UNIVERSITÀ DEGLI STUDI DI CATANIA

DOTTORATO DI RICERCA IN INFORMATICA – XXIII ciclo

Tesi di Dottorato

Advanced Techniques for Image Analysis and Enhancement

Applied Image Processing Algorithms for Embedded Devices

Giuseppe MESSINA

Tutor	**Coordinatore**
Ch.mo Prof. Sebastiano Battiato	Ch.mo Prof. Domenico Cantone

ANNO ACCADEMICO 2009-2010

To my wife Concetta, Andrea Paolo and Gioele.

Acknowledgements

I would like to take this opportunity to thank all my colleagues at the Advanced System Technology, Catania Lab (STMicroelectronics) and the whole Image Processing Lab (Catania University) in particular prof. Battiato for his patience. A special thanks also to Valeria and Mirko for our big achievements, and to Giovanni Maria and Daniele for the excellent job. Finally I would like to acknowledge Tony and Rosetta for their valuable companionship on the long way to our final Ph.D. target. After three years on my shoulders I will certainly miss them.

Contents

Preface

The research activities, described in this thesis, have been mainly focused on images analysis and quality enhancement. Specifically the research regards the study and development of algorithms for color interpolation, contrast enhancement and red-eye removal, which have been exclusively oriented to mobile devices. Furthermore an images analysis for forgeries identification and image enhancement, usually directed by investigators (Forensic Image Processing) has been conducted.

The thesis is organized in three main parts: Image Processing for Embedded Devices; Image Analysis and Enhancement; Forensics Image Processing.

Image Processing for Embedded Devices

Imaging consumer, prosumer and professional devices, such as digital still and video cameras, mobile phones, personal digital assistants, visual sensors for surveillance and automotive applications, usually capture the scene content by means of a single sensor (CCD or CMOS), covering its surface with a Color Filter Array (CFA), thus significantly reducing costs, sizes and registration errors. The most common arrangement of spectrally selective filters is known as Bayer pattern [25]. This simple CFA, taking into account human visual system characteristics , consists of a simple RGB lattices and contains twice as many green as red or blue sensors (human eyes are more sensitive to green with respect to the other primary colors)[BCGM10]. Some spatially undersampled color channels are then provided by the sensor and the full color information is reconstructed by color interpolation algorithms (demosaicing) [GMTB08]. Demosaicing is a very critical task. A lot of annoying artifacts that heavily degrade picture quality can be generated in this step: zipper effect, false color, etc. False colors are evident color errors which arise near the object boundaries, whereas zipper effect artifacts manifest as “on-off” patterns and are caused by an erroneous interpolation across edges. The green channel is less affected by aliasing than the red and blue channels, because it is sampled at a higher rate. Simple intra-channel interpolation algorithms (e.g., bilinear, bicubic) cannot be then applied and more advanced solutions (inter-channel), both spatial and frequency domain based, have been developed [GMT10a]. In embedded devices the complexity of these algorithms must be pretty low. Demosaicing approaches are not always able to completely eliminate false colors and zipper effects, thus imaging pipelines often include a post-processing module, with the aim of removing residual artifacts [TGM09].

It is worthwhile to understand that picture quality is strictly related not only to the number of pixels composing the sensor, but also to the quality of the demosaicing algorithm within the Image Generation Pipeline (IGP). The IGP usually consists of a preprocessing block (auto-focus, auto-exposure, etc.), a white balancing, a noise reduction, a color interpolation, a color matrixing step (that corrects the colors depending on the sensor architecture), and postprocessing blocks (sharpening, stabilization, compression, red-eyes removal, etc.) [BBMP10,BCM10].

Color interpolation techniques should be implemented by considering the artifacts introduced by the sensor and the interactions with the other modules composing the image processing pipeline, as it has been well analyzed in [5]. This means that demosaicing approaches have to guarantee the rendering of high quality pictures avoiding typical artifacts, which could be emphasized by the sharpening module, thus drastically deteriorating the final image quality. In the meantime, demosaicing should avoid introducing false edge structures due to residual noise (not completely removed by the noise reduction block) or green imbalance effects. Green imbalance is a mismatch arising in some sensors because the photosensitive elements that capture G intensity values at G_R locations can have a different response than the photosensitive elements that capture G intensity values at and G_B locations. This effect is mainly due to crosstalk [73].

In order to broaden the know-how and to allow an efficient study of the state of the art in the color interpolation field, an extensive patent research has been performed, since this problem dealt with industrial processes [BGMT08]. In Chapter 1 we describe the state of the art of demosaicing techniques. A complete excursus on color interpolation techniques is described, from spatial to frequency domain approaches, also in terms of color artifacts removal. Thus in Chapter 2 a new color interpolation technique, developed for embedded devices, is described. The method has been compared with other state of the art approaches and has shown good performances both in term of color reconstruction and artifact reduction. Furthermore it has been established that the method is also able to drastically reduce residual noise, preserving details [GMT10].

Image Analysis and Enhancement

Red-eye artifact is caused by the flash light reflected off a person's retina. This effect often occurs when the flash light is very close to the camera lens, as in most compact imaging devices. To reduce these artifacts, most cameras have a red-eye flash mode which fires a series of pre-flashes prior to picture capturing. Rapid pre-flashes cause pupil contraction thus minimizing the area of reflection; it does not completely eliminate the red-eye effect, though reduces it. The major disadvantage of the pre-flash approach is power consumption (e.g., flash is the most power-consuming device of the camera). Besides, repeated flashes usually cause uncomfortable feeling.

Alternatively, red-eyes can be detected after photo acquisition. Some photo-editing software make use of red-eye removal tools which require considerable user interaction. To overcome this problem, different techniques have been proposed in literature (see [56, 106] for recent reviews in the field). Due to the growing interest of industry, many automatic algorithms, embedded on commercial software, have been patented in the last decade [57]. The huge variety of approaches has permitted to explore different aspects of red-eyes identification and correction [BFGMR10,BFMGR10b]. The big challenge now is to obtain the best results with the minor number of visual errors.

To this end, several low-level feature eyes classification techniques have been analyzed [BGMM09,GGMT10]. This has allowed us to identify a methodology that has improved the performance of the techniques previously developed at the state of the art. In Chapter 3 we provide an overview of well-known automatic red eye detection and correction techniques, pointing out working principles, strengths and weaknesses of the various solutions [MM10]. Finally we describe our advanced red-eyes removal pipeline (see section 3.3). After an image filtering pipeline devoted to select only the potential regions in which red-eye artifacts are likely to be, a cluster-based boosting on gray codes based features is employed for classification purpose. Red-eyes are then corrected through desaturation and brightness reduction. Experiments on a representative dataset confirm the real effectiveness of the proposed strategy which also allows to properly managing the multi-modally nature of the input space. The obtained results have pointed out a good trade-off between overall hit-rate and false positives. Moreover, the proposed approach has shown good performance in terms of quality measure [BFGMR10a]. This activity has also produced the filling of a patent application to the Italian patent office [MGF09] and its extension to the U.S. patent office (which is at a preliminary stage).

Finally we have faced the exposure correction of wrongly acquired images. The problem of the proper exposure settings for image acquisition is of course strictly related with the dynamic range of the real scene [BMC08]. In many cases some useful insights can be achieved by implementing ad-hoc metering strategies. Alternatively, it is possible to apply some tone correction methods that enhance the overall contrast of the most salient regions of the picture. The limited dynamic range of the imaging sensors doesn't allow to recover the dynamic of the real world. In Chapter 4 we present a brief review of automatic digital exposure correction methods trying to report the specific peculiarities of each solution. Starting from exposure metering techniques, which are used to establish the correct exposition settings, we describe automatic methods to extract relevant features and perform corrections [CM10].

Forensics Image Processing

The analysis and improvement of image quality, for forensic use, are the subject of recent studies and have had a major boost with the advent of the digital imaging [12]. In this context it was agreed to deal with two different aspects of the image processing for forensics use: tampering identification into digital images (through the analysis of available data) and quality improvement of digital evidence for investigations purpose.

Nowadays the ubiquity of video surveillance systems and camera phones have made available a huge amount of digital evidences to the investigators [BFMP10, BM10a]. Unfortunately one of the main problem is the malleability of digital images for manipulation. Hence the detection of tampering into digital images is a research topic of particular interest, both in academia contents, as evidenced by the recent literature, and in forensics field, as there are several requirements to validate the reliability of digital evidence in legal processes [BMR09].

One of the existing approaches considers the possibility of exploiting the statistical distribution of DCT coefficients of JPEG images in order to reveal the irregularities due to the presence of a signal superimposed onto the original (e.g., due to copy and paste). As recently demonstrated [45–48], the relationship between the quantization tables used to compress the signal, before and after the forgery, highlights anomalies in the histograms of DCT coefficients, especially for certain frequencies.

Starting from an initial study of the prior art some preliminary results of the detection of forgery (i.e., detection of tampering of the image) have been presented to the anti-pedophilia group (Crime Against Children) at Interpol in Lyon (France). This presentation formed the basis for a possible collaboration with international intelligence agencies and allowed the identification of the issues of existing approaches. The research was then directed to a detailed analysis of the performance of existing approaches to assess their effectiveness and weaknesses, and then outlined the basis for the implementation of an approach robust enough to deal with quality tests disappointed by the existing methods. The first results obtained from the analysis of robustness of some algorithms were then presented at an international conference [BM09]. These studies, described in Chapter 5, have highlighted the gaps in the known techniques and also has emphasized the need to create a database of forged images [BMT10], which could be candidate as reference point for the scientific community [81].

The second research field, described in Chapter 6, involves the quality improvement of data for investigative use. The quality of images obtained from digital cameras and camcorders today is greatly improved since the first models of the eighties. Unfortunately it is still not unusual, in the forensic field, to get into images wrongly exposed, noisy and/or corrupted by motion blur. This is often due to the presence of old devices or to an illumination of the scene that does not allow a correct acquisition (for example in night shots). Extrapolating some details, normally invisible to human eye, by improving the quality of the image, is of crucial aspect to facilitate investigations [BMS10]. The current rules, for the production of forensic digital material, require to fully document the steps of images processing, in order to allow the exact replication of the enhancement [1, 3]. The automation of image enhancement techniques are therefore required and must be thoroughly documented. This chapter describes a tool for the automation of image enhancement technique, that allows both the automatic image correction and the generation of a script able to repeat the chain of enhancement steps. This work has been developed through a request received from the RIS (Scientific Investigations Department) of Messina (Italy), which has expressed a strong interest into this research area.

Personal Bibliography

[GMTB08] M.I. Guarnera, G. Messina, V. Tomaselli, and A. Bruna. Directionally filter based demosaicing with integrated antialiasing. In *Proc. of the International Conference in Consumer Electronics*, Las Vegas (NE), January 2008. ICCE 2008.

[BGMT08] S. Battiato, M.I. Guarnera, G. Messina, and V. Tomaselli. Recent patents on color demosaicing,. *Recent Patents on Computer Science, Bentham Science Publishers Ltd*, 1(2), 2008.

[BMC08] S. Battiato, G. Messina, and A. Castorina. *Single-Sensor Imaging: Methods and Applications for Digital Cameras*, chapter 12.Exposure

Correction for Imaging Devices: an Overview, pages 323 – 349. CRC Press, 2008. ISBN: 978-1420054521.

[TGM09] V. Tomaselli, M.I. Guarnera, and G. Messina. False-color removal on the ycc color space. In *Proc. of the IS&T/SPIE Electronic Imaging 2009*, San Jose (CA), January 2009. SPIE 2009.

[BGMM09] S. Battiato, M. Guarnera, T. Meccio, and G. Messina. Red eye detection through bag-of-keypoints classification. In Berlin Heidelberg 2009, editor, *LNCS 5646*, pages 180 – 187, Salerno,Italy, September 2009. 15th International Conference on Image Analysis and Processing, Springer-Verlag.

[BM09] S. Battiato and G. Messina. Digital forgery estimation into dct domain - a critical analysis. In *ACM Multimedia 2009 Workshop Multimedia in Forensics*, Beijing, China, October 2009. ACM.

[BMR09] S. Battiato, G. Messina, and R. Rizzo. *IISFA Memberbook 2009, Digital Forensics, Condivisione della conoscenza tra i membri dell'IISFA ITALIAN CHAPTER*, chapter 1. Image Forenscis: Contraffazione Digitale e Identificazione della Camera di Acquisizione: Status e Prospettive, pages 1–48. International Information Systems Forensics Association, Dicembre 2009.

[MGF09] G. Messina, M. Guarnera, and G. M. Farinella. Method and Apparatus for Filtering Red and/or Golden Eye Artifacts in Digital Images Italian Patent Pending, 18 December 2009, Application number IT-RM09A000669.

[GGMT10] M. Guarnera, I. Guarneri, G. Messina, and V. Tomaselli. A signature analysis-based method for elliptical shape detection. In *Proc. of the IS'T/SPIE Electronic Imaging 2009*, San Jose (CA), January 2010. SPIE 2010.

[BMS10] S. Battiato, G. Messina, and D. Strano. Chain of evidence generation for contrast enhancement in digital image forensics. In *Proc. of the IS'T/SPIE Electronic Imaging 2009*, San Jose (CA), January 2010. SPIE 2010.

[GMT10] M. Guarnera, G. Messina, and V. Tomaselli. Adaptive color demosaicing and false color removal. *Journal of Electronic Imaging, Special Issue on Digital Photography, SPIE and IS&T*, 19(2):021105–1–16, Apr-Jun 2010.

[BMT10] S. Battiato, G. Messina, and L. Truppia. An improved benchmarking dataset for forgery detection strategies. In *Minisymposium on Image and Video Forensics*, Cagliari (Italy), June 2010. SIMAI.

[BFGMR10] S.Battiato, G.M. Farinella, M. Guarnera, G. Messina, and D. Ravì. Boosting gray codes for red eyes removal. In *International Conference on Pattern Recognition ICPR 2010*, pages 1–4, Instanbul (TK), August 2010.

[BFGMR10a] S.Battiato, G.M. Farinella, M. Guarnera, G. Messina, and D. Ravì. Red-eyes removal through cluster based linear discriminant analysis. In *IEEE ICIP 2010 - International Conference on Image Processing*, pages 1–4, Hong Kong, September 2010.

