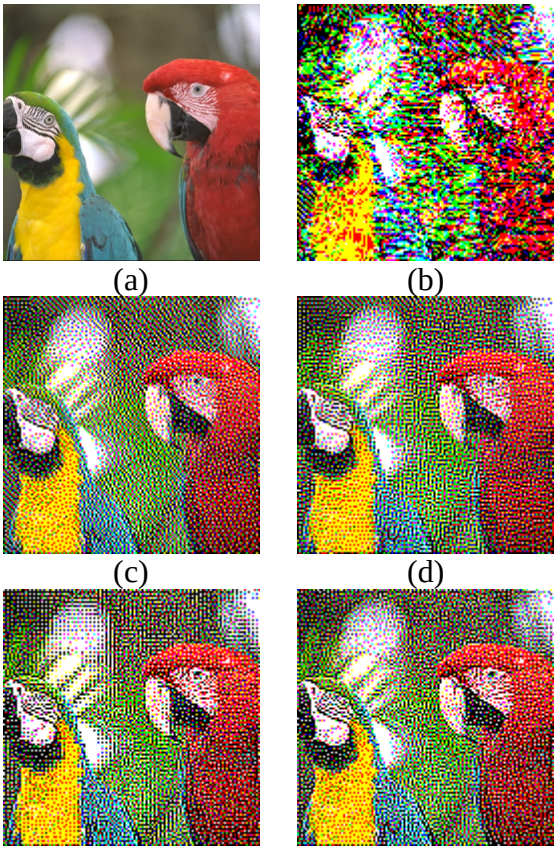
Figure 7.15. Performance in terms of directional distribution of dots of down-sampled halftones

Figure 7.14 and 7.15 show the performance of various algorithms in terms of the directional distribution of dots in their halftoning results and their down-sampled versions of different constant gray-level inputs. The performance is measured in terms of the variance of $(1\text{-}D_{0,\max(\lambda,3)}(\alpha))$ for all α, where λ is the principle wavelength [Ulichney 88] of the input gray level. Logarithmic scale is used for the abscissa in the plot. The plot shows that the proposed algorithm outperforms the other approaches.



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

Figure 7.16. (a) Original image “Parrots” and halftone results of (b) [Floyd 76], (c) CH-SED, (d) CH-AED, (e) CH_p-MED and (f) the proposed algorithm.



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

Figure 7.17. Down-sampling results of Figures 7.16a-7.16f.

As mentioned in Section 7.1, the hardcopy of a color image is produced by halftoning the color planes of the color image independently with a binary halftoning algorithm. The binary algorithm presented in this Chapter can be easily extended to generate a color print of color image.

Figure 7.16 shows color prints when testing image “Parrots” was used. In the evaluation, a true color image is first decomposed into three-color planes and each plane is halftoned with various multiresolution halftoning algorithms. We can see from the simulation results that the proposed algorithm is better than the others. Figure 7.17 shows the downsampled results of Figure 7.16 that R was selected to be 1.

7.5 Summary

In this Chapter, we presented a constrained halftoning framework for MED algorithms to realize multi-resolution halftoning. Based on this framework, we proposed a multi-resolution halftoning algorithm for producing scalable color-quantized images for printing applications. Given an image, the algorithm produces a set of halftoning results of different resolution and embeds those of lower resolution into the full-scale one such that they can be extracted by direct down-sampling. Simulation results show that the proposed algorithm can provide good halftoning results at different resolutions. Some critical factors for achieving good constrained halftoning results are also discussed.

In the next Chapter, we will extend the proposed framework to produce scalable color-quantized images for display purpose.

Chapter 8.

On the Production of General Scalable Halftoned Color-quantized Images

8.1 Introduction

Most halftoning algorithms are originally proposed for binary halftoning which emulates a gray level image with a binary image [Ulichney 91, Lau 00, Floyd 76, Jarvis 76, Stucki 81]. To apply them to color quantization, the most straightforward approach is to consider each color component plane as an individual gray scale image and handle them separately [Gentile 90, Zakhor 93, Damera-Venkata 03].

However, this approach may only work in printing applications in which the output colors are composed of 3 or 4 bi-level fixed color components (CMY or CMYK). Color quantization is actually a vector quantization instead of a bi-level uniform scalar quantization as in the case of binary halftoning. It is not a combination of several independent bi-level uniform scalar quantization processes either. The color reduction process involved in printing applications is only a special case of color quantization in where the three-dimensional color space is separated into three one-

dimensional spaces and processed separately. In this special case, the involved palette contains colors which are uniformly distributed over the color space. In practice, the aforementioned straightforward extension of binary halftoning only works when a uniform palette [Heckbert 82] is used in a color quantization process.

In general, when a low-end display unit such as a VGA monitor is involved, the palette colors are not uniformly distributed in the color space and hence the extension of binary halftoning to produce color-quantized image is not as straightforward as most people assume. Besides, this approach is not an effective approach as it does not take the correlation among color components into account.

When one delivers media information to diverse clients over heterogeneous networks, clients may support different display resolutions and systems may have different caching capabilities. In that case, it is desirable to make media information scalable such that it can be delivered efficiently and reliability. Since color-quantized images are widely used in multimedia applications nowadays, it is desirable to make them scalable such that their downscaled versions can be obtained directly with the images through some simple operations.

In this Chapter, a multiscale color error diffusion algorithm is proposed. Unlike those halftoning algorithms for printing color images, the proposed algorithm does not handle color planes separately and is able to handle color quantization in which any arbitrary palettes can be used. With any arbitrary palette, this algorithm is able to produce an output of high visual quality for any given full-color image. Directional hysteresis can be completely eliminated and color impulse can be greatly reduced. Based on the idea presented in Chapter 7, the proposed multiscale color error diffusion algorithm is then further extended to produce scalable halftoned color quantized images.

This Chapter is organized as follows. A multiscale color error diffusion algorithm is first proposed in Section 8.2. Section 8.3 presents an extension of the algorithm to generate scalable halftoned color-quantized image. The performance of the algorithm and its extension is evaluated in Section 8.4 and 8.5. A conclusion is given in Section 8.6.

8.2 Multiscale Color Error Diffusion Algorithm

In our proposed algorithm, color quantization is performed in YIQ color space so as to reduce the correlation among different color components. Another reason for doing so is that Euclidean distance in YIQ space matches HVS response more closely as compared with that in RGB space. This allows the color quantizer to select a visually more appropriate palette color with a given input. Without lose of generality, hereafter, we assume the color palette and the input image are defined in YIQ space. If they are not, color transformation will be required to transform their colors from their original domain to YIQ domain before color quantization.

Let $\mathbf{X}$ be a 24-bit $N \times N$ true-color image each pixel of which is represented as $\vec{\mathbf{X}}_{(i,j)} = (X_{(i,j)Y}, X_{(i,j)I}, X_{(i,j)Q})$, where $X_{(i,j)c}$ for $c \in \{Y, I, Q\}$ is the intensity value of the c^{th} primary color component of the $(i, j)^{th}$ pixel of the image.

The proposed algorithm is an iterative algorithm developed based on Katsavounidis's work [Katsavounidis 97]. Let $\mathbf{U}$ be an image which reports the current status of the image being processed at the beginning of a particular iteration. At each iteration, the algorithm first locates a pixel location based on the maximum energy guidance with an energy pyramid $\mathbf{E}$ associated with $\mathbf{U}$. The details of the pyramid will be elaborated later. The selected pixel is then color-quantized with a predefined set of colors (palette). The quantization error is diffused with a non-casual

filter to neighboring pixels to update $\mathbf{U}$. These procedures are repeated until all pixels are color-quantized. At the start of the first iteration, $\mathbf{U}$ is initialized to be $\mathbf{X}$.

8.2.1 Constructing Energy Pyramid $\mathbf{E}$

Let $\mathbf{M}$ be a mask of size $N \times N$ which defines which pixels have been color-quantized. Specifically, its element $M_{(i,j)}$ is 0 if $\vec{\mathbf{X}}_{(i,j)}$ has been color-quantized or else it is 1.

A multiscale representation of a given color image $\mathbf{U}$ is defined as a sequence of matrices $\{\mathbf{U}^0, \cdots \mathbf{U}^l, \cdots \mathbf{U}^L\}$, where $L = \log_2 N$ and $\mathbf{U}^L = \mathbf{U}$. $\mathbf{U}^l$ is of size $2^l \times 2^l$ and its $(i,j)^{th}$ element is a triplet ($U^l_{(i,j)Y}$, $U^l_{(i,j)I}$, $U^l_{(i,j)Q}$) for $i, j = 0,1,\ldots,2^l - 1$. Elements of $\mathbf{U}^l$ for $l = 0, 1 \cdots L-2$ is defined as

$$U^l_{(i,j)c} = \sum_{m=0}^{1}\sum_{n=0}^{1} U^{l+1}_{(2i+m,2j+n)c} \qquad \text{for } c \in \{Y,I,Q\} \quad (8.1)$$

while elements of $\mathbf{U}^{L-1}$ is defined as

$$U^{L-1}_{(i,j)c} = \begin{cases} \dfrac{1}{S}\displaystyle\sum_{m=0}^{1}\sum_{n=0}^{1} M_{(2i+m,2j+n)} U^L_{(2i+m,2j+n)c} & \text{if } S \neq 0 \\ 0 & \text{else} \end{cases} \quad \text{for } c \in \{Y,I,Q\} \quad (8.2)$$

where

$$S = \sum_{m=0}^{1}\sum_{n=0}^{1} M_{(2i+m,2j+n)} \quad (8.3)$$

The energy pyramid $\mathbf{E}$ associated with image $\mathbf{U}$ is then constructed with $\{\mathbf{E}^l \mid l = 0,1,\cdots L\}$, where $\mathbf{E}^l$ is the energy plane of matrix $\mathbf{U}^l$. The $(i,j)^{th}$ element of $\mathbf{E}^l$ is defined as

$$E^{l}_{(i,j)} = \begin{cases} \left| U^{l}_{(i,j)Y} + U^{l}_{(i,j)I} + U^{l}_{(i,j)Q} \right| & if\ 0 \le l < L \\ \left| M_{(i,j)} \left(U^{L}_{(i,j)Y} + U^{L}_{(i,j)I} + U^{L}_{(i,j)Q} \right) \right| & if\ l = L \end{cases}$$

$$\text{for } i, j = 0,1,\ldots,2^{l} - 1 \quad (8.4)$$

8.2.2 Searching the Pixel for Color Quantization

The location of a pixel to be color-quantized is determined via maximum energy guidance with energy pyramid $\mathbf{E}$. The location is obtained by searching the energy pyramid from the coarsest level $\mathbf{E}^0$ to the finest level $\mathbf{E}^L$. Note that $\mathbf{E}^0$ contains only one element $E^0_{(0,0)}$.

Assume that we are now at position $(l,(i,j))$ which corresponds to the $(i,j)^{th}$ element of a particular level l. We check $\{E^{l+1}_{(2i+m,2j+n)} \mid m,n = 0,1\}$ and proceed to the position $(l+1,(2i+p,2j+q))$ such that $E^{l+1}_{(2i+p,2j+q)}$ is maximum in $\{E^{l+1}_{(2i+m,2j+n)} \mid m,n = 0,1\}$ and $p,q \in \{0,1\}$. If more than one position satisfies the criterion, one of them will be randomly selected.

8.2.3 Color Quantization and Error Diffusion

Let $(L,(m,n))$ be the position that we finally reach at the finest level of the pyramid $\mathbf{E}$ in the search and $C = \{\hat{\mathbf{v}}_i : i = 1,2,\cdots N_c\}$ be the given color palette. $\vec{\mathbf{U}}_{(m,n)} = (U_{(m,n)Y}, U_{(m,n)I}, U_{(m,n)Q})$ is then color-quantized. The best-matched color in the palette, say $\hat{\mathbf{v}}_k$, is selected based on the minimum Euclidean distance criterion in YIQ color space as follows.

$$\left\| \vec{\mathbf{U}}_{(m,n)} - \hat{\mathbf{v}}_k \right\| \le \left\| \vec{\mathbf{U}}_{(m,n)} - \hat{\mathbf{v}}_l \right\| \qquad \forall\ \hat{\mathbf{v}}_l \in C \quad (8.5)$$

The quantization error $\vec{\varepsilon} = \hat{\mathbf{v}}_k - \vec{\mathbf{U}}_{(m,n)}$ is then diffused to $\vec{\mathbf{U}}_{(m,n)}$'s neighborhood to update image $\mathbf{U}$ with a non-causal filter. In formulation, it is given as

$$\vec{\mathbf{U}}_{(i,j)} = \vec{\mathbf{U}}_{(i,j)} + W_{(m-i,n-j)}\vec{\varepsilon}$$

$$\text{for} \qquad i = m \pm 1 \text{ and } j = n \pm 1 \qquad (8.6)$$

where W is defined as $W = \begin{bmatrix} W_{(-1,-1)} & W_{(-1,0)} & W_{(-1,1)} \\ W_{(0,-1)} & W_{(0,0)} & W_{(0,1)} \\ W_{(1,-1)} & W_{(1,0)} & W_{(1,1)} \end{bmatrix} = \begin{bmatrix} 0.0833 & 0.1667 & 0.0833 \\ 0.1667 & -1.0000 & 0.1667 \\ 0.0833 & 0.1667 & 0.0833 \end{bmatrix}$.

To handle the boundary and the corner pixels, W is modified to be $\begin{bmatrix} 0.0000 & 0.0000 & 0.0000 \\ 0.2500 & -1.0000 & 0.2500 \\ 0.1250 & 0.2500 & 0.1250 \end{bmatrix}$ and $\begin{bmatrix} 0.0000 & 0.0000 & 0.0000 \\ 0.0000 & -1.0000 & 0.4000 \\ 0.0000 & 0.4000 & 0.2000 \end{bmatrix}$ respectively to avoid energy leakage.

8.3 Extension to Produce Scalable Halftoned Color-quantized Images

With the framework presented in the previous Section 8.2, a color quantization algorithm for generating scalable color quantization images is proposed in this Section.

Consider the case that one wants to produce a color quantization result of a given image $\mathbf{I}$ which embeds a set of color quantization results of downscaled versions of $\mathbf{I}$. Let $\mathbf{I}^r$ be one of the downscaled versions of $\mathbf{I}$. Without loss of generality, we assume that $\mathbf{I}$ is of size $N \times N$ and $\mathbf{I}^r$ is of size $(N/s_r) \times (N/s_r)$, where $s_r \in \{2^r \mid r = 1,2...R; R < L\}$ is a desirable scaling factor. The objective of the proposed algorithm is to produce an output such that all $\mathbf{Y}^r$ can be obtained by simply down-sampling $\mathbf{Y}$, where $\mathbf{Y}$ and $\mathbf{Y}^r$ be, respectively, the color quantization result of $\mathbf{I}$ and $\mathbf{I}^r$.

Note $\mathbf{I}$ can be downscaled with any approach to obtain $\mathbf{I}^r$, producing different results. In our proposed algorithm, $\mathbf{I}^r$ is obtained by averaging $\mathbf{I}$ as follows.

$$I^r_{(i,j)c} = \frac{1}{s_r \times s_r} \sum_{m=0}^{s_r-1} \sum_{n=0}^{s_r-1} I_{(s_r i+m, s_r j+n)c} \quad \text{for } i, j = 0,1,...,(N/s_r)-1 \text{ and } c \in \{Y, I, Q\} \quad (8.7)$$

where $I^r_{(i,j)c}$ and $I_{(i,j)c}$ are, respectively, the c^{th} color components of the $(i, j)^{th}$ pixels of $\mathbf{I}^r$ and $\mathbf{I}$.

In the proposed algorithm, starting with $r = R$, we iteratively generate $\mathbf{Y}^r$ with $\mathbf{I}^r$ and then use $\mathbf{Y}^r$ as a constraint to produce $\mathbf{Y}^{r-1}$ in the next iteration until $\mathbf{Y}$ is eventually obtained.

As selected by the user, $\mathbf{Y}^R$ is of the lowest resolution to be supported in the scalable $\mathbf{Y}$. There is no constraint to generate it and one can make use of the multiscale error diffusion algorithm presented in Section 8.2 to generate it with $\mathbf{X} = \mathbf{I}^R$.

To obtain $\mathbf{Y}^r$ with $\mathbf{I}^r$ for $0 < r < R$, the same multiscale error diffusion algorithm presented in Section 8.2 can be used by embedding a constraint in the initialization stage. Suppose one has already obtained $\mathbf{Y}^r$ with $\mathbf{I}^r$ and starts to produce $\mathbf{Y}^{r-1}$ with $\mathbf{I}^{r-1}$. At the start of the first iteration, after initializing $\mathbf{U}$ to be $\mathbf{X} = \mathbf{I}^{r-1}$, we force the down-sampled elements of $\mathbf{Y}^{r-1}$ to be

$$Y^{r-1}_{(2i,2j)c} = Y^r_{(i,j)c} \qquad \text{for } i, j = 0,1,...(N/s_r)-1 \qquad (8.8)$$

where $Y^r_{(k,l)c}$ is the c^{th} color component of the $(k,l)^{th}$ element of $\mathbf{Y}^r$, and then diffuse the quantization error at positions $(2i,2j)$'s with eqn.(8.6) to update $\mathbf{U}$. Note assignment (8.8) guarantees that $\mathbf{Y}^r$ can be obtained by simply down-sampling $\mathbf{Y}^{r-1}$.

This completes the first iteration and the following iterations are carried out as usual as it is presented in Section 8.2 until $\mathbf{Y}^{r-1}$ is obtained.

8.4 Performance on Multiscale Color Error Diffusion

In this Section, the performance of the proposed multiscale color error diffusion algorithm is evaluated. The focus is on its performance on generating non-scalable halftoned color-quantized images. For the performance on generating scalable halftoned color-quantized images, it will be discussed in Section 8.5.

8.4.1 Details of the Simulation

Simulations were carried out on a number of *de facto* standard 24-bit full-color images of size 256×256 each to evaluate the performance of the proposed algorithm. These images were color-quantized with color palettes of different size. The color palettes were generated with different palette generation algorithms.

For comparison, some other error diffusion algorithms for color quantization were also evaluated [Akarun 97, Breaux 99, Orchard 91, Özdemir 00]. Unlike most color halftoning algorithms which are dedicated for printing applications, these evaluated algorithms are not straightforward extension of binary halftoning and are able to handle color quantization in which any arbitrary palettes can be used. In the realization of Orchard's algorithm [Orchard 91], Floyd-Steinberg filter [Floyd 76] was used in error diffusion.

S-CIELab color difference ($\Delta\mathbf{E}$) metric [Zhang 96] is a spatial extension of the CIELab color difference ($\Delta\mathbf{E}$) metric [CIE 78] and it is defined as the Euclidean distance between the original color pixel and its reproduction in S-CIELab color

metric space. It is widely accepted and used for measuring color reproduction error when a continuous-tone color image is reproduced with halftoning.

8.4.2 Simulation Results

Tables 8.1 and 8.2 show the performance of different algorithms in terms of the average of the $\Delta\mathbf{E}$ values of all pixels in the color quantization outputs. Table 8.1 shows the case of using color palettes obtained with median-cut algorithm [Heckbert 82] while Table 8.2 shows the case of using palettes obtained with octree algorithm [Gervautz 90]. The proposed algorithm is obviously superior to the others in both cases. In our simulation, the algorithm worked well with any tested palettes which were obtained with different palette generation algorithms.