[BFGMR10b] S. Battiato, G.M. Farinella, M. Guarnera, G. Messina, and D. Ravì. Red-eyes removal through cluster based boosting on gray codes. *EURASIP Journal on Image and Video Processing, Special Issue on Emerging Methods for Color Image and Video Quality Enhancement*, pages 1–19,ISSN: 1687-5176, 2010.

[BBMP10] S. Battiato, A. Bruna, G. Messina, and G. Puglisi. *Image Processing for Embedded Devices - From CFA data to image/video coding* , eBook, Bentham Science Publishers Ltd., eISBN: 978-1608051700, vol.1, November 2010.

[BCGM10] A. R. Bruna, A. Capra, M. Guarnera, and G. Messina. *Image Processing for Embedded Devices - From CFA data to image/video coding*, volume 1, chapter 2. Notions about Optics and Sensors. S. Battiato and A. Bruna and G. Messina and G. Puglisi, eBook, Bentham Science Publishers Ltd., eISBN: 978-1608051700, vol.1, November 2010.

[CM10] A. Castorina and G. Messina. *Image Processing for Embedded Devices - From CFA data to image/video coding*, volume 1, chapter 3. Exposure Correction. S. Battiato and A. Bruna and G. Messina and G. Puglisi, eBook, Bentham Science Publishers Ltd., eISBN: 978-1608051700, vol.1, November 2010.

[GMT10a] M. Guarnera, G. Messina, and V. Tomaselli. *Image Processing for Embedded Devices - From CFA data to image/video coding*, volume 1, chapter 7. Demosaicing and Aliasing Correction. S. Battiato and A. Bruna and G. Messina and G. Puglisi, eBook, Bentham Science Publishers Ltd., eISBN: 978-1608051700, vol.1, November 2010.

[MM10] T. Meccio and G. Messina. *Image Processing for Embedded Devices - From CFA data to image/video coding*, volume 1, chapter 8. Red Eyes Removal. S. Battiato and A. Bruna and G. Messina and G. Puglisi, eBook, Bentham Science Publishers Ltd., eISBN: 978-1608051700, vol.1, November 2010.

[BCM10] S. Battiato, A. Castorina, and G. Messina. *Image Processing for Embedded Devices - From CFA data to image/video coding*, volume 1, chapter 13. Beyond Embedded Devices. S. Battiato and A. Bruna and G. Messina and G. Puglisi, eBook, Bentham Science Publishers Ltd., eISBN: 978-1608051700, vol.1, November 2010.

[BFMP10] S. Battiato, G.M. Farinella, G. Messina, and G. Puglisi. *IISFA Memberbook 2010, Digital Forensics, Condivisione della conoscenza tra i membri dell'IISFA ITALIAN CHAPTER*, chapter 11. Digital Video Forensics: Status e Prospettive, pages 271-296. International Information Systems Forensics Association, December 2010. in press.

[BM10a] S. Battiato and G. Messina Video digitali in ambito forense. In *Ciberspazio e Diritto*, vol. IV, 2010, in press.

Part I

Image Processing for Embedded Devices

1. Color Interpolation and False Color Removal

The simplest demosaicing (or color interpolation) method is the bilinear interpolation, a proper average on each pixel depending on its position in the Bayer Pattern. For a pixel, we consider its eight direct neighbors and then we determine the two missing colors of this pixel by averaging the colors of the neighboring ones. We actually have 4 different cases of averaging which correspond to the red pixel, the blue pixel, the green pixel on a red row and the green pixel on the blue row. On each of them, the averaging will be slightly different. Assuming the notation used in Fig.(**1.1**) and considering, as example, the pixels R_{33}, B_{44}, G_{43} and G_{34}, the bilinear interpolation proceeds as follows:

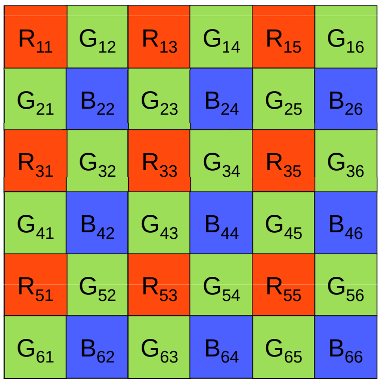


Figure 1.1 : Example of Bayer pattern.

The interpolation on a red pixel (R_{33}) produces the RGB triplet as:

$$\begin{aligned} Red &= R_{33} \\ Green &= \frac{G_{23}+G_{34}+G_{32}+G_{43}}{4} \\ Blue &= \frac{B_{22}+B_{24}+B_{42}+B_{44}}{4} \end{aligned} \tag{1.1}$$

The interpolation on a green pixel in a red row (G_{34}):

$$\begin{aligned} Red &= \frac{R_{33}+R_{35}}{2} \\ Green &= G_{34} \\ Blue &= \frac{B_{24}+B_{44}}{2} \end{aligned} \tag{1.2}$$

The interpolation on a green pixel in a blue row (G_{43}):

$$\begin{aligned} Red &= \frac{R_{33}+R_{53}}{2} \\ Green &= G_{43} \\ Blue &= \frac{B_{42}+B_{44}}{2} \end{aligned} \tag{1.3}$$

The interpolation on a blue pixel (B_{44}):

$$\begin{aligned} Red &= \frac{R_{33}+R_{35}+R_{53}+R_{55}}{4} \\ Green &= \frac{R_{33}+R_{35}+R_{53}+R_{55}}{4} \\ Blue &= B_{44} \end{aligned} \tag{1.4}$$

Despite this interpolation is very simple, the results are unsatisfactory: as many other traditional color interpolation methods, usually results present color edge artifacts, due to the non-ideal sampling performed by the CFA.

The term aliasing refers to the distortion that occurs when a continuous time signal is sampled at a frequency lower than twice its highest frequency. As stated in the Nyquist-Shannon sampling theorem, an analog signal that has been sampled can be perfectly reconstructed from the samples if the sampling rate exceeds *2B* samples per second, where *B* is the highest frequency in the original signal. If the highest frequency in the original signal is known, this theorem gives the lower bound on sampling frequency assuring perfect reconstruction. On the other hand, if the sampling frequency is known, the Nyquist-Shannon theorem gives the upper bound to the highest frequency (called Nyquist frequency) of the signal to allow the perfect reconstruction. In practice, neither of these two statements can be completely satisfied because they require band-limited original signals, which do not contain energy at frequencies higher than a certain bandwidth B. An example of band-limited signal is depicted in Fig.(**1.2**).

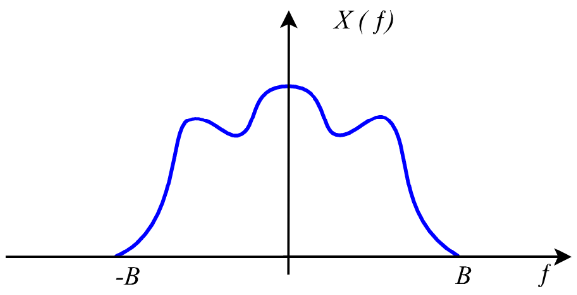


Figure 1.2 : Example of a bandlimited signal.

In real cases a ”time-limited” or a ”spatial-limited” signal can never be perfectly band-limited. For this reason, an anti-aliasing filter is often placed at the input of digital signal processing systems, to restrict the bandwidth of the signal to approximately satisfy the sampling theorem. In case of any imaging devices an optical low pass filter smoothes the signal in the spatial optical domain in order to reduce the resolution below the limit of the digital sensor, which is strictly related to the pixel pitch, Xs, which is the distance between two adjacent pixels. As explained in [116], the sampling frequency of the sensor is

$$fs = \frac{1}{Xs} \tag{1.5}$$

To reproduce a spatial frequency there must be a pair of pixels for each cycle. One pixel is required to respond to the black half cycle and one pixel is required to respond to the white half cycle. In other words one pixel can only represent a half signal cycle, and hence the highest frequency the array can reproduce is half its sampling frequency and it is called Nyquist frequency:

$$f_N = \frac{1}{2Xs} \tag{1.6}$$

In a two dimensional array, having the same pixel pitch in both directions, Nyquist and sampling frequencies are equal in both X and Y axes. If a Bayer color filter array is applied on the sensor surface, the Nyquist and the sampling frequencies are different for the G channel and the R/B channels. Red and blue channels have the same pattern, so they have the same Nyquist frequency. In particular, let *p* be the monochrome pixel pitch; as noticeable from Fig.(**1.3(a)**) the red and blue horizontal and vertical pixel pitch *Xs* is

$$Xs = 2p \tag{1.7}$$

and hence the Nyquist frequency for red and blue channels in both horizontal and vertical directions is

$$f_{N_{RB_{hv}}} = \frac{1}{4p} \tag{1.8}$$

Looking at Fig.(**1.3(b)**), is possible to derive the diagonal spacing between two adjacent red/blue pixels:

$$Xs = \sqrt{2}p \tag{1.9}$$

the diagonal Nyquist frequency becomes:

$$f_{N_{RB_d}} = \frac{\sqrt{2}}{4p} \tag{1.10}$$

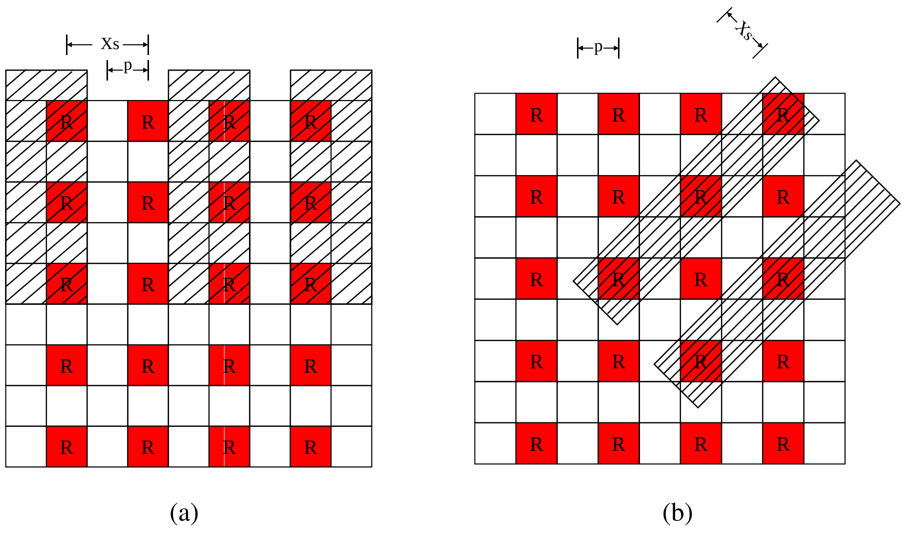


Figure 1.3 : Red array line pairs at the Nyquist frequency.

This means that the diagonal Nyquist frequency is larger than the horizontal/vertical one by a $\sqrt{2}$ factor.

As far as the green channel is concerned, the horizontal and vertical pixel pitch equals the monochrome pixel pitch (see Fig.(**1.4**)), and hence its Nyquist frequency equals that of the monochrome array:

$$f_{N_{G_{hv}}} = \frac{1}{2p} \tag{1.11}$$

The diagonal Nyquist frequency, instead, equals that of the red/blue channels, which has been already shown in (1.10).

From this analysis it is easily derivable that red and blue channels are more affected by aliasing effects than the green channel. Despite the application of an optical anti-aliasing filter, aliasing artifacts often arise due to the way the signal is reconstructed in terms of color interpolation.

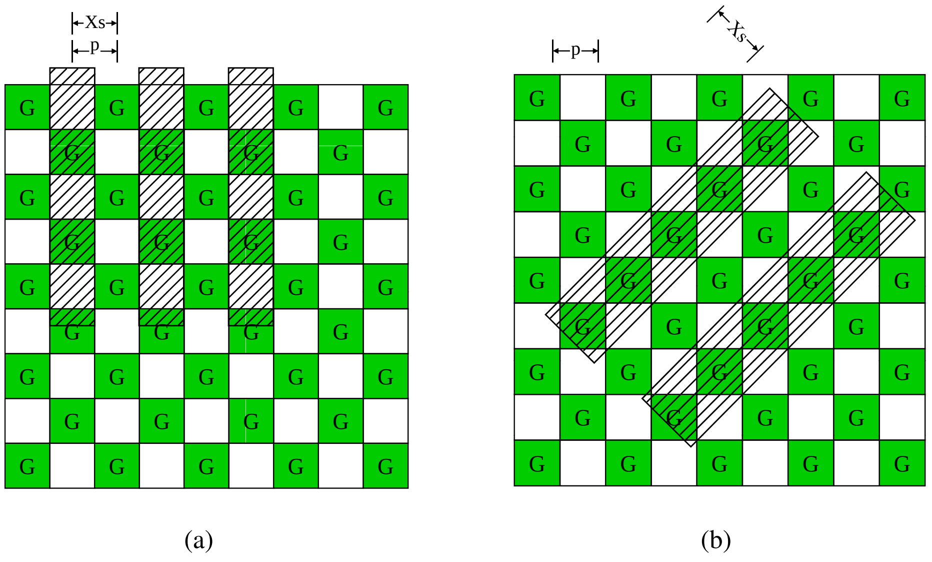


Figure 1.4 : Green array line pairs at the Nyquist frequency.

Color interpolation techniques should be implemented by considering the artifacts introduced by the sensor and the interactions with the other modules composing the image processing pipeline, as it has been well analyzed in [5]. This means that demosaicing approaches have to guarantee the rendering of high quality pictures avoiding typical artifacts, which could be emphasized by the sharpening module, thus drastically deteriorating the final image quality. In the meantime, demosaicing should avoid introducing false edge structures due to residual noise (not completely removed by the noise reduction block) or green imbalance effects. Green imbalance is a mismatch arising at G_R and G_B locations. This effect is mainly due to crosstalk [73].

In the last years a wide variety of works has been produced about color interpolation, exploiting a lot of different approaches [64]. In this Chapter we review some of the state of the art solutions devoted to demosaicing and antialiasing, paying particular attention to the patents [21].

1.1 Color Interpolation Techniques

Demosaicing solutions can be basically divided into two main categories: spatial-domain approaches and frequency-domain approaches.