Parallel processing is very attractive in many real-time multimedia applications to reduce processing time. In order to take its advantage, the proposed algorithm can be modified to be block-based as follows. The input image is first partitioned into a number of non-overlapped blocks of size 32×32 each. Then, the proposed algorithm is applied to these blocks to produce the halftoned color-quantized output blocks separately. The second last columns of Tables 8.1 and 8.2 show the performance of this block-based approach. One can see that its performance is a bit poorer than that the original proposed approach but still much better than all the others. Since this block-based approach can reduce the processing time considerably with parallel processing, sometimes it would be even more suitable than the original proposed algorithm to process an image especially in real-time applications.

Figures 8.1 and 8.2 show the halftoned color-quantized outputs of various algorithms when palettes of different sizes are used. The palettes were obtained with median-cut algorithm. It can be found that the proposed algorithm can eliminate the

directional hysteresis and reduce color impulses whereas these artefacts are highly visible in the outputs of Orchard's [Orchard 91] and Akarun's [Akarun 97] algorithms especially when the palette size is small. For example, in Figures 8.2b and 8.2c, in the sky, one can see ripple patterns that propagate from the top of the left to the middle of the right of the pictures. They are induced by directional hysteresis and composed of a number of ordered color impulses. The proposed algorithm does not generate directional hysteresis and it can significantly reduce color impulses as shown in Figure 8.2f. Özdemir's algorithm [Özdemir 00] introduces directional hysteresis as well but its effect is less visible as compared with Orchard's [Orchard 91] and Akarun's [Akarun 97] algorithms. However, one can still find the ripple patterns under the blue ball in Figure 8.1d. Besides, severe pattern noise can be found in Figures 8.1d (e.g. above the yellow ball) and 8.2d (e.g. the yellow cap and the sky). Noticeable false contour and color shift can be found in the outputs of Breaux's algorithm [Breaux 99]. In particular, they appear in the middle of Figures 8.1e and 8.2e.

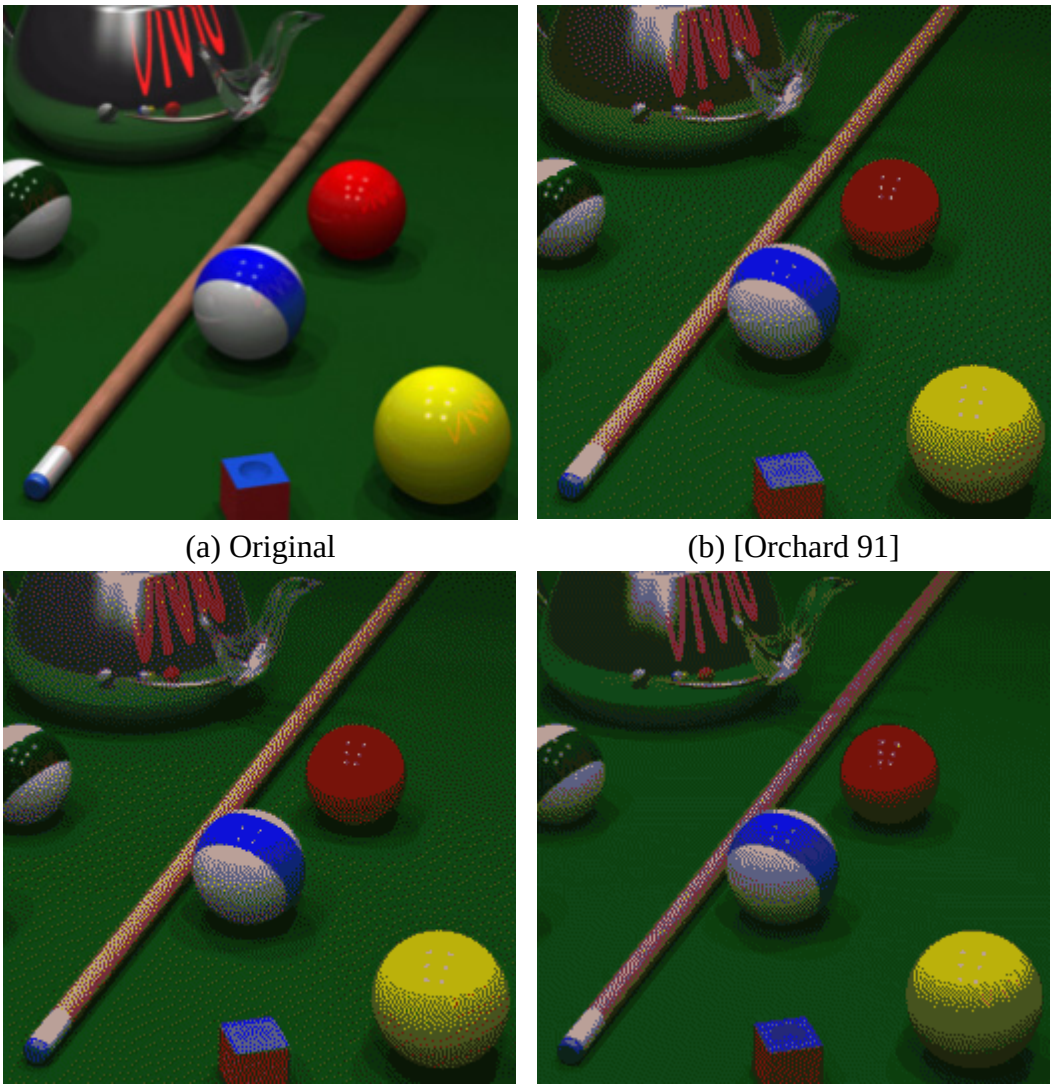
Figures 8.1g and 8.2g show the results of our proposed block-based approach. The quality of its outputs is similar to that of the original proposed algorithm. In theory, this block-based approach introduces blocking artefact. However, it is found that this blocking artefact is invisible in the simulation results. When processing time is the most concern and parallel processing is allowed, it would be more appropriate to use this block-based approach. Besides, this approach is helpful to reduce the size of the required processing buffer during color quantization.

		Average of S-CIELAB difference $\Delta\mathbf{E}$					
Palette Size		[Orchard 91]	[Akarun 97]	[Breaux 99]	[Özdemir 00]	Proposed (block-based)	Proposed
16	Lenna	28.932	28.691	27.349	29.786	26.103	26.234
	Baboon	38.544	37.492	38.602	40.300	38.253	38.265
	Fruits	48.323	47.141	49.759	51.069	47.633	47.846
	Cycles	38.573	37.385	37.458	40.845	36.453	36.560
	Airplane	23.251	21.944	21.913	30.349	21.437	21.452
	Caps	42.948	41.874	40.465	45.144	40.170	40.333
	Window	21.928	20.978	20.666	22.152	20.303	20.396
	Pool	33.662	34.552	31.617	32.476	30.436	30.423
	Average	**34.520**	**33.757**	**33.479**	**36.515**	**32.599**	**32.689**
32	Lenna	22.418	22.354	22.134	27.260	21.286	21.351
	Baboon	30.584	30.459	31.978	35.630	30.985	30.949
	Fruits	32.955	33.129	29.607	41.042	29.047	29.237
	Cycles	30.282	30.147	31.038	32.788	29.920	29.922
	Airplane	17.464	17.503	18.706	28.927	17.165	17.160
	Caps	34.332	34.280	31.172	34.704	30.765	30.860
	Window	15.346	15.296	15.188	17.815	14.777	14.922
	Pool	23.750	23.476	20.976	22.217	20.335	20.498
	Average	**25.891**	**25.831**	**25.100**	**30.048**	**24.285**	**24.362**
64	Lenna	18.666	18.558	17.740	20.493	17.264	17.277
	Baboon	26.978	26.711	27.779	33.028	26.021	26.092
	Fruits	23.086	22.974	20.538	33.233	20.282	20.361
	Cycles	22.976	22.783	23.941	27.456	22.792	22.772
	Airplane	14.251	14.155	13.99	18.000	13.229	13.193
	Caps	21.452	21.343	18.656	23.189	19.201	19.180
	Window	12.578	12.495	12.698	14.377	11.720	11.727
	Pool	16.581	16.396	15.311	15.889	15.002	15.009
	Average	**19.571**	**19.427**	**18.832**	**23.208**	**18.189**	**18.201**
128	Lenna	16.123	15.874	15.362	20.438	14.706	14.696
	Baboon	21.096	20.753	21.842	31.428	20.555	20.571
	Fruits	18.689	18.621	16.693	24.575	16.037	16.056
	Cycles	18.857	18.527	18.713	23.251	17.981	18.002
	Airplane	10.547	10.280	10.214	16.728	9.515	9.524
	Caps	16.490	16.334	14.616	20.761	14.317	14.353
	Window	10.726	10.561	10.816	14.894	10.291	10.270
	Pool	13.419	13.331	11.850	14.474	11.177	11.221
	Average	**15.743**	**15.535**	**15.013**	**20.819**	**14.322**	**14.337**

Table 8.1. Average of S-CIELab difference ($\Delta\mathbf{E}$) metric of the halftoned color-quantized outputs of various algorithms (Palettes were generated with median-cut algorithm [Heckbert 82].)