1.1.1 Spatial-Domain Approaches

In this sub-section we describe some recent solutions, devoted to demosaicing, which are typically fast and simple to be implemented inside a system with low capabilities (e.g., memory requirement, CPU, low-power consumption, etc.).

In the following we present techniques based on spatial and spectral correlations.

Spatial Correlation Based Approaches

One of the principles of color interpolation techniques is to exploit spatial correlation. According to this principle, within a homogeneous image region, neighboring pixels share similar color values, so a missing value can be retrieved by averaging the pixels close to it. In presence of edges, the spatial correlation principle can be exploited by interpolating along edges and not across them. Techniques which disregard directional information often produce images with color artifacts. Bilinear interpolation belongs to this class of algorithms. On the contrary, techniques which interpolate along edges are less affected by this kind of artifact. Furthermore, averaging the pixels which are across an edge also leads to a decrease in the sharpness of the image.

Edge based color interpolation techniques are widely disclosed in literature, and can be differentiated according to the number of directions, the way adopted to choose the direction and the interpolation method.

The method in [66] discloses a technique which firstly interpolates the green color plane, then interpolates the remaining two planes. A missing G pixel can be interpolated horizontally, vertically or by using all the four samples around it. With reference to the neighborhood of Fig.(**1.5**) the interpolation direction is chosen through two values:

$$\Delta H = |-A3 + 2 \cdot A5 - A7| + |G4 - G6| \tag{1.12}$$

$$\Delta V = |-A1 + 2 \cdot A5 - A9| + |G2 - G8| \tag{1.13}$$

which are composed of Laplacian second-order terms for the chroma data and gradients for the green data, where the Ai can be either R or B.

Once the G color plane is interpolated, R and B at G locations are interpolated. In particular, a horizontal predictor is used if their nearest neighbors are in the same row, whereas a vertical predictor is used if their nearest neighbors are in the same column. Finally, R is interpolated at B locations and B is interpolated at R locations.

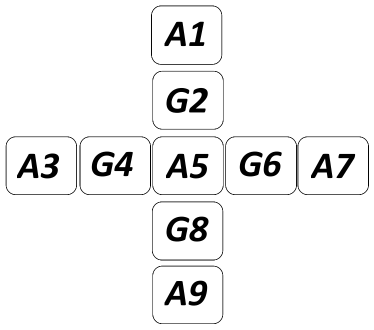


Figure 1.5 : Considered neighborhood.

Although the interpolation is not just an average of the neighboring pixels, wrong color can be introduced near edges. To improve the performances, in [82] a con-

$$\begin{bmatrix} 1 & -1 & -2 & 1 & 1 \end{bmatrix} \qquad \begin{bmatrix} 1 \\ -1 \\ -2 \\ 1 \\ 1 \end{bmatrix}$$

(a) Horizontal mask (b) Vertical mask

Figure 1.6 : Variation masks proposed in [109].

trol factor of the Laplacian correction term is introduced. This control mechanism allows increasing the sharpness of the image, reducing at the same time wrong colors and ringing effects near edges. In particular, if the Laplacian correction term is greater than a predefined threshold, it is changed by calculating an attenuating gain, which depends on the minimum and maximum values of the G channel and of another color channel. A drawback of these methods is that G can be interpolated only in horizontal and vertical directions; R and B can be interpolated only in diagonal directions (in case of B and R central pixel) or in horizontal and vertical directions (in case of G central pixel).

The approach proposed in [109], similarly to the previous one, interpolates the missing G values in either horizontal or vertical direction, and chooses the direction depending on the intensity variations within the observation window. The variation filters, shown in Fig.(**1.6**), take into account both G and non-G intensity values. In this case, the interpolation of G values is achieved through a simple average of the neighboring pixels in the chosen direction, but the quality of the image is improved by applying a sharpening filter. One important peculiarity of this method is the $G_R - G_B$ mismatch compensator step, which tries to overcome the green imbalance issue. In some sensors the photosensitive elements that capture G intensity values at G_R locations can have a different response than the photosensitive elements that capture G intensity values at G_B locations. The $G_R - G_B$ mismatch module applies gradient filters and curvature filters to derive the maximum variation magnitude. If this value exceeds a predefined threshold value, the $G_R - G_B$ smoothed intensity value is selected, otherwise the original G intensity value is selected. To interpolate the missing R and B values, the color correlation is exploited. In fact, discontinuities of all the color components are assumed to be equal. Thus, color discontinuity equalization is achieved by equating the discontinuities of the remaining color components with the discontinuities of the green color component. Methods which use color correlation in addition to edge estimation usually provide higher quality images.

All the already disclosed methods propose an adaptive interpolation process in which some conditions are evaluated to decide between the horizontal and vertical interpolation. When neither a horizontal edge nor a vertical edge is identified, the interpolation is performed using an average value among surrounding pixels. This means that resolution in appearance deteriorates in the diagonal direction. Moreover, in regions near the vertical and horizontal Nyquist frequencies, the interpolation direction can abruptly change, thus resulting in unnaturalness in image quality. To overcome the above mentioned problems, the method in [139] prevents an interpolation result from being changed discontinuously with a change in the correlation direction. First of all, vertical (ΔV) and horizontal correlation values (ΔH) of a target pixel to be interpolated are calculated by using the equations in (1.12). Then, a coefficient term, depending on the direction in which the target pixel has higher correlation, is computed:

$$K = \begin{cases} 0 & if\,\Delta H = \Delta V \\ 1 - \dfrac{\Delta V}{\Delta H} & if\,\Delta H > \Delta V \\ \dfrac{\Delta H}{\Delta V} - 1 & if\,\Delta H < \Delta V \end{cases} \tag{1.14}$$

Thus K has values in the range [-1,1].

The K coefficient is used to weight the interpolation data in the vertical or horizontal direction with the interpolation data in the diagonal direction. If K has a positive value ($\Delta V < \Delta H$), that is a vertical edge is found, a weighted average of the vertical interpolated value (V_{value}) and the two-dimensional interpolated value ($2D_{value}$) is calculated using the (1.15), where Ka is the absolute value of the coefficient K.

$$Output = V_{value} \times Ka + 2D_{value} \times (1 - Ka) \tag{1.15}$$

Obviously, if K is a negative value a weighted average of the horizontal interpolated value and the two-dimensional interpolated value is computed. As a result, a proportion of either the vertical or horizontal direction interpolation data can be continuously changed without causing a discontinuous change in interpolation result when the correlation direction changes.

The approach proposed in [110] is composed by an interpolation step followed by a correction step. The authors consider the luminance channel as proxy for G color, and the chrominance channel as proxy for R and B. Since the luminance channel is more accurate, it is interpolated before the chrominance channels. The luminance is interpolated as accurate as possible in order to not produce wrong modifications in the chrominance channels. However, after the interpolation step, luminance and chrominances are orderly refined. The interpolation phase is based on the analysis of the gradients in four directions (east, west, north and south), defined as follows:

$$\begin{array}{l} \Delta W = \left|2L_{(x-1,y)} - L_{(x-3,y)} - L_{(x+1,y)}\right| + \left|C_{(x,y)} - C_{(x-2,y)}\right| \\ \Delta E = \left|2L_{(x+1,y)} - L_{(x-1,y)} - L_{(x+3,y)}\right| + \left|C_{(x,y)} - C_{(x+2,y)}\right| \\ \Delta N = \left|2L_{(x,y-1)} - L_{(x,y-3)} - L_{(x,y+1)}\right| + \left|C_{(x,y)} - C_{(x,y-2)}\right| \\ \Delta S = \left|2L_{(x,y+1)} - L_{(x,y-1)} - L_{(x,y+3)}\right| + \left|C_{(x,y)} - C_{(x,y+2)}\right| \end{array} \tag{1.16}$$

Since the aim is to interpolate along edges and not across them, an inverted gradient function is formed:

$$f_{grad}(x) = \begin{cases} \frac{1}{x} & if\, x \neq 0 \\ 1 & if\, x = 0 \end{cases} \tag{1.17}$$

where x represents one of the gradient of (1.16). This function allows to weight more the smallest gradients and to follow the edge orientation. The interpolation of missing luminance values is performed using the normalized inverted gradient functions which weight both luminance and chrominance values in the neighborhood. The chrominance values are used in the interpolation of luminance to get a more accurate estimation. Similarly, chrominances are interpolated by using both luminance and chrominance data. The correction step comprises the luminance correction first, and then the chrominance correction.

The method in [105] aims to generate images with sharp edges. And also in this case a high frequency component, derived from the sensed color channel, is added to the low frequency component of the interpolated channels. This technique takes into account eight different directions, as it shown in Fig.(**1.7**), and uses 5×5 elliptical Gaussian filters to interpolate the low frequency component of each color channel (even the sensed one). For each available direction there is a different Gaussian filter, having the greater coefficients along the identified direction. These filters have the advantage of interpolating the missing information without generating annoying jaggy edges.

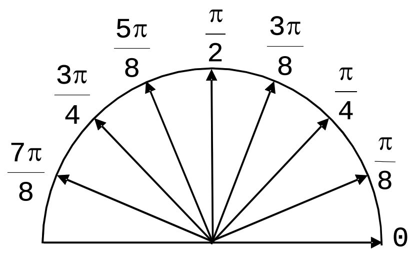


Figure 1.7 : Quantized directions for spatial gradients.

After having computed the low frequency component, for each color channel, an enhancement of the high frequencies content is obtained taking into account the color correlation (1.22). In particular, a correction term is calculated as the difference between the original sensed value and its low pass component, as it is retrieved through the directional Gaussian interpolation:

$$\Delta_{Peak} = G - G_{LPF} \tag{1.18}$$

This correction term is then added to the low frequency component of the channels to be estimated:

$$H = H_{LPF} + \Delta_{Peak} \tag{1.19}$$

The low frequency component, in this method, is calculated according to the identified direction, so it is less affected by false colors than previous inventions. Moreover, this solution provides a simple and effective method for calculating direction and amplitude values of spatial gradients, without making use of a first rough interpolation of the G channel. More specifically, 3×3 Sobel operators are applied directly on the Bayer pattern to calculate horizontal and vertical gradients. The orientation of the spatial gradient at each pixel location is given by the following equation:

$$or(x,y) = \begin{cases} \arctan\left(\dfrac{P' * Sobel_y(x,y)}{P' * Sobel_x(x,y)}\right) & \text{if } P' * Sobel_x(x,y) \neq 0 \\ \dfrac{\pi}{2} & \text{otherwise} \end{cases} \tag{1.20}$$

where $P' * Sobel_y$ and $P' * Sobel_x$ are the vertical and horizontal Sobel filtered values, at the same pixel location. The orientation $or(x,y)$ is quantized in eight predefined directions. Since the image could be deteriorated by noise, and the calculation of direction could be sensitive to it, a more robust estimation of direction is needed. For this reason, Sobel filters are applied on each 3×3 mask within a 5×5 window, thus retrieving nine gradient data. In addition to the orientation, the amplitude of each spatial gradient is calculated, by using the following equation:

$$mag(x,y) = \left(P' * Sobel_x(x,y)\right)^2 + \left(P' * Sobel_y(x,y)\right)^2 \tag{1.21}$$

The direction of the central pixel is finally derived through the “weighted-mode” operator, which provides an estimation of the predominant amplitude of the spatial gradient around the central pixel. This operator substantially reduces the effect of noise in estimating the direction to use in the interpolation phase.

Spectral Correlation Based Approaches

In this class of algorithms final RGB values are derived taking into consideration the inter-channel color correlations in a limited region. Gunturk *et al.* [65] has demonstrated that high frequency components of the three color planes are highly correlated, but not equal. This suggests that any color component can help to reconstruct the high frequencies for the remaining color components. For instance, if the central pixel is red R, the green G component can be determined as:

$$G(i,j) = G_{LPF}(i,j) + R_{HPF}(i,j) \tag{1.22}$$

where $R_{HPF}(i,j) = R(i,j) - R_{LPF}(i,j)$ is the high frequency content of the R channel, and G_{LPF} and R_{LPF} are the low frequency components if the G and R channels, respectively.

This implies that the G channel can take advantage of the R and B information. Furthermore for real world images the color difference planes ($\Delta_{GR} = G - R$ and $\Delta_{GB} = G - B$) are rather flat over small regions, and this property is widely exploited in demosaicing and antialiasing techniques. This model using channel differences (that can be viewed as chromatic information), is nearer to the Human Color Vision system that is more sensitive to chromatic changes than luminance changes in low spatial frequency regions. Like the previous example, if the central pixel is R, the green component can be derived as:

$$G = R + \Delta_{GR} \tag{1.23}$$

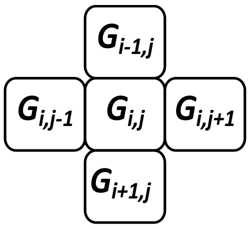


Figure 1.8 : Pattern of five pixels used to calculate an edge metric on a central G pixel of the LF (low frequency) G color channel.

The method proposed in [80] belongs to this class. The technique generates by first an estimation of all color channels (R, G and B) containing the Low Frequencies (LF) only. This is obtained by taking into consideration an edge strength metric to inhibit smoothing of detected edges. Then a difference between the estimated smoothed values and the original Bayer pattern values is performed to obtain the corresponding High Frequency (HF) values. Finally the low frequency channels and the corresponding estimated high frequency planes are combined into the final RGB image. In particular the high frequency values are obtained through the relations described in Table.1.1.

Table 1.1 : Color Correlations defined in [80].

	At a Red Pixel	**At a Green Pixel**	**At a Blue Pixel**
R	R	$R_{LPF}+G-G_{LPF}$	$R_{LPF}+B-B_{LPF}$
G	$G_{LPF}+R-R_{LPF}$	G	$G_{LPF}+B-B_{LPF}$
B	$B_{LPF}+R-R_{LPF}$	$B_{LPF}+G-G_{LPF}$	B

Each smoothed LF image is formed by a two-dimensional interpolation combined with a low-pass filtering excepted for pixels that maximize the edge strength metric. For example, if the central pixel is a G pixel the four adjacent G pixels, which will be taken into consideration to estimate the edge strength, are generated by interpolation (see Fig.(**1.8**)). Thus the measure of edge strength E_{ij}, that is proportional to the square of the actual edge difference, is then calculated according to:

$$E_{ij} = (G_{i,j} - G_{i,j-1})^2 + (G_{i,j} - G_{i,j+1})^2 + (G_{i,j} - G_{i-1,j})^2 + (G_{i,j} - G_{i+1,j})^2 \tag{1.24}$$

By considering this edge metric the algorithm reduce the presence of color artifacts on edges boundaries.