		Average of S-CIELAB difference $\Delta\mathbf{E}$					
Palette Size		[Orchard 91]	[Akarun 97]	[Breaux 99]	[Özdemir 00]	Proposed (block-based)	Proposed
16	Lenna	31.056	31.246	26.266	31.303	26.914	27.162
	Baboon	38.406	38.761	40.176	38.309	38.285	38.214
	Fruits	52.095	51.788	51.680	51.848	52.846	53.120
	Cycles	39.834	39.756	37.813	39.404	36.407	36.413
	Airplane	34.237	34.220	33.141	33.648	33.369	33.376
	Caps	38.501	38.231	37.469	38.454	38.820	38.648
	Window	19.442	19.569	18.304	18.646	18.292	18.328
	Pool	28.543	28.483	27.472	28.195	26.350	26.227
	Average	**35.264**	**35.257**	**34.040**	**34.976**	**33.910**	**33.936**
32	Lenna	24.650	24.802	20.687	22.387	20.965	21.088
	Baboon	35.748	35.267	35.049	35.421	33.011	32.998
	Fruits	33.829	33.821	31.837	33.835	31.580	31.724
	Cycles	31.433	31.364	30.298	31.449	28.948	28.930
	Airplane	24.687	24.748	25.933	24.169	25.418	25.338
	Caps	17.714	17.705	15.042	19.451	15.240	15.271
	Window	17.139	17.235	16.770	16.988	16.035	16.072
	Pool	27.426	27.390	26.516	27.121	25.422	25.259
	Average	**26.578**	**26.542**	**25.266**	**26.353**	**24.577**	**24.585**
64	Lenna	17.028	16.902	15.714	16.028	15.428	15.453
	Baboon	28.286	28.200	28.226	26.609	25.768	25.725
	Fruits	24.589	24.534	22.203	23.448	21.887	21.943
	Cycles	21.266	21.118	19.940	20.495	19.206	19.197
	Airplane	20.755	20.750	22.666	20.600	22.053	21.968
	Caps	14.031	14.069	11.342	17.491	11.715	11.779
	Window	11.429	11.455	10.401	10.573	9.678	9.675
	Pool	15.156	15.095	12.839	14.438	12.584	12.571
	Average	**19.068**	**19.016**	**17.917**	**18.710**	**17.290**	**17.290**
128	Lenna	14.063	13.773	12.895	12.504	12.429	12.428
	Baboon	22.298	21.910	22.619	20.826	20.576	20.579
	Fruits	18.012	18.002	16.834	17.307	15.769	15.840
	Cycles	18.381	18.279	17.895	17.443	16.971	16.982
	Airplane	13.936	13.773	13.148	23.092	13.289	13.396
	Caps	11.043	11.081	9.440	9.702	9.251	9.301
	Window	9.599	9.501	9.348	9.347	8.643	8.593
	Pool	13.172	13.076	10.650	11.517	10.485	10.517
	Average	**15.063**	**14.924**	**14.104**	**15.217**	**13.427**	**13.455**

Table 8.2. Average of S-CIELab difference ($\Delta\mathbf{E}$) metric of the halftoned color-quantized outputs of various algorithms (Palettes were generated with octree algorithm [Gervautz 90].)

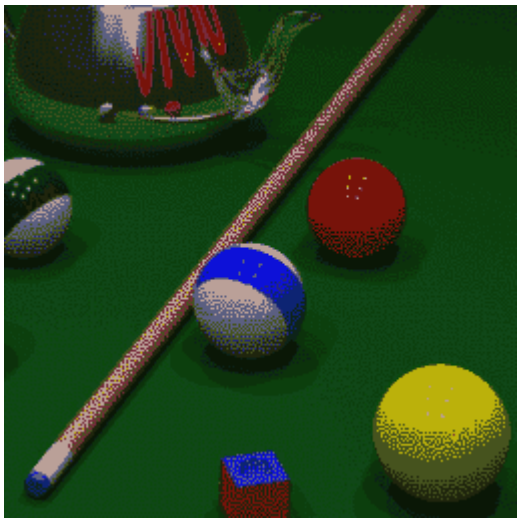


(a) Original (b) [Orchard 91]

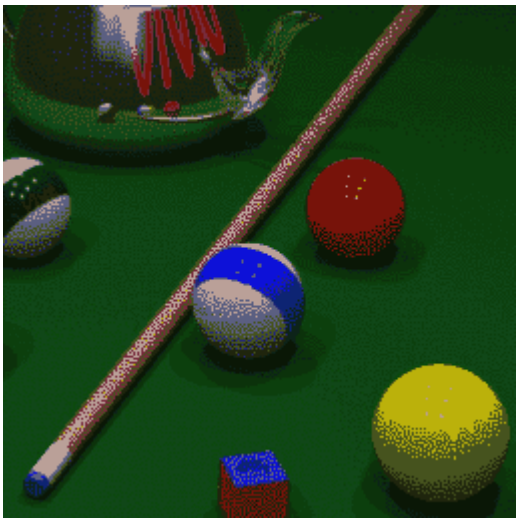
(c) [Akarun 97] (d) [Özdemir 00]

Figure 8.1. Color quantization results of “Pool” (Palette size = 16): (a) Original, (b) [Orchard 91], (c) [Akarun 97], (d) [Özdemir 00], (e) [Breaux 99], (f) the proposed algorithm and (f) the proposed block-based algorithm.

(e) [Breaux 99]

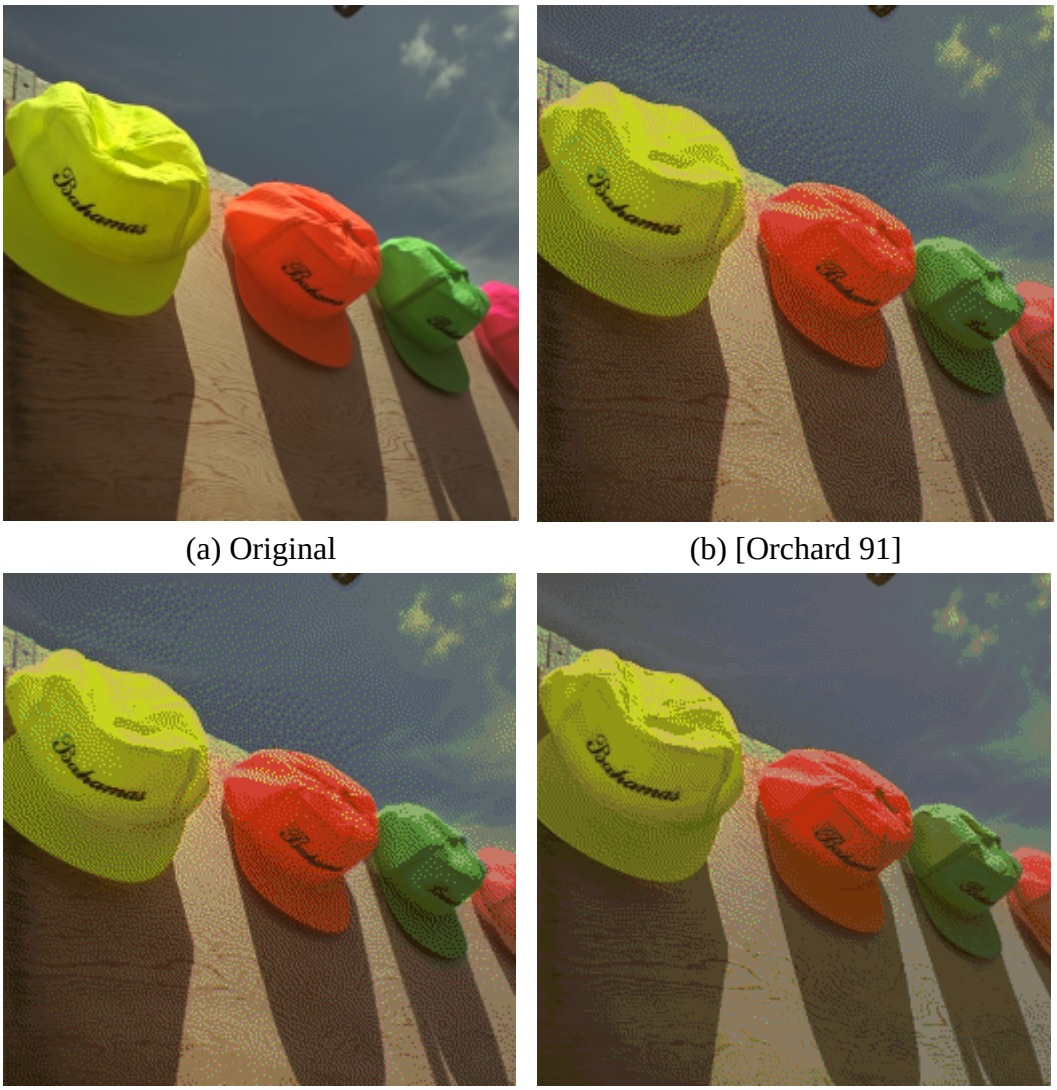


(f) Proposed



(g) Proposed (block-based)

Figure 8.1(continue). Color quantization results of “Pool” (Palette size = 16): (a) Original, (b) [Orchard 91], (c) [Akarun 97], (d) [Özdemir 00], (e) [Breaux 99], (f) the proposed algorithm and (f) the proposed block-based algorithm.



(a) Original (b) [Orchard 91]

(c) [Akarun 97] (d) [Özdemir 00]

Figure 8.2. Color quantization results of “Caps” (Palette size = 32): (a) Original, (b) [Orchard 91], (c) [Akarun 97], (d) [Özdemir 00], (e) [Breaux 99], (f) the proposed algorithm and (g) the proposed block-based algorithm

(e) [Breaux 99]

(f) Proposed

(g) Proposed (block-based)

Figure 8.2(continue). Color quantization results of “Caps” (Palette size = 32): (a) Original, (b) [Orchard 91], (c) [Akarun 97], (d) [Özdemir 00], (e) [Breaux 99], (f) the proposed algorithm and (g) the proposed block-based algorithm

8.5 Performance on Generating Scalable Halftoned Color-quantized Images

In this Section, performance evaluation of the proposed algorithm in generating scalable halftoned color-quantized image is given. The details of the realization of the proposed algorithm are first described in Section 8.5.1. Secondly, simulation results are given in Section 8.5.2.

8.5.1 Details of the Simulation

Simulation was carried out to evaluate the performance of the algorithm on a number of *de facto* standard 24-bit full color images. Each of them is of size 256×256. For each testing image, a set of color palettes of different size were generated with median-cut algorithm [Heckbert 82] for color quantization. The proposed algorithm was applied to all testing images to obtain their corresponding halftoned color quantization results with the generated color palettes to evaluate its performance. In its realization, parameter R was selected to be 4.

For comparison, halftoned color quantization results were also produced with some other color quantization algorithms [Orchard 91, Akarun 97, Özdemir 00] and then down-sampled to produce various downscaled versions. Unlike most color halftoning algorithms which are dedicated for printing applications [Damera-Venkata 03, Lau 00], these evaluated algorithms [Orchard 91, Akarun 97, Özdemir 00] are not straightforward extension of binary halftoning and are able to handle color quantization in which any arbitrary palettes can be used. Among them, Orchard's algorithm [Orchard 91] forms a common framework that most of these algorithms adopt. In its realization, Floyd-Steinberg filter [Floyd 76] was used in error diffusion.

Both Akarun's algorithm [Akarun 97] and Özdemir algorithm [Özdemir 00] adopt the framework presented in Orchard's algorithm. In particular, Akarun's algorithm [Akarun 97] uses an adaptive error diffusion filter to prevent texture contours, color impulses and color shift. Instead of the conventional Euclidean distance criterion, Özdemir's algorithm [Özdemir 00] uses a weighted sum of the distances among color vectors as a searching criterion in its color quantization process to prevent excess accumulation of quantization errors.

8.5.2 Simulation Results

Table 8.3 shows the performance of various algorithms in terms of the average S-CIELAB difference (ΔE) value of all pixels in their color quantization outputs and their corresponding down-sampled versions. The palettes used to obtain Table 8.3 are of size 16. Tables 8.4, 8.5 and 8.6, respectively, show the case when palettes of size 32, 64 and 128 were used. Simulation results show that the proposed algorithm can provide a better result than the other algorithms especially when downscaling is performed. Our proposed algorithm is obviously better and the superiority is very significant when the scaling factor is large.

Figures 8.3b-8.3e show the processing results of different evaluated algorithms. Figure 8.3a is the original 256×256 24-bit full-color image for reference. The palette used to generate Figures 8.3b-8.3e is of size 64 and was obtained with Figure 8.3a using median-cut algorithm. The processing results of the algorithms are more or less the same except that some false contour can be found in Figure 8.3d.

Figures 8.4 to 8.6 show the downscaled versions of Figure 8.3. The downscaling of Figures 8.3b-8.3e was carried out by simple down-sampling. Figures 8.4a, 8.5a and 8.6a show $\mathbf{I}^r$ for $r = 2$, 3 and 4 respectively and are used as references

for evaluating how close the downscaling outputs of the color-quantization results obtained with different algorithms to the downscaled original. The corresponding downscaling ratios used to produce Figures 8.4, 8.5 and 8.6 were, respectively, 2, 4 and 8. Accordingly, they are of size 128×128, 64×64 and 32×32 respectively. For easier inspection, these Figures are zoomed with nearest-neighbor interpolation to make their size as large as the original full-scaled version. One can see that the downscaled versions of Figure 8.3e (Figures 8.4e, 8.5e and 8.6e) can faithfully report the content of the downscaled versions of Figure 8.3a while the others cannot.

Unlike the other algorithms [Orchard 91, Akarun 97, Özdemir 00] in which causal diffusion filters and predefined processing sequences are used in the error diffusion process, the proposed algorithm color quantizes pixels in a so-called ‘maximum energy guidance’ manner and diffuses the quantization errors with a non-causal diffusion filter. This approach completely removes the artefacts caused by directional hysteresis.

	Average of S-CIELAB difference $\Delta\mathbf{E}$							
	[Orchard 91]	[Akarun 97]	[Özdemir 00]	Proposed	[Orchard 91]	[Akarun 97]	[Özdemir 00]	Proposed
	Full-scaled (256x256)*				Down-sampled versions s_r =2 (128x128)*			
Lenna	28.9316	28.6911	29.7862	27.4111	30.6230	30.3740	31.3529	27.3935
Baboon	38.5437	37.4918	40.3003	39.8194	40.8781	40.1357	43.1127	39.4561
Peppers	40.7377	41.2408	39.6435	40.3174	42.6166	43.7538	41.9237	40.3149
Fruits	48.3228	47.1405	51.0687	48.3742	49.5522	48.6611	52.3894	48.0071
Cycles	38.5730	37.3853	40.8454	37.3234	40.5470	39.6401	42.6159	36.9831
Airplane	23.2511	21.9437	30.3493	21.9443	24.5735	23.2945	31.4577	21.7703
Parrots	36.9315	36.4938	42.0641	35.2689	37.5506	37.3442	42.5235	35.4676
Caps	42.9482	41.8739	45.1439	40.6307	43.5955	43.0314	46.1908	40.4780
Windows	21.9278	20.9777	22.1515	20.6720	22.9303	21.7949	22.3660	20.1365
Pool	33.6615	34.5516	32.4756	30.9726	34.1458	36.1527	32.3452	30.8626
Average	**35.3829**	**34.7790**	**37.3829**	**34.2734**	**36.7013**	**36.4182**	**38.6278**	**34.0870**
	Down-sampled versions s_r =3 (64x64)*				Down-sampled versions s_r =4 (32x32)*			
Lenna	34.7154	34.0515	34.3766	27.1739	41.1666	41.2214	39.9279	24.7768
Baboon	43.4302	43.0038	45.1606	38.8943	48.4784	47.1034	49.1323	37.3437
Peppers	47.2924	48.0682	46.3931	39.9708	55.8807	56.7514	55.0129	37.1602
Fruits	51.0632	49.8831	53.4490	47.0863	54.7632	52.8876	57.1326	44.0221
Cycles	43.2724	43.3180	43.8239	35.9865	47.4550	47.3311	47.0905	33.2558
Airplane	25.7758	25.1903	31.7653	21.4674	27.3820	27.4064	31.2663	19.9550
Parrots	39.0093	38.7003	43.0514	35.4608	43.8259	41.9619	46.7252	34.0159
Caps	43.4285	42.4930	46.8236	39.9573	45.6638	45.5602	49.4209	39.7279
Windows	24.4389	23.3427	23.7348	20.1155	26.2632	24.1583	25.4539	19.1232
Pool	35.7304	38.1435	32.9412	30.5566	38.4069	40.2626	34.7212	29.7837
Average	**38.8157**	**38.6194**	**40.1519**	**33.6669**	**42.9286**	**42.4644**	**43.5884**	**31.9164**

Table 8.3. Average of S-CIELAB color difference ($\Delta\mathbf{E}$) metric of the halftoned color-quantized outputs of various algorithms with s_r = 1,2,3 and 4. (Palettes were generated with median-cut algorithm [Heckbert 82] of size 16.)

	Average of S-CIELAB difference $\Delta\mathbf{E}$							
	[Orchard 91]	[Akarun 97]	[Özdemir 00]	Proposed	[Orchard 91]	[Akarun 97]	[Özdemir 00]	Proposed
	Full-scaled (256x256)*				Down-sampled versions s_r =2 (128x128)*			
Lenna	22.4175	22.3544	27.2595	22.5245	24.2977	24.2976	28.6136	22.8641
Baboon	30.5843	30.4593	35.6299	32.3477	34.3662	33.9958	38.6649	31.6702
Peppers	31.6980	31.7508	31.8524	31.2796	34.6418	34.7815	34.0940	31.6536
Fruits	32.9554	33.1293	41.0423	30.1852	35.3753	35.4104	42.1195	30.0504
Cycles	30.2819	30.1468	32.7884	30.6668	32.7247	32.8151	34.5624	30.3137
Airplane	17.4641	17.5028	28.9268	17.6052	19.2816	19.2016	29.5116	17.5777
Parrots	29.6796	30.0374	37.5147	30.5404	30.7102	31.1610	37.6660	30.5348
Caps	34.3318	34.2797	34.7037	31.0265	35.2753	35.3314	35.2457	31.0604
Windows	15.3460	15.2962	17.8146	15.4933	16.5484	16.5228	18.3259	15.5339
Pool	23.7504	23.4761	22.2170	21.0557	25.2317	24.8715	22.7847	21.1863
Average	**26.8509**	**26.8433**	**30.9749**	**26.2725**	**28.8453**	**28.8389**	**32.1588**	**26.2445**
	Down-sampled versions s_r =3 (64x64)*				Down-sampled versions s_r =4 (32x32)*			
Lenna	29.3368	29.2217	32.3329	23.2344	36.0307	36.1518	37.9147	21.3391
Baboon	37.7552	37.5718	41.4138	31.4499	42.7903	43.4498	44.4923	29.7858
Peppers	40.9603	41.0161	39.3051	31.8327	51.2370	51.3724	47.5772	28.5411
Fruits	38.5338	39.1456	44.7961	30.0118	44.7680	46.3067	49.5972	28.0543
Cycles	37.1760	37.4183	37.7353	29.1946	42.6326	42.3749	42.0244	25.7103
Airplane	21.3561	21.1215	29.8527	17.4254	24.2762	24.0501	30.4577	16.4926
Parrots	32.7922	33.3502	38.5964	30.5012	38.4354	38.9822	41.7222	28.6721
Caps	36.3559	36.1046	35.7409	31.1569	38.5637	38.8126	37.7851	30.3367
Windows	19.2987	19.0378	20.2223	15.5761	22.7904	22.2913	22.8106	15.0503
Pool	27.7247	27.4291	24.2689	21.3982	32.5475	32.0929	28.4391	20.1785
Average	**32.1290**	**32.1417**	**34.4264**	**26.1781**	**37.4072**	**37.5885**	**38.2821**	**24.4161**

Table 8.4. Average of S-CIELAB color difference ($\Delta\mathbf{E}$) metric of the halftoned color-quantized outputs of various algorithms with s_r = 1,2,3 and 4. (Palettes were generated with median-cut algorithm [Heckbert 82] of size 32.)