In [35] a method based on the smooth hue transition algorithms by using the color ratio rule is proposed. This rule is derived from the photometric image formation model, which assumes the color ratio is constant in an object. Each color channel is composed of the Albedo multiplied by the projection of the surface normal onto the light source direction. The Albedo is the fraction of incident light that is reflected by the surface, and is function of the wavelength (is different for each color channel) in a Lambertian surface (or even a more complicate Mondrian). The Albedo is constant in a surface, then the color channel ratio is hold true within the object region. This class of algorithms, instead of using inter-channel differences, calculates the green channel using a well-known interpolation algorithm (i.e., bilinear or bicubic), and then computes the other channels using the red to green and blue to green ratios, defined as:

$$H_b = \frac{B}{G} \;\; and \;\; \mathrm{H_r} = \frac{R}{G}. \tag{1.25}$$

An example of such method is described in [111]. In this work the Bayer data are properly processed by a LPF circuit and an adaptive interpolation module. The LPF module cuts off the higher frequency components of the respective color signals R, G and B and supplies R_{LPF}, G_{LPF} and B_{LPF}. On the other hand, the adaptive interpolation circuit calculates a local pixel correlation from the color signals R and G and executes interpolation with a pixel which maximizes the correlation to obtain a high resolution luminance signal.

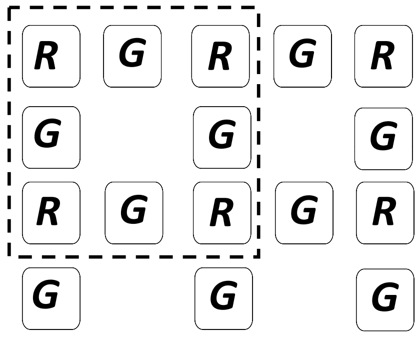


Figure 1.9 : RG pixel map for luminance interpolation.

The authors assume that, since the color signals R and G have been adjusted by the white balance module, they have almost identical signal levels and thus they can be considered as luminance signals. Taking into consideration the Bayer pattern selected in Fig.(**1.9**), they consider the luminance signals arranged as shown in Fig.(**1.10**), where the value Y_5 has to be calculated according to the surrounding values.

The correlation S for a set of pixels Y_n along a particular direction can be defined, similarly to the (1.25), as follows:

$$S = \frac{min(Y_n)}{max(Y_n)} \tag{1.26}$$

where $S \leq 1$ and the maximum correlation is obtained when $S = 1$.

The correlation is calculated for the horizontal, vertical and diagonal directions, and interpolation is executed in a direction which maximizes the correlation. For instance, for the vertical direction:

$$min(Y_n) = min(Y_1, Y_4, Y_7) \cdot min(Y_2, Y_8) \cdot min(Y_3, Y_6, Y_9) \tag{1.27}$$

and

$$max(Y_n) = max(Y_1, Y_4, Y_7) \cdot max(Y_2, Y_8) \cdot max(Y_3, Y_6, Y_9) \tag{1.28}$$

The correlations in the horizontal and diagonal directions are computed in a similar way. If the direction which maximizes the correlation is the vertical one, the interpolation is executed as follows:

$$Y_5 = \frac{(Y_2 + Y_8)}{2} \tag{1.29}$$

Another way to decide the direction is to consider the similarities between the pixels. The dispersion degree σ_R of the color R is calculated as:

$$\sigma_R = \frac{min(R_1, R_2, R_3, R_4)}{max(R_1, R_2, R_3, R_4)} \tag{1.30}$$

If the dispersion degree is greater than a threshold, interpolation along a diagonal direction is executed. On the contrary, when the dispersion degree is small, correlation of the color R is almost identical in any directions, so it is possible to interpolate only G along the vertical or horizontal direction. This implies that the interpolation is executed only with G having the highest frequency, thus enabling to obtain an image of a higher resolution.

Once the luminance signal Y is interpolated, a high pass filter (HPF) is applied to Y and the color signal R and G. The HPF creates a luminance signal Y_{HPF} containing higher frequency components only. Finally, an adder combines the already computed color signals R_{LPF}, G_{LPF} and B_{LPF} with the higher frequency component luminance signal Y_{HPF}.

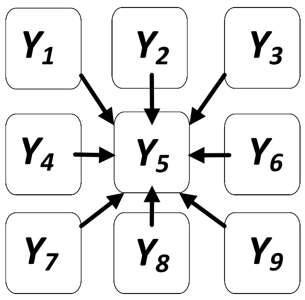


Figure 1.10 : Luminance map of analyzed signal.

Non Adaptive Approaches

The pattern based interpolation techniques perform, generally, a statistical analysis, by collecting actual comparisons of image samples with the corresponding full-color images. Chen *et al.* [34] propose a method to improve the sharpness and reduce the color fringes with a limited hardware cost. The approach consists of two main steps:

1. Data training phase:
   (a) Collecting samples and corresponding full-color images;
   (b) Forming pattern indexes, by selecting the concentrative window for each color in the Bayer samples and quantizing all the values on the window;
   (c) Calculating the errors between the reconstructed pixels and the actual color values;
   (d) Estimating the optimal combination of pattern indexes to be sorted into a database.
2. Data practice phase:
   (a) For each pixel a concentrative window is chosen, and within it, the pixels are quantized in two levels (Low, High) to form a pattern index,

as shown in Fig.(**1.11**). This index is then used as key for the database matching.

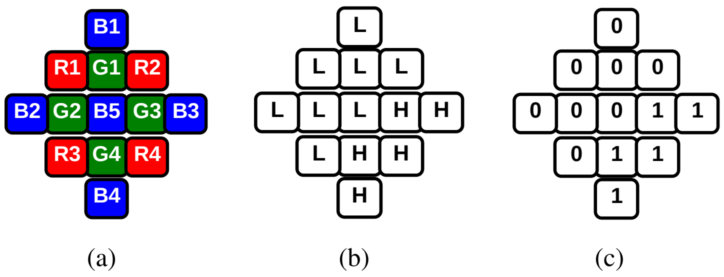


Figure 1.11 : Relationship between a color filter array and a concentrative window. **(a)** Bayer Pattern, **(b)** Quantization of acquired samples in two levels: Low (L) and High (H), **(c)** Resulting pattern index.

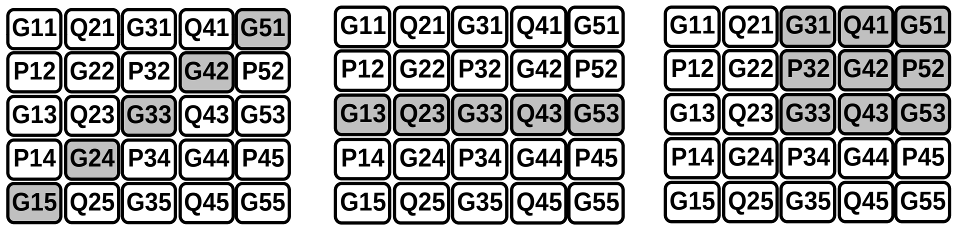
During the data training phase, the proposed method assumes that the reconstructed value (Rec_{value}) is function of the original value ($Orig_{value}$) and the feasible coefficient set ($feasible\_coefficient\_set$), which can be expressed as:

$$Rec_{value} = \frac{Orig_{value} * feasible\_coefficient\_set}{(sum\_of\_coefficients)} \tag{1.31}$$

Once the value has been calculated for each $feasible\_coefficient\_set$, the system chooses the set having the minimal error between the calculated values and the real value. These results are then stored into the database. During the data-practice phase, the reconstruction is based on color differences rules applied to the pixel neighborhood.

A simpler technique [86] uses a plurality of stored interpolation patterns. To select the correct interpolation pattern an analysis of the input signal is performed using gradient and uniformity estimation. In practice, by first the G channel is interpolated using the 8 stored fixed patterns (Horizontal, Vertical, the two Diagonals and the four corners). To achieve this purpose the uniformity and the gradient are estimated in the surrounding of the selected G pixel. The minimum directional data estimation $G_v(i)$ $(i \in [1..8])$, obtained through the eight fixed patterns, defines the best match with the effective direction.

For example, Fig.(**1.12.(a)**) shows an interpolation pattern in which low luminance pixels are arranged along the diagonal.



(a) (b) (c)

Figure 1.12 : Some samples of interpolation patterns.

The directional data $G_v(1)$, which represents a numerical value of the similarity between the surround of the pixel to be interpolated and the interpolation pattern, is obtained through the following expression:

$$G_v(1) = \frac{|G_{33} - G_{51}| + |G_{33} - G_{42}| + |G_{33} - G_{24}| + |G_{33} - G_{15}|}{4} \tag{1.32}$$

The remaining seven directional data are calculated in a similar manner, taking into account the fixed direction. The smallest directional data from $G_v(1)$ to $G_v(8)$ identifies the interpolation pattern which is the best fit to the image neighborhood of the pixel to be interpolated.

When one interpolation pattern only is present, providing the smallest directional value, it is chosen to perform the interpolation. On the contrary, when two or more interpolation patterns provide the smallest directional value, a correlation with the interpolation patterns of the surrounding pixels, whose optimum interpolation pattern has already been determined, is considered.

Specifically, if one of the interpolation patterns having the smallest value is the interpolation pattern of one surrounding G pixel, this pattern is chosen for performing the interpolation. Otherwise it is impossible to determine a specific pattern to use for the interpolation, and thus a simple low pass filter is applied.

If $G_v(1)$ is the smallest directional value:

$$G_0 = \frac{G_{15} + 2G_{24} + 2G_{33} + 2G_{42} + G_{51}}{8} \tag{1.33}$$

$$P_0 = \frac{P_{14} + P_{34} + P_{32} + P_{52}}{4} \tag{1.34}$$

$$Q_0 = \frac{Q_{23} + Q_{25} + Q_{43} + Q_{41}}{4} \tag{1.35}$$

where P and Q represent the R and B or B and R values.

If it is impossible to determine a specific pattern:

$$G_0 = \frac{(G_{22} + G_{24} + G_{42} + G_{44} + 4G_{33})}{8} \tag{1.36}$$

$$P_0 = \frac{(P_{34} + P_{32})}{2} \tag{1.37}$$

$$Q_0 = \frac{(Q_{23} + Q_{43})}{2} \tag{1.38}$$

Once the missing values for the G pixels have been processed, the algorithm calculates the missing values for the R and B pixels. If the interpolation patterns, estimated for the already processed G pixels, describe a fixed direction in the surrounding of the R/B pixel (that is several patterns indicate the same direction) then this pattern is used to perform the interpolation. Otherwise the numerical directional data are estimated. Like the G case, eight different interpolation patterns are stored in the interpolation storage memory and a directional data value is computed for each of these patterns. When there are two or more patterns having the smallest directional data value, correlations with the interpolation patterns of the already interpolated *G* pixels are evaluated. The reason why *G* pixels are taken into consideration instead of *R* and *B* pixels is that *G* pixels are more suitable for pattern detection than *R* and *B* pixels.

This class of techniques is very robust to noise, because it takes into consideration the interpolation patterns of the already processed pixels, but introduces jagged edges in abrupt diagonal transitions, due to the equations used in the interpolation step.

Iterative Approaches

In this category we collect all approaches that derive interpolation through an iterative process able to find after a limited number of cycles the final mosaicized image. In particular, in [70, 71, 83], starting from an initial rough estimate of the interpolation, the input data are properly filtered (usually using a combination of directional high-pass filters with some global smoothing) to converge versus stable conditions. These methods proceed in different ways with respect to the local image analysis but share the overall basis methodology.

In [83] a color vector image is formed containing the original Bayer values. After an initial estimate of the *RGB* original value for each pixel such quantity is updated by taking into account two different functions: “roughness” and “preferred direction”. The final missing color are defined by finding the values that minimize a weighted sum of *Rough* and *CCF* (Color Compatibility Function) functions over the image by using the following formula:

$$Q = \sum_{(m,n)} Rough(m,n) + \lambda \sum_{(m,n)} CCF(m,n) \tag{1.39}$$

where λ is a positive constant while $Rough(m,n)$ is defined in this case as the local summation of approximated local gradients and $CCF(m,n)$ is a function that penalizes local abruptly changes. By using the classic Gauss-Siedel approach the method converges after 4-5 iterations.

In [70] and [71] the luminance channel is properly extracted from input Bayer data and analyzed in a multiscale framework by applying smoothing filtering along preferred directions. Chrominance components are smoothed by isotropically smoothing filters. The final interpolated image is obtained after a few iterations. Just before to start a new iteration the pixel values are reset to the original (measured) values.

1.1.2 Frequency-domain Approaches

Demosaicing is an ill posed problem and thus it cannot have a unique solution. This can be easily understood by considering that different real images can have the same mosaiced representation [8]. The mosaicing operation cannot be inverted and thus, it necessary to consider a priori assumptions to extrapolate the missing information. All the demosaicing algorithms use specific a priori assumptions to design the interpolator operator. One of the a priori assumption is the band limited of image signal and the limit is due to the sampling rate of the color channels.

In natural images, the energy spectrum is primarily present in a low frequency region and high frequencies along the horizontal and vertical axes [49], and the human visual system is more sensitive to these high frequencies than to the ones present at the corner of the spectrum. The demosaicing algorithms in the frequency domain exploit these band limit assumptions.

Fourier Transform Analysis and Processing

Several demosaicing algorithms in the Fourier domain have been proposed in literature [59, 76, 77] exploiting the spectrum properties of the CFA mosaiced images. The spectral representation of a CFA image can be directly derived from its representation in the spatial domain.