	Average of S-CIELAB difference $\Delta\mathbf{E}$							
	[Orchard 91]	[Akarun 97]	[Özdemir 00]	Proposed	[Orchard 91]	[Akarun 97]	[Özdemir 00]	Proposed
	Full-scaled (256x256)*				Down-sampled versions s_r =2 (128x128)*			
Lenna	18.6657	18.5575	20.4925	18.6208	20.9581	20.7883	21.8426	19.0129
Baboon	26.9775	26.7113	33.0281	27.8488	32.0848	31.8978	36.7743	27.4917
Peppers	23.4616	23.3081	29.3473	23.4029	26.8182	26.5622	31.0551	24.0759
Fruits	23.0858	22.9736	33.2330	21.4357	25.5479	25.5183	34.6237	21.6159
Cycles	22.9758	22.7825	27.4555	23.5371	26.5774	26.4171	29.3368	23.3967
Airplane	14.2509	14.1547	18.0002	13.9584	15.9356	15.8684	19.1846	14.0700
Parrots	20.5500	20.4900	30.0457	20.9540	21.8649	22.0246	30.5253	20.9501
Caps	21.4520	21.3434	23.1885	19.5369	22.4616	22.5172	23.4974	19.5127
Windows	12.5780	12.4953	14.3768	12.4579	14.3018	14.2672	14.9191	12.6608
Pool	16.5812	16.3962	15.8892	15.6198	18.0645	18.0366	16.7088	15.8052
Average	**20.0578**	**19.9213**	**24.5057**	**19.7372**	**22.4615**	**22.3898**	**25.8468**	**19.8592**
	Down-sampled versions s_r =3 (64x64)*				Down-sampled versions s_r =4 (32x32)*			
Lenna	25.9064	25.6385	26.0224	18.9407	33.3853	32.8163	32.5465	17.4808
Baboon	36.3970	36.4102	39.5776	27.0497	41.7126	40.9524	42.0561	25.8955
Peppers	34.1833	33.7415	35.9509	24.6303	46.2606	45.8069	44.4703	22.8059
Fruits	29.6034	29.5295	37.0887	21.6121	36.9739	36.8846	41.8582	19.6450
Cycles	32.2630	32.3674	32.6888	23.3430	39.0878	39.5178	37.2943	21.3194
Airplane	18.8005	18.3783	20.8280	14.1276	23.0214	22.5154	23.2848	13.4145
Parrots	24.5321	25.1225	31.8742	20.9267	31.5233	32.5357	36.9606	19.5793
Caps	24.5855	24.0761	24.5514	19.9253	29.5883	28.3059	27.8746	18.9590
Windows	17.8718	17.5988	17.4789	12.8881	22.1165	21.8629	20.5554	11.8399
Pool	21.4229	21.3180	19.3507	16.0724	28.0546	28.5135	24.8562	14.9362
Average	**26.5566**	**26.4181**	**28.5412**	**19.9516**	**33.1724**	**32.9711**	**33.1757**	**18.5876**

Table 8.5. Average of S-CIELAB color difference ($\Delta\mathbf{E}$) metric of the halftoned color-quantized outputs of various algorithms with s_r = 1,2,3 and 4. (Palettes were generated with median-cut algorithm [Heckbert 82] of size 64.)

	Average of S-CIELAB difference $\Delta\mathbf{E}$							
	[Orchard 91]	[Akarun 97]	[Özdemir 00]	Proposed	[Orchard 91]	[Akarun 97]	[Özdemir 00]	Proposed
	Full-scaled (256x256)*				Down-sampled versions s_r =2 (128x128)*			
Lenna	16.1225	15.8735	20.4377	16.3105	19.1181	19.0161	22.0774	16.7176
Baboon	21.0958	20.7530	31.4279	22.5380	26.9982	26.8287	34.7378	22.1804
Peppers	19.9866	19.8180	32.1305	20.8132	23.5107	23.3576	33.0290	21.6769
Fruits	18.6890	18.6213	24.5754	17.3134	21.8485	21.8378	26.2477	17.8355
Cycles	18.8566	18.5267	23.2508	19.1023	22.7989	22.3365	25.1983	19.3348
Airplane	10.5471	10.2797	16.7275	10.3404	12.4678	12.2743	17.5030	10.6316
Parrots	16.7393	16.6889	25.0309	16.5420	18.1145	18.1686	25.1460	16.6164
Caps	16.4898	16.3344	20.7611	14.7842	17.5491	17.5269	21.0446	14.8646
Windows	10.7261	10.5613	14.8944	11.0626	12.8276	12.5645	15.2334	11.3159
Pool	13.4193	13.3309	14.4740	12.0818	15.3282	15.1016	15.2422	12.6234
Average	**16.2672**	**16.0788**	**22.3710**	**16.0888**	**19.0562**	**18.9013**	**23.5459**	**16.3797**
	Down-sampled versions s_r =3 (64x64)*				Down-sampled versions s_r =4 (32x32)*			
Lenna	24.8135	24.8190	26.1226	16.8708	32.7471	32.3377	32.8690	14.7253
Baboon	32.1874	32.2442	37.2683	21.9321	38.1254	38.0333	38.8644	20.9293
Peppers	31.6937	31.2189	37.2750	22.5465	45.3312	44.2084	44.5471	20.7955
Fruits	26.6113	26.5632	30.1573	17.8837	34.2962	34.2082	35.7380	16.0017
Cycles	29.0977	28.3924	28.6367	19.5093	36.3676	36.3883	33.9957	17.4611
Airplane	15.8925	15.7843	19.1479	10.7797	20.4903	20.3739	20.6309	9.6819
Parrots	21.3384	21.6091	26.8414	16.7017	28.4869	28.5532	31.2748	15.9072
Caps	19.7349	19.4825	22.2714	15.1887	24.7237	23.2565	25.0248	14.6449
Windows	16.6058	16.4387	17.3283	11.3712	21.1030	21.1645	19.7571	10.5326
Pool	19.5703	19.2888	18.4424	13.2590	27.0574	27.1062	24.0921	11.9816
Average	**23.7546**	**23.5841**	**26.3491**	**16.6043**	**30.8729**	**30.5630**	**30.6794**	**15.2661**

Table 8.6. Average of S-CIELAB color difference ($\Delta\mathbf{E}$) metric of the halftoned color-quantized outputs of various algorithms with s_r = 1,2,3 and 4. (Palettes were generated with median-cut algorithm [Heckbert 82] of size 128.)

(a) Original

(b) [Orchard 91]

(c) [Özdemir 00]

(d) [Akarun 97]

(e) Proposed

Figure 8.3. Color quantization results of full-scaled “Caps” (palette size = 64): (a) Original (b) [Orchard 91], (c) [Özdemir 00], (d) [Akarun 97], (e) proposed algorithm

(a) Original

(b) [Orchard 91]

(c) [Özdemir 00]

(d) [Akarun 97]

(e) Proposed

Figure 8.4. Color quantization results of down-sampled versions of "Caps" with s_r =2 (Palettes were generated with median-cut algorithm [Heckbert 82] of size 64): (a) Original (b) [Orchard 91], (c) [Özdemir 00], (d) [Akarun 97], (e) proposed algorithm

(a) Original

(b) [Orchard 91]

(c) [Özdemir 00]

(d) [Akarun 97]

(e) Proposed

Figure 8.5. Color quantization results of down-sampled versions of "Caps" with s_r =3 (Palettes were generated with median-cut algorithm [Heckbert 82] of size 64): (a) Original (b) [Orchard 91], (c) [Özdemir 00], (d) [Akarun 97], (e) proposed algorithm

(a) Original

(b) [Orchard 91]

(c) [Özdemir 00]

(d) [Akarun 97]

(e) Proposed

Figure 8.6. Color quantization results of down-sampled versions of "Caps" with s_r =4 (Palettes were generated with median-cut algorithm [Heckbert 82] of size 64): (a) Original (b) [Orchard 91], (c) [Özdemir 00], (d) [Akarun 97], (e) proposed algorithm

8.6 Summary

In this Chapter, a multiscale color error-diffusion algorithm for color quantization is proposed. It makes use of the multiscale technique to perform color quantization in YIQ color space with error diffusion using a non-casual filter. This eliminates directional hysteresis and reduces color impulses. Simulation results demonstrated that the proposed algorithm can achieve a remarkable improvement in the quality of halftoned color-quantized images both subjectively and objectively in term of S-CIELAB color difference ($\Delta\mathbf{E}$) metric [Zhang 96] as compared with the other available algorithms.

The proposed algorithm can be modified to process an image in a block-based approach to save processing time when parallel processing is allowed. Simulation results show that the blocking artefact is invisible in the output of this block-based algorithm and its performance is comparable to that of the original proposed algorithm, which makes it suitable for real-time applications.

It is always useful to produce scalable color-indexed images for delivering media information to diverse clients over heterogeneous networks reliably and efficiently. The proposed multiscale error diffusion framework for color quantization is extended to produce scalable color-indexed images. For any given image, this algorithm can produce a high-quality directional-hysteresis free output and simultaneously embed a set of color quantization results of the downscaled versions of the given image without any memory overhead. With the color-quantization output of the proposed algorithm, Images of desirable resolutions can be extracted by simply down-sampling the output. The proposed algorithm works with any arbitrary color palettes of different size.

Chapter 9.

Conclusions

9.1 The Work Done

Color quantization reduces the number of colors used to represent an image [Heckbert 82] and has been widely used in a number of applications such as displaying a color image on a low-end display, delivering images over the Internet and producing hardcopies of color images. In this thesis, the main thrust of our present works is the investigation on the following areas where various problems are remained unsolved, not properly solved, or not effectively solved.

(1) Restoration of color-quantized images in which no error diffusion is involved

(2) Restoration of color-quantized images in which error diffusion is involved

(3) Production of scalable color prints

(4) Formation of color-quantized image and scalable color-indexed images.

In Chapters 3 to 6, four restoration algorithms are proposed to tackle the problem of the restoration of color-quantized images. Among them, the two proposed in Chapter 3 and 4 are dedicated for restoring color-quantized images in which error diffusion is not involved. One of them is regularization-based and the other is POCS-

based. These algorithms make a good use of the palette to derive useful *a priori* information and then uses the *a priori* information to formulate the restoration algorithms. Simulation results showed that the algorithms could provide good restoration performance in terms of SNRI and CIELAB color difference metrics [CIE 78].

Obviously, restoring a halftoned color-quantized image is different from restoring a color-quantized image in which error diffusion is not involved. It is because the degradation models of these two cases are different. Accordingly, dedicated restoration algorithms should be developed to handle the problem of restoring halftoned color-quantized image. Chapters 5 and 6 presented another two restoration algorithms to tackle this problem. One of them is based on POCS theory [Youla 82] while the other is based on simulated annealing [Kirkpatrick 83]. These algorithms make a good use of the involved color palette and the error diffusion process to derive useful *a priori* information to formulate the algorithms. Simulation results showed that the two restoration algorithms can handle halftoned color-quantized images well and provide good restoration performance in terms of both SNRI and CIELAB color difference metrics.

When one renders halftone images for different printers of different resolutions, it is desirable to make the output scalable in a way that it embeds low resolution halftone images into a full-scale halftone image, and, through a very simple procedure such as down-sampling, the low resolution halftone images can be obtained from the high resolution halftone image directly. Theoretically, this can be handled by constrained halftoning. We found that, due to their frame based nature, multiscale error diffusion algorithms could be modified to realize constrained halftoning easily and effectively. In Chapter 7, we extend the idea of multiscale error diffusion to

propose an algorithm for generating scalable color halftones for printing applications. Unlike those conventional error diffusion algorithms [Floyd 76, Wong 96], the proposed algorithm can take care of the constrained pixel very well before handling the unconstrained pixels. This allows one to compensate for the disturbance caused by the mismatch in the dot assignment of constrained pixels when handling the unconstrained pixels. Also, discussions of some critical factors for achieving good constrained halftoning results are also provided. Analysis and simulation results showed that good halftoning results could be provided by the proposed algorithm.

In Chapter 8, a multiscale vector error diffusion framework is proposed to produce halftoned color-quantized image. Unlike those color halftoning algorithms for printing purpose, the proposed algorithm does not handle color planes separately and is able to handle color quantization using any arbitrary palette. Furthermore, the proposed algorithm can completely eliminate directional hysteresis and reduce color impulses as compared with the other algorithms. In order to take the advantage of parallel processing, the proposed algorithm was modified to process an image in a block-based approach. Theoretically, blocking artefact will occur in any block-based approach, but simulation results showed that this blocking artefact is invisible. Furthermore, the proposed algorithm can be used as a framework for generating scalable color-indexed images and a multiscale multiresolution color error diffusion algorithm is proposed accordingly. The algorithm can produce a high quality directional hysteresis free output and simultaneously embed a set of color quantization results of the downscaled versions.

On the whole, we have developed a set of image processing techniques for restoration and formation of color-quantized images. The proposed techniques can accomplish significant improvement over many of the existing techniques.

9.2 Future Works

The work presented in this thesis can be extended in different aspects. First of all, based on the proposed restoration algorithms developed in Chapters 3 to 6, we believe that an investigation on their applications in the area of image coding would be promising. In most current image coding systems, the compression stage and the restoration stage operate separately. Accordingly, the two processes cannot make use of each other to optimize the overall coding performance. It would be advantageous to take the contribution of the restoration to be performed in the decompression process into account during image compression. The compression and the restoration can then be jointly optimized by adjusting the parameters of the encoder according to the difference between the original and the compressed image.

Various halftoning algorithms are proposed in this thesis to generate scalable color halftones. The statistical characteristics of their outputs are different from that of the halftones generated with conventional algorithms. It would be interesting to investigate if one can encode these scalable halftones more efficiently by making use of the correlation of halftones in different resolutions.

Besides embedding lower resolution halftones into the full-scale halftone, the multiscale multiresolution error diffusion algorithm proposed in Chapter 7 can also be extended to embed a pattern into a halftone such that watermarking can be achieved. A further investigation on this extension is valuable.

The degradation models of color quantization algorithms proposed in Chapters 7 and 8 are completely different from the conventional algorithms. There should be some dedicated restoration algorithms for their outputs. In short, there is ample scope for much challenging work in these areas.

Appendix A (Testing Images)

Lenna

Baboon

Fruits

Peppers

Couple

Girl

Airplane

Boat

Tiffany

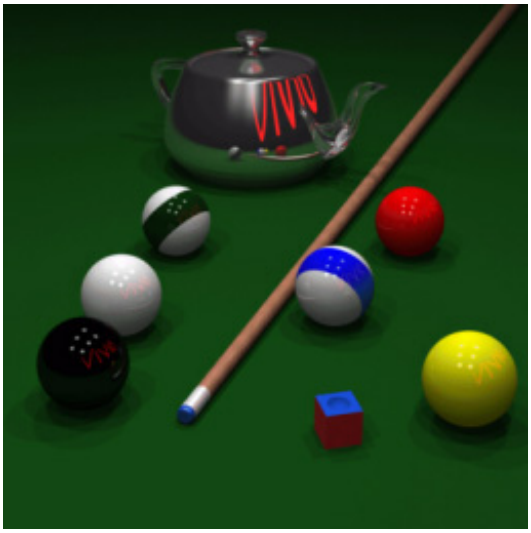
Cycles

Caps

Parrots

Melon

Windows



Pool

Bibliography

[Akarun 97] L. Akarun, Y. Yardımcı, and A.E. Çetin, “Adaptive Methods for Dithering of Color Images,” IEEE Transaction On Image Processing, Vol. 6, No. 7, Jul 1997, pp.950-955.

[Akarun 96] L. Akarun, D. Özdemir and Ö. Yalçın, “A modified quantisation algorithm for dithering of colour images,” Electronics Letters, Vol. 32, No. 13, Jun 1996, pp.1185-1186.

[Altunbasak 01] H. Altunbasak, and H. J. Trussell, “Colorimetric restoration of digital images,” IEEE Transaction On Image Processing, Vol. 10, No. 3, March 2001, pp.393–402.

[Andrews 77] H. C. Andrews and B. R. Hunt. Digital Image Restoration. Prentice Hall, Englewood Cliffs, HN, 1977.

[Angelopoulos 94] G. Angelopoulos and I. Pitas, “Multichannel Wiener filters in color image restoration,” IEEE Transaction On Circuits and Systems for Video Technology, Vol. 4, No. 1, Feb 1994, pp. 83–87.

[Banham 97] M. R. Banham and A. K. Katsaggelos, “Digital image restoration,” IEEE Signal Processing Magazine, Vol. 14, Mar 1997, pp.24-41.

[Barni 00] M. Barni, V. Cappellini and L. Mirri, “Multichannel m-filtering for color image restoration,” Proc. IEEE ICIP’2000, Vol. 1, 2000, pp. 529–532.

[Biemond 90] J. Biemond, R. L. Lagendijk and R. M. Mersereau, Iterative methods for image deblurring, Proc. IEEE Vol. 78, No. 5, May 1990, pp.856 – 883.

[Bregman 65] L. M. Bregman, "The method of successive projections for finding a common point of convex sets," Soviet Math. Doklady, 6:688-692, 1965.

[Breaux 99] N. Breaux and C. H. H. Chu, "Halftoning for Colour-Indexed Displays," in Proc, ICIP, Oct 1999, pp.597-601.

[Chan 98] Y. H. Chan, "A modified multiscale error diffusion technique for digital halftoning," IEEE Signal Processing Letters, Vol. 5, No. 11, Nov 1998, pp.277-280.

[Chan 04] Y. H. Chan and S. M. Cheung, "Feature-preserving multiscale error diffusion for digital halftoning," Journal of Electronic Imaging, Vol. 13, No. 3, Jul 2004, pp.639-645.

[Chan 05] Y.H. Chan and Y.H. Fung, "A regularized constrained iterative restoration algorithm for restoring color-quantized images," *Signal Processing*, Vol.85, No.7, Jul 2005, pp.375-1387.

[Chang 01] P. C. Chang, C. S. Yu and T. H. Lee, "Hybrid LMS-MMSE inverse halftoning technique," IEEE Trans On Image Processing, Vol. 10, No. 1, Jan 2001, pp.95 – 103.

[CIE 78] C.I.E. Recommendations on uniform color spaces, color difference equations, psychometric color terms, Supplement No. 2 to CIE Publication No.15 (E.-1.3.1), 1971/(TC- 1.3.), (1978).

[Connolly 95] C. Connolly, T.W.W. Leung, J. Nobbs, Colour measurement by video camera, J. Soc. Dyers Colour 111, 1995, pp.373 – 375.

[Damera-Venkata 03] N. Damera-Venkata, B. L. Evans and V. Monga, “Colour Error Diffusion Halftoning,” IEEE Signal Processing Magazine, Vol. 20, No. 4, Jul 2003, pp.51-58.

[Dines 77] K. A. Dines and A. C. Kak, “Constrained least squares filtering,” IEEE Transaction On Acoustics, Speech and Signal Processing, Vol. 25, No. 4, Aug 1977, pp.346-350.

[Floyd 76] R. W. Floyd and L. Steinberg, “An Adaptive Algorithm for spatial Greyscale,” Proceedings of the society for Information Display, Vol. 17, No. 2, 1976, pp.75-77.

[Fung 04a] Y. H. Fung and Y. H. Chan, “POCS-based algorithm for restoring colour-quantised images,” IEE Proceedings-Vision, Image and Signal Processing, Vol. 151, No. 2, Apr 2004, pp.119-127.

[Fung 04b] Y. H. Fung and Y. H. Chan, "Restoration of halftoned color-quantized images using projection onto convex sets," Proc. IEEE ICIP’04, Singapore, Oct 24-27, 2004, pp.325-328.

[Galatsanos 89] N. P. Galatsanos and R. T. Chin, “Digital restoration of multichannel images,” IEEE Transaction On Acoustic, Speech and Signal Processing, Vol. 37, No. 3, 1989, pp.415–421.

[Galatsanos 91a] N. P. Galatsanos, A. K. Katsaggelos, R. T. Chin and A. D. Hillery, “Least squares restoration of multichannel images,” IEEE Transaction On Signal Processing, Vol.39, No. 10, 1991, pp.2222–2236.