A color image I can be represented as:

$$I(x,y) = \{C_i(x,y)\}, i \in \{R,G,B\}, (x,y) \in \mathbb{N}^2 \tag{1.40}$$

where C_i are the color vectors in the lattice *(x,y)*. Thus an image is expressed as a vector of three dimensions for each pixel. The color triplets C_i form a linear vector space of three dimensions. If we call I_{CFA} the spatial multiplexed version of the image I with a CFA pattern, we have:

$$I_{CFA}(x,y) = \sum_{i \in \{R,G,B\}} C_i(x,y) \cdot D_i(x,y) \tag{1.41}$$

where $D_i(x,y)$ are the sampling functions that have value 1 if the color channel is present at the location *(x,y)*, or 0 if not present. In case of the Bayer arrangement of CFA, the D_i represent the disjoint shifted lattices and can be expressed in terms of cosine modulation:

$$\begin{aligned} D_R(x,y) &= \frac{1}{4}(1+\cos(\pi x))(1+\cos(\pi y)) \\ D_G(x,y) &= \frac{1}{2}(1-\cos(\pi x)\cos(\pi y)) \\ D_B(x,y) &= \frac{1}{4}(1-\cos(\pi x))(1-\cos(\pi y)) \end{aligned} \tag{1.42}$$

The mosaiced image I_{CFA} in the Fourier domain is the Fourier transform of the (1.41):

$$\hat{I}_{CFA}(u,v) = \sum_{i\in\{R,G,B\}} \hat{C}_i(u,v) * \hat{D}_i(u,v) = \hat{R}_{CFA}(u,v) + \hat{G}_{CFA}(u,v) + \hat{B}_{CFA}(u,v) \tag{1.43}$$

where $*$ denotes the convolution operator, the $\hat{\cdot}$ represents the Fourier Transform, $\hat{R}_{CFA}$, $\hat{G}_{CFA}$ and $\hat{B}_{CFA}$ are Fourier Transform of the sub-sampled color components. The modulation functions defined in (1.42) are based on cosine and have their Fourier Transform expressed in Dirac. These transforms can be compactly arranged in a matricial form:

$$\hat{D}_i(u,v) = \Delta(u)^T M_{3x3} \Delta(v) \tag{1.44}$$

where

$$\Delta(u) = [\ \delta(u+0.5) \quad \delta(u) \quad \delta(u-0.5)\ ]^T$$

and

$$\Delta(v) = [\ \delta(v+0.5) \quad \delta(v) \quad \delta(v-0.5)\ ]^T$$

As expressed in (1.43), the Fourier Transform of the sub-sampled color channels can be derived by convolving the original $\hat{C}_i$ channels with the corresponding modulation functions $D_i(x,y)$; making the matrices of (1.44) explicit:

$$\begin{aligned}
\hat{R}_{CFA}(u,v) &= \hat{C}_R(u,v) * \left(\Delta(u)^T \begin{bmatrix} -\frac{1}{16} & \frac{1}{8} & -\frac{1}{16} \\ -\frac{1}{8} & \frac{1}{4} & -\frac{1}{8} \\ -\frac{1}{16} & \frac{1}{8} & -\frac{1}{16} \end{bmatrix} \Delta(v) \right) \\
\hat{G}_{CFA}(u,v) &= \hat{C}_G(u,v) * \left(\Delta(u)^T \begin{bmatrix} \frac{1}{8} & 0 & \frac{1}{8} \\ 0 & \frac{1}{2} & 0 \\ \frac{1}{8} & 0 & \frac{1}{8} \end{bmatrix} \Delta(v) \right) \\
\hat{B}_{CFA}(u,v) &= \hat{C}_B(u,v) * \left(\Delta(u)^T \begin{bmatrix} -\frac{1}{16} & -\frac{1}{8} & -\frac{1}{16} \\ \frac{1}{8} & \frac{1}{4} & \frac{1}{8} \\ -\frac{1}{16} & -\frac{1}{8} & -\frac{1}{16} \end{bmatrix} \Delta(v) \right)
\end{aligned} \tag{1.45}$$

This matrix representation is useful because it clarifies how the samples are scaled replications of the Fourier transform of the full resolution channels. The $\hat{G}_{CFA}$ formula in (1.45) points that the replications are placed on the diagonal directions only, while the $\hat{R}_{CFA}$ and $\hat{B}_{CFA}$ have replications also on the horizontal and vertical directions. The Fig.(**1.13**) shows the Fourier transform of the sub-sampled green channel and of the red and blue channels. In Fig.(**1.13(b)**) the spectrum of the whole I$_{CFA}$ is shown. It is evident the overlapping among the base band and the shifted replication, that is the cause of color artifacts.

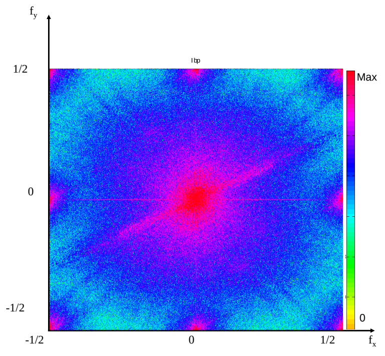
To overcome the aforementioned overlapping, Alleysson [10] started from the commonality between the human visual system (HVS) and the CFA based image sensors to sample one color only in each location (that is a pixels for imaging devices, a cone or a rod for the human eye) and thus spatial and chromatic information is mixed together. It is also known that the HVS encode the color information into luminance and opponent color signals. Similarly, for CFA sensors each color sample is composed by a spatial information due its position and chromatic information due to its spectral sensitivity. According to this representation, the (1.40) can be rewritten as:

$$I(x,y) = \{C_i(x,y)\} = \phi(x,y) + \{\psi_i(x,y)\} = \sum_{i \in \{R,G,B\}} p_i \cdot C_i(x,y) + \{\psi_i(x,y)\} \tag{1.46}$$

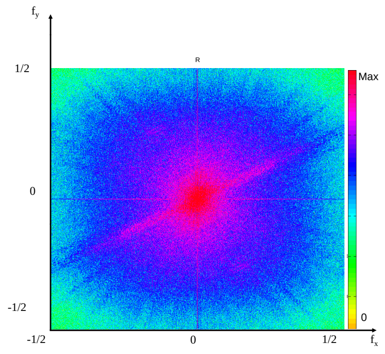
The spatial information, expressed by the scalar term ϕ, is composed by a weighted sum of each color channel, while $\{\ \psi_i\ \}$ is a vector of three opponent color components. Subtracting the luminance to the color image, the chrominance information is obtained. For CFA images the modulation functions can be rewritten as composed by a constant part p_i and by a fluctuation part with null mean $\tilde{D}_i$:

$$D_i(x,y) = p_i + \tilde{D}_i(x,y) \tag{1.47}$$

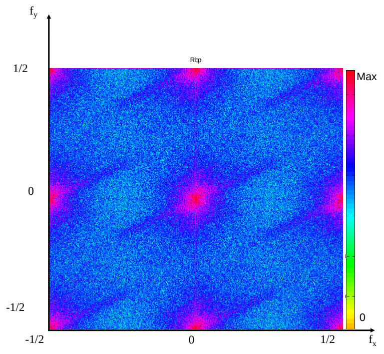
(a) Input image.



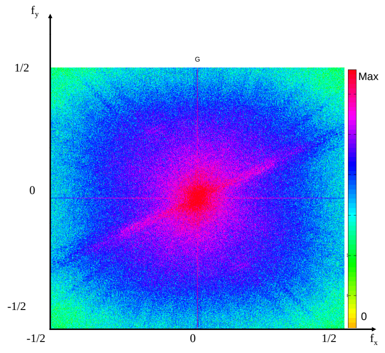
(b) Global CFA Spectrum.



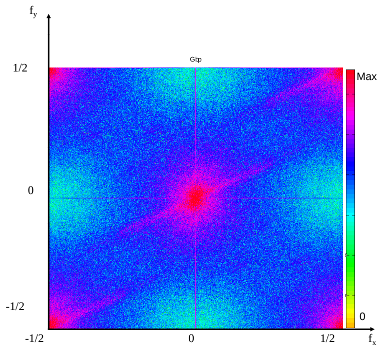
(c) RB original Spectrum.



(d) RB CFA Spectrum.



(e) G original Spectrum.



(f) G CFA Spectrum.

Figure 1.13 : Example of spectrum with relative channels.

The p_i represent the probability of presence of each color channel in the CFA. Since in the Bayer pattern the green components are twice the red and blue pixels, then $p_R = \frac{1}{4}$, $p_G = \frac{1}{2}$ and $p_B = \frac{1}{4}$. According to the (1.41) the I_{CFA} is now:

$$I_{CFA}(x,y) = \sum_{i \in \{R,G,B\}} p_i \cdot C_i(x,y) + \sum_{i \in \{R,G,B\}} C_i(x,y) \cdot \tilde{D}_i(x,y) \tag{1.48}$$

The first term represents the luminance and the second vectorial term represents the chromatic components. this representation highlights how the luminance term in (1.48) is the same of (1.46), that is the luminance in CFA images is exactly present and not subjected to interpolation even if it is subjected to aliasing with chrominances. Thus a good estimation of luminance information is fundamental. The localization of luminance is performed on the Fourier domain. Exploding the (1.43) with the equations in (1.45), it easily to rewrite $\hat{I}_{CFA}$ as:

$$\begin{aligned} &\hat{I}_{CFA}(u,v) = \sum_{i \in [R,G,B]} p_i \cdot \hat{C}_i(u,v) \\ &+\frac{1}{8} \sum_{k \in \{-0.5,0.5\}} \sum_{l \in \{-0.5,0.5\}} \left[\hat{C}_R(u-k,v-l) - \hat{C}_B(u-k,v-l)\right] \\ &+\frac{1}{16} \sum_{k \in \{-0.5,0.5\}} \sum_{l \in \{-0.5,0.5\}} \left[\hat{C}_R(u-k,v-l) - 2\hat{C}_G(u-k,v-l) + \hat{C}_B(u-k,v-l)\right] \end{aligned} \tag{1.49}$$

If we pose:

$$\begin{aligned} &\hat{L}(u,v) = \frac{1}{4}\hat{C}_R(u,v) + \frac{1}{2}\hat{C}_G(u,v) + \frac{1}{4}\hat{C}_B(u,v) \\ &\hat{C}_1(u,v) = \\ &\tfrac{1}{16} \textstyle\sum_{k \in \{-0.5,0.5\}} \sum_{l \in \{-0.5,0.5\}} \left[\hat{C}_R(u-k,v-l) - 2\hat{C}_G(u-k,v-l) + \hat{C}_B(u-k,v-l)\right] \\ &\hat{C}_2(u,v) = \frac{1}{8} \sum_{k \in \{-0.5,0.5\}} \sum_{l \in \{-0.5,0.5\}} \left[\hat{C}_R(u-k,v-l) - \hat{C}_B(u-k,v-l)\right] \end{aligned} \tag{1.50}$$

where $\hat{L}(u,v)$ is the luminance, $\hat{C}_1(u,v)$ and $\hat{C}_2(u,v)$ are the chrominance, the (1.49) becomes:

$$\hat{I}_{CFA}(u,v) = \hat{L}(u,v) + \hat{C}_1(u,v) + \hat{C}_2(u,v) \tag{1.51}$$

The relations in (1.50) can be expressed in matricial form:

$$\begin{bmatrix} \hat{L}(u,v) \\ \hat{C}_1(u,v) \\ \hat{C}_2(u,v) \end{bmatrix} = \begin{bmatrix} \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \\ -\frac{1}{4} & \frac{1}{2} & -\frac{1}{4} \\ -\frac{1}{4} & 0 & \frac{1}{4} \end{bmatrix} \cdot \begin{bmatrix} \hat{C}_R(u,v) \\ \hat{C}_G(u,v) \\ \hat{C}_B(u,v) \end{bmatrix} \tag{1.52}$$

The inverse of this matrix represents the relation between the RGB values in the CFA image and the luminance/chrominance signals in the Fourier domain.

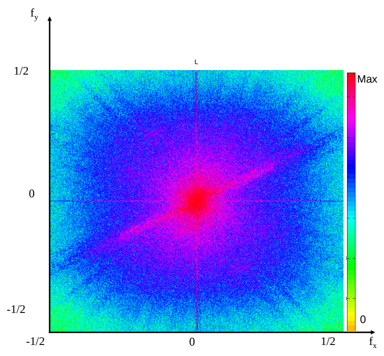
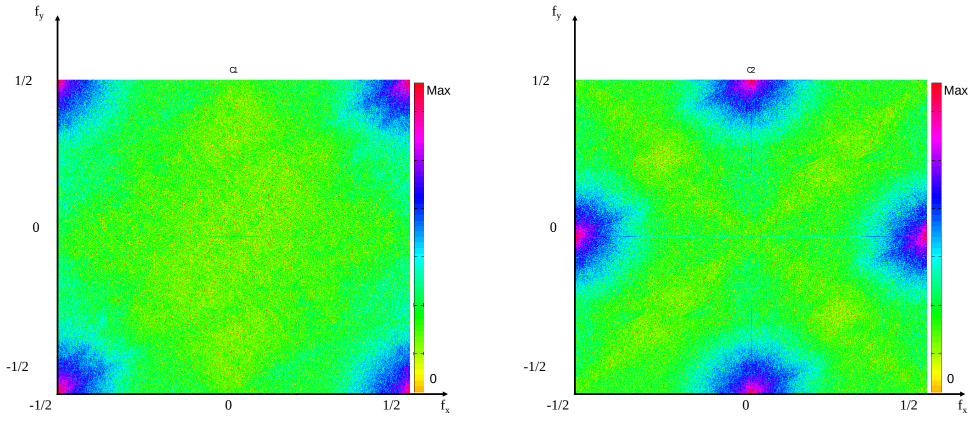


Figure 1.14 : Luminance Spectrum.

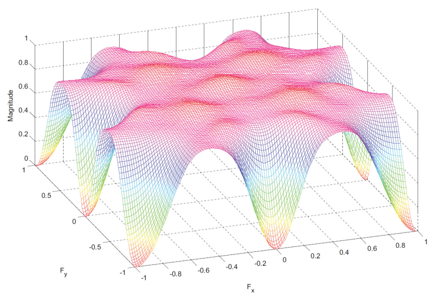
The spectrum of $\hat{L}(u,v)$ is not shifted, $\hat{C}_1(u,v)$ is located at the corner of the spectrum, while $\hat{C}_2(u,v)$ is located at the sides of the spectrum, as shown in Fig.(**1.14**) and Fig.(**1.15**). The smoothness of the color difference channels implies a more limited band for $\hat{C}_1(u,v)$ and $\hat{C}_2(u,v)$ and, consequently, the replication are more compact and less overlapping than the R and B subsampled channels. This allows to design better performing filters to discriminate luminance from the shifted bands than in the R, G and B representation.