[Galatsanos 91b] N. P. Galatsanos and R. T. Chin, “Restoration of color images by multichannel Kalman filtering,” IEEE Transaction On Signal Processing, Vol. 39, No. 10, 1991, pp.2237–2252.

[Gentile 90] R. S. Gentile, E. Walowit, and J. P. Allebach, “Quantization and multi-level halftoning of color images for near original image quality,” Proc. SPIE, Vol. 1249, 1990, pp.249–259.

[Geman 84] S. Geman and D. Geman, “Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images,” IEEE Transaction On Pattern Analysis. Machine Intelligence, Vol. 6, No. 6, Nov 1984, pp.721-741.

[Gersho 90] A. Gersho, R. M. Gray, “Vector Quantization and Signal Compression,” Kluwer Academic Press, Dordrecht, MA, 1990.

[Gervautz 90] M. Gervautz, W. Purgathofer, “A simple method for color quantization: octree quantization,” Graphics Gems, Academic Press, New York, 1990, pp.287 – 293.

[Gilyazov 00] S. F. Gilyazov, N. L. Gol ʻdman, “Regularization of Ill-posed Problems by Iteration Methods,” Kluwer Academic Publishers, Dordrecht, 2000.

[Heckbert 82] P. Heckbert, “Color image quantization for frame buffer display,” Computer. Graph. Vol. 16, No. 3, 1982, pp.297 – 307.

[Hein 95] S. Hein and A. Zakhor, “Halftone to Continuous-Tone Conversion of Error-Diffusion Coded Images,” IEEE Transaction On Image Processing, Vol. 4, Feb 1995, pp.208-216.

[Hunt 84] B. R. Hunt and O. Kubler, “Karhunen-Loeve multispectral image restoration, Part 1: Theory,” IEEE Transaction On Acoustics, Speech and Signal Processing, Vol. 32, No. 3, 1984, pp.592–600.

[Hunt 77] B. R. Hunt, “Bayesian methods in nonlinear digital image restoration,” IEEE Transaction On Computers, Vol. 26, No. 3, Mar 1977, pp.219-229.

[Hunt 73] B. R. Hunt, “The application of constrained least squares estimation to image restoration by digital computer,” IEEE Transaction On Computers, Vol. 22, No. 9, 1973, pp.805-812.

[Ingber 93] L. Ingber, “Simulated annealing: practice versus theory”, Mathl. Comput. Modelling 18, 11, 1993, pp.29-57.

[Jarvis 76] J. F. Jarvis, C. N. Judice and W. H. Ninke, “A Survey of Techniques for the Display of Continuous Tone Pictures on Bi-level Displays,” Computer Graphics and Image Processing, 5, 1976, pp.13-40.

[Jolliffe 86] I. T. Jolliffe, Principal Components Analysis, Springer, New York, 1986.

[Katsaggelos 89] A. K. Katsaggelos, “Iterative image restoration algorithms,” Optical Engineering, Vol. 28, No. 7, 1989, pp.735-748.

[Katsavounidis 97] I. Katsavounidis and C. C. J. Kuo, “A multiscale error diffusion technique for digital halftoning,” IEEE Transaction On Image processing, Vol. 6, No. 3, 1997, pp.483-490.

[Kaulgud 99] N. Kaulgud, U. B. Desai, “Restoration of color images subjected to interchannel blurring,” Proceedings of the IEEE ISCAS’ Vol. 99, No. 4, 1999, pp.72 – 75.

[Kim 01] D. S. Kim, S. H. Park, “Postprocessing for vector-quantized images based on projection onto hypercubes,” IEEE Transaction On Circuits and Systems for Video Technology, Vol. 11, No. 7, Jul 2001, pp.802-814.

[Kirkpatrick 83] S. Kirkpatrick, C. D. Gelatt and M. P. Vecchi, “Optimation by Simulated Annealing”, *Science*, 220, 1983, pp.671-680.

[Kite 00] T. D. Kite, N. Damera-Venkata, B. L. Evans, and A. C. Bovik, “A Fast, High-Quality Inverse Halftoning Algorithm for Error Diffused Halftones,” IEEE Transactions On Image Processing, Vol. 9, No. 9, Sep 2000, pp.1583-1592.

[Kolpatzik 92] B. W. Kolpatzik and C. A. Bouman, “Optimized error diffusion for image display,” Journal of Electronics Imaging, Vol. 1, Jul 1992, pp.277-292.

[Lau 01] D. L. Lau and G. R. Arce, “Modern Digital Halftoning,” Marcel Dekker, Inc., 2001.

[Lau 00] D. L. Lau, G. R. Arce and N. C. Gallagher, “Digital colour halftoning with generalized error diffusion and multichannel green-noise masks,” IEEE Transaction On Image Processing, Vol. 9, No. 5, May 2000, pp.923-935.

[Mese 01] M. Mese and P. P. Vaidyanathan, “Look-up table (LUT) method for inverse halftoning,” IEEE Transaction On Image Processing, Vol. 10, No. 10, Oct 2001, pp.1566-1578.

[Miller 70] K. Miller, “Least-squares method for ill-posed problems with a prescribed bound,” SIAMJ. Math. Anal., 1, 1970, pp.52-74.

[Orchard 91] M. T. Orchard and C. A. Bouman, “Color quantization of images,” IEEE Transaction On Signal Processing, Vol. 39, No. 12, Dec 1991, pp.2677-2690.

[Özdemir 00] D. Özdemir and L. Akarun, “Fuzzy Error Diffusion”, IEEE Trans. On Image Processing, Vol. 9, No. 4, Apr 2000, pp.683-690.

[Özdemir 01] D. Özdemir and L. Akarun, “Fuzzy Algorithms for combined quantization and dithering,” IEEE Transaction On Image Processing, Vol. 10, No. 6, Jun 2001, pp. 923-931.

[Peli 91] E. Peli, “Multiresolution error convergence halftone algorithm,” Journal of the Optical Society of America A, Vol. 8, No. 4, 1991, pp.625-633.

[Puzicha 00] J. Puzicha, M. Held, J. Ketterer, J. M. Buhmann and D. W. Fellner, “On spatial quantisation of colour images,” IEEE Transaction On Image Processing, Vol. 9, No. 4, Apr 2000, pp.666-682.

[Schafer 81] R. W. Schafer, R. M. Merserau, M. A. Richards, “Constrained iterative restoration algorithms,” Proc. IEEE, Vol. 69, No. 4, 1981, pp.432 – 450.

[Scheunders 98] P. Scheunders, “Joint quantisation and error-diffusion of colour images using competitive learning,” IEE Proceedings, Vision, Image and Signal Processing, Vol. 145, No. 2, 1998, pp.137-140.

[Sezan 90] M. I. Sezan and A. M. Teklap, “Adaptive image restoration with artefact suppression using the theory of convex projections,” IEEE Transaction On Acoustics, Speech, and Signal Processing, Vol. 38. No. 1, 1990, pp.81-185.

[Sezan 82] M. I. Sezan and H. Stark, “Image restoration by the method of convex projections: Part 2 – applications and numerical results,” IEEE Transaction On Medical Imaging, Vol. 1, No. 2, 1982, pp.95-101.

[Sezan 91] M. I. Sezan and H. J. Trussell. “Prototype image constraints for set-theoretic image restoration,” IEEE Transaction On Signal Processing, Vol. 39. No. 10, 1991, pp.2275-2285.

[Stucki 81] P. Stucki, “MECCA – a multiple error correcting computation algorithm for bi-level image hard copy reproduction,” Research report RZ1060, IBM Research Laboratory, Zurich, Switzerland, 1981

[Tikhonov 77] A. N. Tikhonov, V. Y. Arsenin, “Solutions of Ill-posed Problems,” Wiley, New York, 1977.

[Trussell 84] H. J. Trussell and M. R. Civanlar, “The feasible solution in signal restoration,” IEEE Transaction On Acoustics, Speech, and Signal Processing, Vol. 32. No 2. 1984, pp.201-212.

[Trussell 79] H. J. Trussell and B. R. Hunt, “Improved methods of maximum a posteriori restoration,” IEEE Transaction On Computers, Vol. 27, No. 1, Jan 1979, pp.57-62.

[Ulichney 88] R. A. Ulichney, “Dithering with blue noise,” Proceeding of the IEEE, Vol. 76, No. 1, 1988, pp.56-79.

[Ulichney 91] R. Ulichney, Digital Halftoning, Cambridge, MA: MIT Press, 1987.

[Witten 82] I. H. Witten and M. Neal, “Using Peano curves for bilevel display of continuous-tone images,” in Proc. IEEE CG&A, May 1982, pp.47-52.

[Wong 95] P. W. Wong, “Inverse Halftoning and kernel estimation for error diffusion,” IEEE Transaction On Image Processing, Vol. 6, Apr 1995, pp.486-498.

[Wong 96] P. W. Wong, “Adaptive error diffusion and its application in multiresolution rendering,” IEEE Transaction On Image Processing, Vol. 5, No. 7, 1996, pp.1184-1196.

[Wong 03] P. W. Wong, “Multi-resolution binary image embedding,” in Proceedings of SPIE, Vol. 50, No. 20, 2003, pp.423-429.

[Youla 82] D. C. Youla and H. Webb. “Image restoration by the method of Convex projections: Part 1-Theory,” IEEE Transactions on Medical Imaging, Vol. 1, No. 2, Oct 1982, pp.81-94.

[Yu 98] G. S. Yu, M. M. K. Liu, and M. W. Marcellin. “POCS-based error concealment for packet video using multiframe overlap information,” IEEE Transaction On Circuits and Systems for Video Technology, Vol. 8, No. 4, Aug 1998, pp.422-434.

[Zakhor 93] A. Zakhor, S. Lin and F. Eskafi, “A new class of B/W halftoning algorithms,” IEEE Transaction On Image Processing, Vol.2, Oct 1993, pp.499-509.

[Zhang 96] X. Zhang and B. Wandell, “A spatial extension of cielab for digital colour image reproduction,” Proc. Soc. Inform. Display 96 Digest, San Diego, 1996, pp.731-734.