(a) $\hat{C}_1(u,v)$ (b) $\hat{C}_2(u,v)$

Figure 1.15 : Chrominances spectrum.

Alleyson [10] proposed the filter in Fig.(**1.16**) to select the luminance. This filter is able to cut the frequencies where $\hat{C}_1$ and $\hat{C}_2$ are located, leaving the luminance unchanged, as can be seen in Fig.(**1.16(b)**). Other studies [9, 89] proposed different filters, depending on the adopted filter design methodology. The demosaicing algorithms based on this frequency analysis start estimating the luminance by filtering the $\hat{I}_{CFA}$ image.



$$\begin{bmatrix}
0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -1 & 0 & -2 & 0 & -1 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 2 & 1 & 2 & 1 & 1 & 0 & 0 \\
0 & -1 & 1 & -5 & 3 & -9 & 3 & -5 & 1 & -1 & 0 \\
1 & 0 & 2 & 3 & 1 & 7 & 1 & 3 & 2 & 0 & 1 \\
0 & -2 & 1 & -9 & 7 & 104 & 7 & -9 & 1 & -2 & 0 \\
1 & 0 & 2 & 3 & 1 & 7 & 1 & 3 & 2 & 0 & 1 \\
0 & -1 & 1 & -5 & 3 & -9 & 3 & -5 & 1 & -1 & 0 \\
0 & 0 & 1 & 1 & 2 & 1 & 2 & 1 & 1 & 0 & 0 \\
0 & 0 & 0 & -1 & 0 & -2 & 0 & -1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0
\end{bmatrix}$$

(a) (b)

Figure 1.16 : Selection filter proposed by Alleysson [10].

The luminance is then subtracted to the CFA image, obtaining the chrominance. The estimated chrominance is still subsampled and multiplexed. A further demulitiplexing step separates the chrominances in three channels, containing each component color where it is defined and zeros otherwise. The final step is the interpolation to recover the missing chrominance information. The interpolation can be simple because it is applied to smooth channels. The results of this interpolation are the difference channels *R-L*, *G-L* and *B-L*. The estimated Luminance is added to these channels to recover the *R, G* and *B* signals.

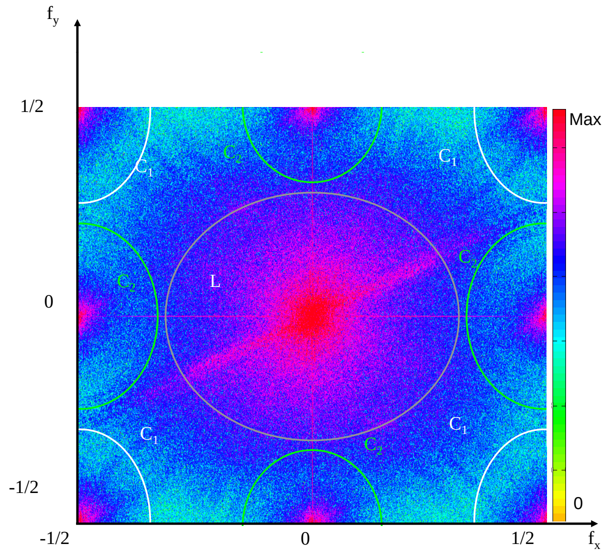


Figure 1.17 : Spectrum areas analysis.

Dubois [41] proposed an alternative demosaicing approach. It is based on the initial estimation of the chromatic components C_1 and C_2. These bands are located at the corners and sides of the Fourier spectrum, and can be isolated using bandpass filters. the first one, H_1 used to estimate C_1, is centered at frequency (0.5, 0.5):

$$\hat{C}_1(u,v) = \hat{I}_{CFA}(u,v) \cdot H_1(u,v) \tag{1.53}$$

The result of this filtering is shifted in base band. Analyzing in Fig.(**1.17**) the I_{CFA} spectrum, is noticeable that the crosstalk in CFA images is mainly in between luminance and C_2 components. Given multiple and shifted copies of the signal C_2, they can be exploited to better recover the original signal. I_{CFA} is filtered by two other bandpass filters H_{2A} and H_{2B}. The results are demodulated in baseband to estimate the C_{2A} and C_{2B}, the two sub-bands so that C_2=C_{2A} + C_{2B}, and placed on the vertical and horizontal axes of the spectrum. The more annoying artifact in demosaiced images is due to the crosstalk caused by the luminance energy near the frequencies ($\frac{1}{2}$, 0) and (0, $\frac{1}{2}$) where the modulated C_2 is present. In this case nothing can be done to perfectly separate the signals. However, the overlapping is often present in only one of the two bands. Leveraging on this behavior, the C_2 can be recovered as an adaptive sum of C_{2A} and C_{2B}, where the component suffering less from crosstalk is weighted more. The weighting function is modulated by a local estimation of the crosstalk, obtained analyzing the energy in two bands along the horizontal and vertical axes, the results are the local average energies e_x and e_y. The estimate of C_2 is obtained through a weighted sum:

$$C_2(x,y) = w(x,y)C_{2A}(x,y) + (1 - w(x,y))C_{2B}(x,y) \tag{1.54}$$

where

$$w = \frac{e_y}{e_x + e_y} \tag{1.55}$$

Once C_1 and C_2 have been estimated (in baseband), the luminance L is recovered by subtracting them to the I_{CFA} image. The inverse of the (1.52) matrix is at last applied to yield the R, G, B values.

Wavelets Based Algorithms

Another exploited research direction for demosaicing is in the wavelet domain [104]. In [135], to overcome the problem of quincoux G pattern, the G_r

(green samples on red rows) and G_b (green samples on blue rows) are separately processed, thus the color channels to be considered are four: R, B, G_r and G_b. Each plane has a dimension half respect to the original one. The interpolation processes these four planes and thus acts as a zooming [125]. The wavelets, as shown in Fig.(**1.18**) represent an image into sub-bands. The LL band contains the most of energy for the image signal. Ignoring the remaining bands, it is possible to reconstruct the image, where the zooming is mainly the antitransform of wavelets. In [135] an approach to interpolate the wavelet coefficients of the other sub-bands is proposed using a local spatial analysis to estimate the missing coefficients. This approach is based on the fact that, using the DWT5/3 (the same wavelets transform used in the JPEG2000 standard), if a coefficient has a value close to zero, the corresponding image region is smooth/ homogeneous. On the opposite, if the value is high, in the corresponding image region there is great variability. Thus a simple spatial correlation estimator of the input image values p_i is considered (in both horizontal and vertical direction):

$$\begin{array}{l} \Delta H = |p_{i,j} - p_{i,j+1}| \\ \Delta V = |p_{i,j} - p_{i+1,j}| \end{array} \tag{1.56}$$

Let consider the case of estimating the coefficients wav_coef_i of the HL band. If the value of ΔH is low, then the correspondent wavelets coefficient in HL is 0 because there is a high correlation between adjacent pixels. If this value is too high, there is a low correlation and the interpolation takes into consideration other pixels:

$$wav\_coef_i = f\left(p_{i,j}, p_{i,j-1}\right) \tag{1.57}$$

If the value is in a predetermined range the coefficients are found using correlated pixels:

$$wav\_coef_i = f\left(p_{i,j}, p_{i,j+1}\right) \tag{1.58}$$

This approach is the same for *LH* band, considering the ΔV threshold measures. The *HH* band is not interpolated due to the lower correlation of the input pixel values and the wavelets coefficients in this band. The demosaicing approach based on the coefficient interpolation is shown in Fig.(**1.18**).

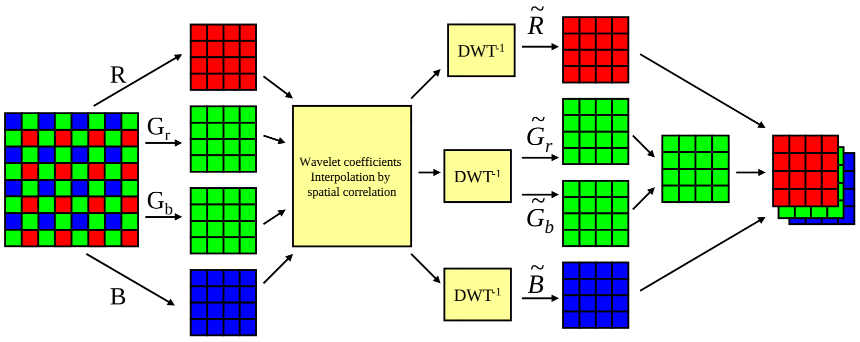


Figure 1.18 : Wavelets-based color interpolation.

This approach does not take into consideration any channel correlation, while another important demosaicing algorithm that uses the interchannel correlation has been proposed by Gunturk *et al.* [65]. Starting from the observation that on natural scenes all the three channels are very likely to have the same edge content, the authors show that the high frequencies subbands *LH*, *HL* and *LL* of each color are highly correlated. This can be expressed in the form

$$\begin{array}{l} |LH\left(K_{i,j}\right) - LH\left(G_{i,j}\right)| < threshold \\ |HL\left(K_{i,j}\right) - HL\left(G_{i,j}\right)| < threshold \\ |HH\left(K_{i,j}\right) - HH\left(G_{i,j}\right)| < threshold \end{array} \tag{1.59}$$

where K is the R or B channel in the CFA image The aliasing is removed using a Projection Onto Convex Sets (POCS) approach [114, 136]. The constraints set are based on the interchannel differences and on the observed data (original CFA pixels). This POCS method projects the initial estimate onto this constraint set to reconstruct the red and blue channels. The observation constraint set ensures that the interpolated color channels are consistent with the observed data; that is, the color samples captured by the sensor can not change during the reconstruction process. The projection onto the ”observation” constraint set is performed by inserting the observed data into their corresponding locations in the color channels at each iteration. The second constraint set imposes the similarity of high frequency of the color channels. The projection onto the ”detail” constraint set is performed by first decomposing the color channels into *LL*, *LH*, *HL*, *HH* subbands and then updating the high frequency subbands of the *R* and *B* channels only if the detail subbands of the difference planes *R*-*G* and *B*-*G* exhibit high values and, at last, restoring them with a bank of synthesis filters. This constraint is able to drastically reduce the aliasing.

A more detailed description of the whole demosaicing algorithm is the following:

1. An initial guess of the full color image is obtained using a simple linear interpolation;
2. The green channel is updated using the high frequencies of the red/blue channels:
   (a) The red and blue channels in the CFA image are a downsampled version of the full color image channels;
   (b) Consider a downsampled version of the green channel corresponding to all the interpolated data in the red and blue location separately;
   (c) Decompose the blue and corresponding green channel, then the red and corresponding green channel into subbands;
   (d) The *LH*, *HL* and *HH* subband of the two green channels are replaced

with the related subbands of the red and blue channels, that corresponds to set to 0 the *threshold* in the (1.59);

(e) Reconstruct the green channel through the inverse transform and place these new pixels in the initial guess of the green channel;

3. Iterate until a stop criterion is reached:

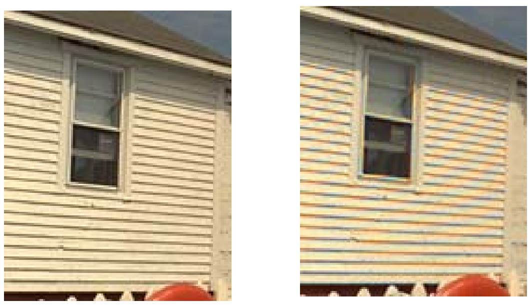
    (a) Decompose the channels and update the red and blue high frequency coefficients that do not verify the (1.59);

    (b) Reconstruct the red and blue channels, and replace the obtained values with the original ones (in the CFA image) at the red/blue location.

1.2 Techniques for Aliasing Correction

The two main types of demosaicing artifacts are false colors and zipper effect. False colors are those artifacts corresponding to noticeable color errors as compared to the original image. One example is shown in Fig.(**1.19(a)**), where the left hand is the full-color original image and the right hand is the demosaiced image with false colors. The zipper effect refers to abrupt or unnatural changes of color differences between neighboring pixels, manifesting as an ”on-off” pattern. One example is shown in Fig.(**1.19(b)**), where the left hand is the full-color image and the right hand is the demosaiced image with the zipper effect around the fence region.

An explanation of how the false colors arise in color interpolation is shown in Fig.(**1.20**). In Fig.(**1.20(a)**), a graphical representation of the light intensity distribution incident to an image sensing array comprising, for example 16 pixels, is depicted. For simplicity it will be assumed that the illumination comprises two colors A (filled circle) and B (square) wherein each color is defined by a


(a)

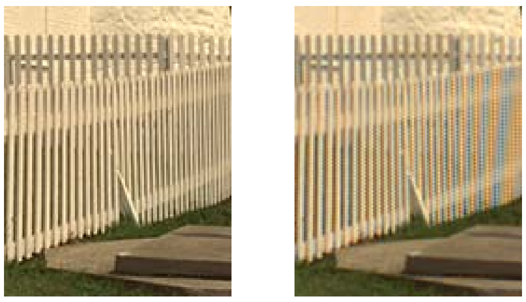
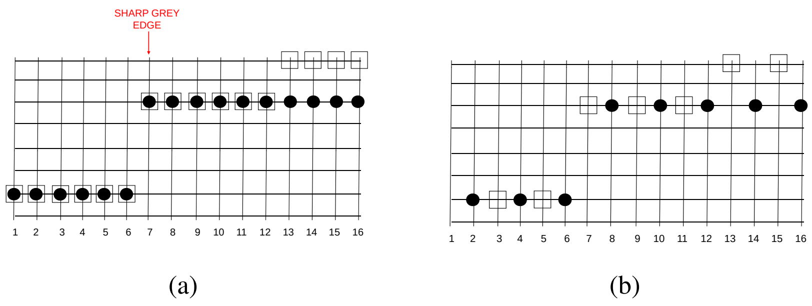

(b)

Figure 1.19 : Example of aliasing artifacts.

selected range of wavelengths different from the selected range of wavelengths which defines the other color. As depicted in Fig.(**1.20(a)**), the incident illumination defines a sharp grey edge between pixels 6 and 7 and a sharp grey to color transition between pixels 12 and 13. Fig.(**1.20(b)**) shows the case in which the illumination incident to an image sensing array having a filter arrangement in which alternate pixels are overlapped by filter transmitting either color A or color B. Thus, each pixel receives a single color of illumination, and linear interpolation between the pixels which sample each color provides the color distribution as shown graphically in Fig.(**1.20(c)**). From this figure, it is evident that the pixels 6 and 7 on each side of the grey edge no longer provide equal intensities for the colors A and B and thus will provide a highly visible color artifact or fringe in the reconstructed image.



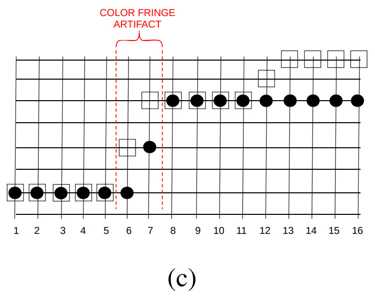


Figure 1.20 : Examples of an aliasing artifact caused by color interpolation.

From this visual example, we can derive that false colors arise if spectral correlation is not well exploited. This concept is also well disclosed in [33], where it is also explained that zipper effects manifest if spatial correlation is disregarded.

Although most of demosaicing solutions aim to eliminate false colors, some artifacts still remain. Thus imaging pipelines often include a post-processing module, with the aim of removing residual artifacts [92]. Post-processing techniques are usually more powerful in achieving false colors removal and sharpness enhancement, because their inputs are fully restored color images. Moreover, to fit some quality criteria, they can be applied more than once. For obtaining better performances, the antialiasing step often follows the color interpolation process, as a postprocessing step. The following subsections disclose a variety of state of the art techniques for false colors and zipper effects reduction.

1.2.1 False Colors Cancelling

Many techniques have been proposed in literature for reducing false colors. The conventional approach to solve this problem is to eliminate the color fringes at the expense of image sharpness by blurring the picture, so that the edges are not sharp enough to create a color fringe. Blurring the image in this manner, however, has its obvious disadvantages resulting in a reduction in resolution. Therefore it is necessary to provide a demosaicing artifact removal technique which reduces color fringing without the amount of blurring otherwise required. An interesting technique to solve color fringes without blurring the images was proposed by Freeman [52]. This approach starts from the consideration that in natural images there is a high correlation between the red, green and blue channel, especially for the high frequencies, so they are likely to have the same texture and edge locations. Because of this inter-channel (or spectral) correlation, the difference between two colors in a neighborhood is nearly constant, while it rapidly increases and decreases in the area of sharp grey edges, where color interpolation has introduced false colors. With reference to the example already shown in Fig.(**1.20**), Fig.(**1.21**) represents the difference between colors A and B for each pixel of Fig.(**1.20(c)**).

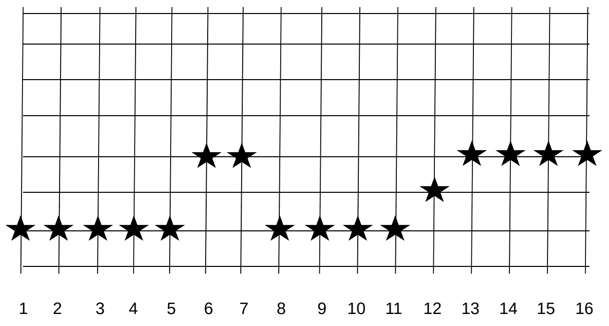


Figure 1.21 : Difference between colors A and B for each pixel.

The rapid increase and decrease in the difference between the two colors in pixels 6 and 7 is a characteristic of the objectionable color fringing and not simply a sudden rise in the difference between colors A and B as occurs after pixel 11 and which is indicative of a change from one color to a different color. Thus, it is not desirable to create such color spikes as a result of the method of interpolation chosen. A better estimate of the actual difference between the values for the colors A and B is provided by the graphical representation of Fig.(**1.22**), where the sharp peaks and valleys are removed and the other sharp transitions retained. Toward this goal a median filter, with a width of N pixels, can be used to replace each value in the graph of Fig.(**1.21**) with the median value of the nearest N pixels. For example, if the width of the median filter is selected to be five pixels, then the value at pixel 6 will become the median value of the pixels 4, 5, 6, 7 and 8.

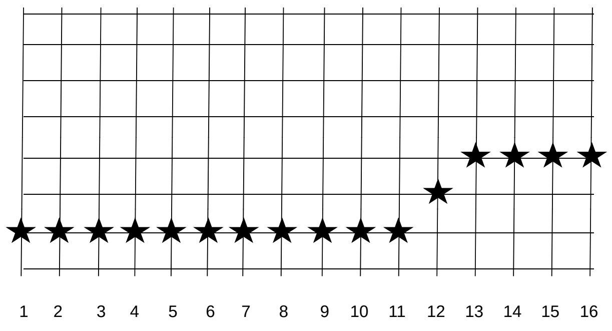


Figure 1.22 : Median filtered difference between colors A and B for each pixel.

Since the median values for each pixel are derived from the values of color A minus the values of color B for each pixel, subtracting the median values from the values of the color A provides the value of the color B for those pixels that receive only A colored light. Similarly, adding the median values to the values of the color B provides the value of the color A for those pixels that receive only B colored light. As depicted in Fig.(**1.23**), the Freeman's approach operates to actively reconstruct the sharp grey edge between pixels 6 and 7 while maintaining the color divergence starting at the pixel 11.

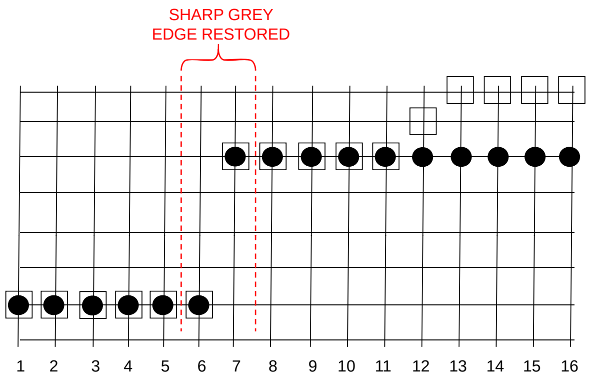


Figure 1.23 : New reconstruction from sampled data.

This method, in a three colors system, operates according to the following rules to obtain the values for the two missing colors of each pixel.

1. Pixels which receive only Red light:

$$\begin{aligned} \hat{R}(i,j) &= R(i,j) \\ \hat{G}(i,j) &= R(i,j) + v_{GR}(i,j) \\ \hat{B}(i,j) &= R(i,j) - v_{RB}(i,j) \end{aligned} \tag{1.60}$$

2. Pixels which receive only Green light:

$$\begin{aligned} \hat{R}(i,j) &= G(i,j) - v_{GR}(i,j) \\ \hat{G}(i,j) &= G(i,j) \\ \hat{B}(i,j) &= G(i,j) - v_{GB}(i,j) \end{aligned} \tag{1.61}$$

3. Pixels which receive only Blue light:

$$\begin{aligned} \hat{R}(i,j) &= B(i,j) + v_{RB}(i,j) \\ \hat{G}(i,j) &= B(i,j) + v_{GB}(i,j) \\ \hat{B}(i,j) &= B(i,j) \end{aligned} \tag{1.62}$$

where

$$v_{CD}(i,j) = median\{C(k,l) - D(k,l)\,|(k,l) \in H\} \tag{1.63}$$

H denotes the support of the $N \times N$ local window centered in (i,j), C and D denote two of the color channels. Analyzing the previous rules, is evident that the original CFA-sampled color value at each pixel is not altered, and it is combined with median-filtered inter-channel differences to obtain the other two missing color values. In general, Freeman's method is rather effective in suppressing demosaicing artifacts, while preserving sharp edges. However, some demosaicing artifacts, especially zipper effects, still remain around sharp edges and fine details. This is partly due to the fact that each pixel has independent inter-channel differences, and filtering the differences separately does not take into account the spectral correlation between color planes. To incorporate median filtering with the spectral correlation for more effective suppression of demosaicing artifacts, Lu and Tan's approach [92] lifts the constraint of keeping the original CFA-sampled color values intact. Furthermore, it makes use of the latest processed color values to filter the subsequent pixels so that estimation errors can be effectively diffused into local neighborhoods. Specifically, it adjusts the three color values at the central pixel of a local window (the window size is equal to the support of the median filter) as follows:

$$\begin{array}{l} \hat{G}(i,j) = \frac{(R(i,j)-v_{RG}(i,j))+(B(i,j)-v_{BG}(i,j))}{2} \\ \hat{R}(i,j) = \hat{G}(i,j) + v_{RG}(i,j) \\ \hat{B}(i,j) = \hat{G}(i,j) + v_{BG}(i,j) \end{array} \qquad (1.64)$$

This approach removes more false colors and artifacts than Freeman's method, but it considerably blurs images, because it adjusts the green channel of each pixel through an average of both the red and blue values of the same pixel. Another interesting technique, which is proposed in [84], updates the R, G, B values adaptively, modifying also the original pixel value which could be corrupted, due to the effect of noise. Two updated values for the green channel are calculated using each color difference domain:

$$\begin{array}{l} G^R(i,j) = R(i,j) + v_{GR}(i,j) \\ G^B(i,j) = B(i,j) + v_{GB}(i,j) \end{array} \tag{1.65}$$

where

$$\begin{array}{l} v_{GR}(i,j) = median\{G(k,l) - R(k,l)\,|(k,l) \in A\} \\ v_{GB}(i,j) = median\{G(k,l) - B(k,l)\,|(k,l) \in A\} \end{array} \tag{1.66}$$

and A denotes the support of the 5x5 local window centered in (i,j).

The updated G value is determined by the weighted sum of two updated G^R and G^B values of each color difference domain and original G value. Subsequently, R and B values are updated using the updated G value. This process is expressed as:

$$\begin{array}{l} \hat{G}(i,j) = \frac{1}{2}G(i,j) + \frac{1}{2}\left\{(1 - a(i,j))\,G^R(i,j) + a(i,j)\,G^B(i,j)\right\} \\ \hat{R}(i,j) = \frac{1}{2}R(i,j) + \frac{1}{2}\left\{\hat{G}(i,j) - v_{GR}(i,j)\right\} \\ \hat{B}(i,j) = \frac{1}{2}B(i,j) + \frac{1}{2}\left\{\hat{G}(i,j) - v_{GB}(i,j)\right\} \end{array} \tag{1.67}$$

where $a(i,j)$ is a weight, expressed as:

$$a(i,j) = \frac{\sigma^2_{(G-R)}(i,j)}{\sigma^2_{(G-R)}(i,j) + \sigma^2_{(G-B)}(i,j)}, 0 < a(i,j) < 1 \tag{1.68}$$

$\sigma^2_{(G-R)}$ and $\sigma^2_{(G-B)}$ represent the variances of interchannel differences.

As it is apparent from the (2.42), the color correction algorithm proposed in [84], thanks to the variance information, weights more the flatter color difference domain than the other. Moreover, the initially interpolated value is not totally exchanged by the updated value, but it is equally weighted for correction. Subsequently, the color values of the central pixel are replaced by $\hat{R}$, $\hat{G}$ and $\hat{B}$ so that they will be involved in filtering the updating pixels.

The local statistics are effectively estimated from a running square window as follows:

$$\begin{array}{l} E\left[A\left(i,j\right)\right]=\frac{\sum_{k,l\in A}e(k,l)\cdot A(k,l)}{\sum_{k,l\in A}e(k,l)} \\ \sigma_A^2\left(i,j\right)=\frac{\sum_{k,l\in A}e(k,l)\cdot(A(k,l)-E[A(i,j)])^2}{\sum_{k,l\in A}e(k,l)} \\ e\left(k,l\right)=1-\left(A\left(i,j\right)-A\left(k,l\right)\right) \end{array} \tag{1.69}$$

Such technique has the disadvantage of weighting the unfiltered values together with the filtered ones, so false colors are reduced, without being completely removed.

In [94], the authors propose to exploit the original uncorrupted Bayer CFA data, present in the demosaiced image, to correct erroneous color components produced by CFA interpolation with a localized color ratio model. This technique is based on the assumption that pixels with similar hues but different intensities should exhibit similar (if not identical) R/G and B/G color ratios. The post-processing procedure starts updating the green channel at the original R and B spatial locations as follows:

$$\hat{G}\left(i,j\right)=H\left(i,j\right)mean_{(k,l)\in\zeta}\left(\frac{G\left(k,l\right)}{H\left(k,l\right)}\right) \tag{1.70}$$

where H is the R or B central channel and

$$\zeta=\left\{\left(i-1,j\right),\left(i,j-1\right),\left(i,j+1\right),\left(i+1,j\right)\right\}. \tag{1.71}$$

In (1.70) the sampled G values are used together with the interpolated H values to obtain a local color ratio description G/R or G/B. The proposed local color ratio creates a model of the hue for the region under consideration and uses it to estimate the G component based on the original component H. It should be noted that this solution avoids extreme transitions in hue in the postprocessed images, thus reducing false colors. The second step consists in updating the R channel at B locations and the B channel at R locations, according to the 1.72

$$\hat{H}(i,j) = \hat{G}(i,j)\, mean_{(k,l)\in\zeta}\left(\frac{H(k,l)}{\hat{G}(k,l)}\right) \tag{1.72}$$

where $\zeta = \{(i-1,j-1),(i-1,j+1),(i+1,j-1),(i+1,j+1)\}$. The (1.72) differs from the (1.70) mainly because ζ is modified to take into account locations of the original R or B CFA data. In the last step of the algorithm R and B color values at G original locations are updated. The (1.72) is applied now using the original G values $G(i,j)$ and $\zeta = \{(i-1,j),(i,j-1),(i,j+1),(i+1,j)\}$. Moreover, two of the values $H(k,l)$ of (1.72) are original components and the other two are corrected components previously obtained using (1.72).

Some newer approaches address the problem of removing color artifacts in the YCrCb domain, instead of the classic RGB color space. In fact, if there is a strong edge in the R channel, there is usually a strong edge at the same location in the G and B channels; on the contrary, the YCrCb domain is less correlated, as demonstrated in [31]. Although edges still tend to be strong in the Y (luminance) plane, the chrominance planes (Cr and Cb) are smoother than the RGB plane, and hence they are more suitable for interpolation. The simplest way to remove color artifacts consists in correcting both the chrominance planes by simply blurring them. One liability with this approach is that there is no discrimination between false colors and genuine chrominance details. Consequently, sharp colored edges in the image begin to bleed color as the blurring becomes more aggressive. Adams *et al.* in [6] address the problem of eliminating low-frequency colored patterns, such as color Moiré, by filtering chrominances according to an activity value depending on the nearby luminance and chrominances. To remove spikes or valleys from these signals, which usually change smoothly, a median filter can be applied, instead of blurring the chrominance planes through an average filter. A median filter can remove false colors pretty well from the image edges, but it could introduce color bleeding artifacts in sharp colored edges. For this reason the technique proposed in [140] modifies chrominance values with respect to luminance and local chromatic dynamic ranges, in order to not reduce chromaticity too much in regions with uniform colors (see also [102]). The dynamic chromatic ranges (*DCr* and *DCb*) and the dynamic luminance range (*DY*) are evaluated in a 5x5 neighborhood of the pixel to be corrected. For each pixel of interest, dynamic luminance and chrominance ranges may be computed as the difference between the maximum and the minimum value in the local neighborhood, as follows:

$$\begin{array}{l} DY = max_I(Y) - min_I(Y) \\ DCr = max_I(Cr) - min_I(Cr) \\ DCb = max_I(Cb) - min_I(Cb) \end{array} \tag{1.73}$$

where *I* is the local neighborhood of the central pixel. Both the dynamic chromatic and luminance ranges are used to calculate a parameter, named *CorrectionFactor*, which determines the strength of the filter on chrominances. To remove possible false colors around sharp edges, the filtering action should be strong. On the contrary, if the luminance edge is weaker than both the chrominance edges, color bleeding has to be avoided by reducing the strength of the filter. Thus, the following equation is used to calculate this parameter:

$$CorrectionFactor = \begin{cases} DY & if\, DY = min_I(DY, DCr, DCb) \\ max_I(DY, DCr, DCb) & otherwise \end{cases} \tag{1.74}$$

The *CorrectionFactor* determines the power of the false colors correction, according to the following rule:

$$\begin{array}{l} Cr = medianCr_I + f(CorrectionFactor) \cdot (originalCr_I - medianCr_I) \\ Cb = medianCb_I + f(CorrectionFactor) \cdot (originalCb_I - medianCb_I) \end{array} \tag{1.75}$$

where *f(x)* is defined as:

$$f(x) = e^{-\frac{1}{2}\left(\frac{x}{sigma}\right)^2} \tag{1.76}$$

where *sigma* is a fixed parameter. The (1.75) updates each chrominance value with a weighted average of the original chrominance (*originalCr* and *originalCb*) and the median value of the chrominance in the neighborhood (*medianCr* and *medianCb*). The weights depend on the *CorrectionFactor* parameter, through the function *f(x)*. The function *f(x)* is the right part of a Gaussian function with expected value equal to zero and standard deviation equal to *sigma*. Fig.(**1.24**) illustrates the trend of the *f(x)*, for a given sigma value (*sigma* = 10).

The function *f(x)* rapidly decreases as the *x* increases, according to the (1.76). The value of the standard deviation *sigma* determines how fast *f(x)* approaches zero. With reference to the (1.75), low values of the *CorrectionFactor* imply a greater contribution of the original chrominance value; on the contrary, as the *CorrectionFactor* increases a higher weight is assigned to the median value. The function *f(x)* avoids discontinuous corrections when dynamic ranges change; in

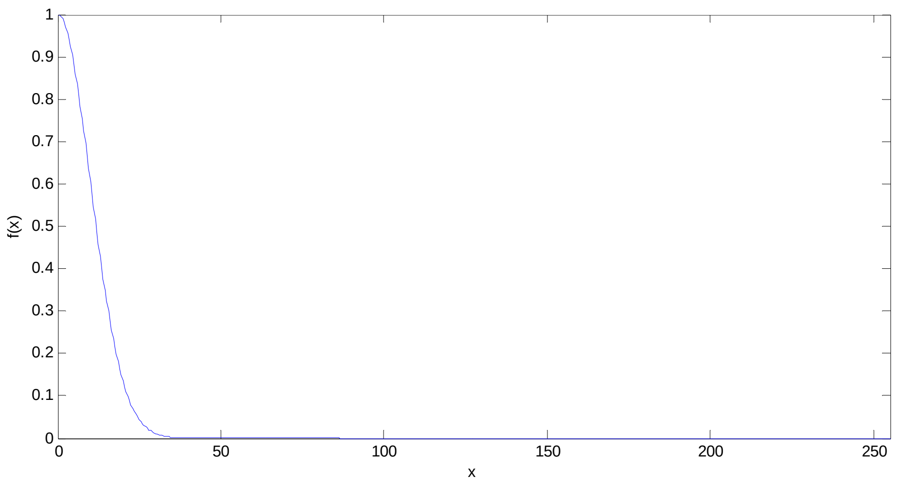


Figure 1.24 : Plot of *f(x)* vs. *x*, with *sigma* = 10.

fact proportions of both the original value and the median filtered value are continuously varied to form the final value. This soft-threshold methodology avoids abrupt transitions between corrected and non-corrected pixels.

Artifacts suppression can also be implemented in the frequency domain because various artifacts often occur in high-frequency components. More specifically, in [103] the authors propose to correct color artifacts by using the high bands inter-channel correlation of the three primary colors. Each pixel is separated in its low and high frequency components; then the high frequencies of the unknown components are replaced with the high frequencies of the Bayer-known component. The low-frequency component is preserved unchanged since the low-frequency components of the color channels are less correlated. For example, for a green pixel in the location (i,j), the green value can be decomposed as:

$$G(i,j) = G^l(i,j) + G^h(i,j) \tag{1.77}$$

where G^l and G^h denote the low and high frequency components of the G channel, respectively. R and B components at G locations are corrected according to the 1.78

$$\begin{aligned} \hat{R}(i,j) &= R^l(i,j) + G^h(i,j) \\ \hat{B}(i,j) &= B^l(i,j) + G^h(i,j) \end{aligned} \tag{1.78}$$

The correction at R and B original locations is performed in a similar way. The selection of the low-frequency components is performed using a low-pass filter while the high frequencies are calculated subtracting the low-frequency values. An effective approach is to select the low and high frequencies using a 1-D filter, so the interpolation is carried out only along the edges of the image.

1.2.2 Zipper Cancelling

Zipper effect is an artifact caused by a not correct spatial correlation exploitation. An artifact introduced by a wrong edge-estimation is usually difficult to remove in a post-processing phase, so it should be avoided during the interpolation step. Nevertheless, some techniques exist which reduce this effect. The simplest approach is the application of a heavy low pass filter to the demosaiced image. An example of antizipper filter is:

$$AZ = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \tag{1.79}$$

This filter removes zipper artifacts, but at the same time consistently reduces resolution thus removing genuine spatial details, which may not be recovered by further image processing algorithms.

In [63] the authors propose a method to eliminate the false color and zipper effect based on an adaptive scheme allowing to determine the specific artifact affecting the pixels. The authors use the spectral correlation between color planes to detect and reduce the artifacts. The block diagram representing the demosaicing artifact removal algorithm is shown in Fig.(**1.25**). Before processing each pixel, the zipper detector block produces a control signal which enables either the false colors removal algorithm or the zipper effect removal algorithm. More specifically, zipper effect arises when the following three conditions are satisfied:

1. Along the horizontal (vertical) direction passing through the central pixel, the inter-channel difference between the green channel and the other one (for which there are the original sampled values) is almost constant;
2. The trend of the central pixel channel in the vertical (horizontal) direction, the inter-channel difference between the green channel and the other one

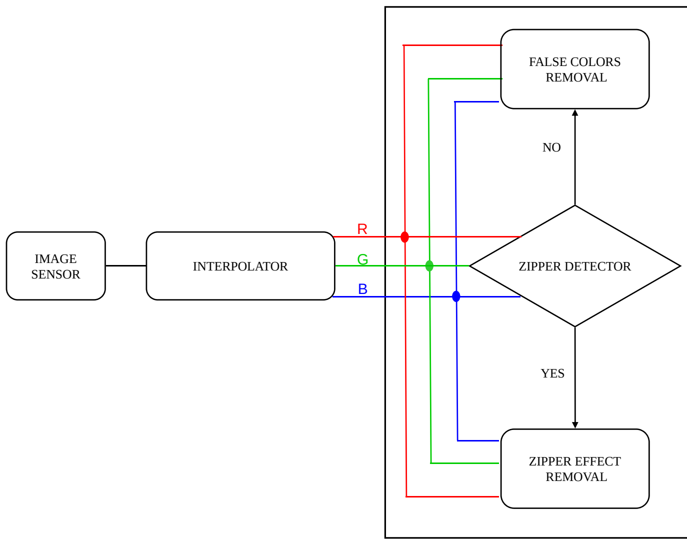


Figure 1.25 : Block diagram of the antialiasing algorithm proposed in [63].

(for which there are the original sampled values), is not constant;

3. The trend of the central pixel channel in the vertical (horizontal) direction is increasing or decreasing, or there is a minimum or a maximum in the central pixel.

Zipper effects are then removed using the following two equations:

1. G central pixel

$$\begin{aligned}\hat{G}(i,j) &= G(i,j)\\ \hat{H}(i,j) &= f(H_{SURROUND})\\ \hat{J}(i,j) &= G(i,j)+v_{JG}\end{aligned} \tag{1.80}$$

2. R/B central pixel (e.g., H)

$$\begin{aligned}\hat{G}(i,j) &= f(G_{SURROUND})\\ \hat{H}(i,j) &= H(i,j)\\ \hat{J}(i,j) &= \hat{G}(i,j)+v_{JG}\end{aligned} \tag{1.81}$$

where $f(\cdot)$ is an average operator, whose inputs are the surrounding pixels ($H_{SURROUND}$ or $G_{SURROUND}$) along the direction having almost constant inter-channel differences and v_{JG} is calculated using the (1.63).

The post-processing approach described in [95] is not only based on the color difference model, but also uses fully adaptive edge-sensing mechanism based on the aggregated absolute differences between the CFA inputs. The spectral correlation between the G and R (or B) components of the full-color image is utilized in the proposed post-processing process to further improve color appearance of the image. Based on the color difference model, the proposed post-processor reevaluates the G components produced by the demosaicking process as follows:

$$\hat{G}(i,j) = H(i,j) + \frac{\sum_{(k,l)\in\zeta} w(k,l)\left(G(k,l) - H(k,l)\right)}{\sum_{(k,l)\in\zeta} w(k,l)} \tag{1.82}$$

where $\zeta = \{(i-1,j),(i,j-1),(i+1,j),(i,j+1)\}$ denotes the locations of the original G components surrounding the interpolated location (i,j); $H(i,j)$ denotes the original R (or B) component at the position under consideration and $w(k,l)$ are the edge-sensing weights, which have to satisfy two conditions:

1. each weight is a positive number, $w(k,l) \geq 0$;
2. the summation of all the weights, $\sum_{(k,l)\in\zeta} w(k,l)$, is equal to unity.

More specifically, these weights are calculated as follows:

$$w(k,l) = \frac{1}{1 + d(k,l)} \tag{1.83}$$

where $d(k,l)$ is the aggregated absolute difference between the G sampled values:

$$d(k,l) = \sum_{(g,h)\in\zeta} \left|G(k,l) - G(g,h)\right| \tag{1.84}$$

These weights are used to regulate the contribution of the neighboring input components $G(k,l)$ in the (1.82). In fact, when no edge is positioned across the directions in which the image is post-processed, the corresponding aggregated absolute difference $d(k,l)$ is small and the CFA component $G(k,l)$, via its corresponding weight $w(k,l)$, contributes greatly in (1.82). The opposite is true in case of an edge. After the post-processing of the G channel is complete, the R (or B) component at B (or R) locations is post-processed as follows:

$$\hat{H}(i,j) = \hat{G}(i,j) + \frac{\sum_{(k,l)\in\zeta} w(k,l)\left(H(k,l) - G(k,l)\right)}{\sum_{(k,l)\in\zeta} w(k,l)} \tag{1.85}$$

The weights are computed as in (1.83), with $d(k,l) = \sum_{(g,h)\in\zeta} |H(k,l) - H(g,h)|$ where $\zeta = \{(i-1,j-1),(i-1,j+1),(i+1,j-1),(i+1,j+1)\}$.
Finally, the R and B components at G locations are processed. In this case the (1.85) is applied again with $\zeta = \{(i-1,j),(i,j-1),(i+1,j),(i,j+1)\}$ which are the locations of the R (or B) pixels surrounding the central G pixel.

2. An Adaptive Color Interpolation Technique

In this chapter we present a new adaptive demosaicing algorithm, with the aim of not magnifying the effect of noise, which is particularly visible in flat regions. The proposed technique decides among three different interpolation approaches, depending on the statistical characteristics of the neighborhood of the pixel to be interpolated. Edges are effectively interpolated through a directional filtering approach, which interpolates the missing colors selecting the suitable filter depending on the edge orientation. Regions close to edges are filtered through a simpler filter. Flat regions are identified and heavily low pass filtered in order to eliminate some residual noise and to minimize the green imbalance issue. A new powerful false colors removal algorithm has been also developed and used as a post-processing step, in order to eliminate residual color artifacts. The experimental results show how sharp edges are preserved, false colors and zipper effects are drastically reduced without accentuating noise.

2.1 Color Interpolation

As already outlined, the proposed color interpolation approach adaptively chooses the reconstruction methodology to recover the missing color information, upon the statistical characteristics of the surrounding pixels. As it is visible in Fig. 2.1 , the method is compound of three main steps:

1. The *direction estimation* stage computes the direction to be used in the interpolation step, if needed;

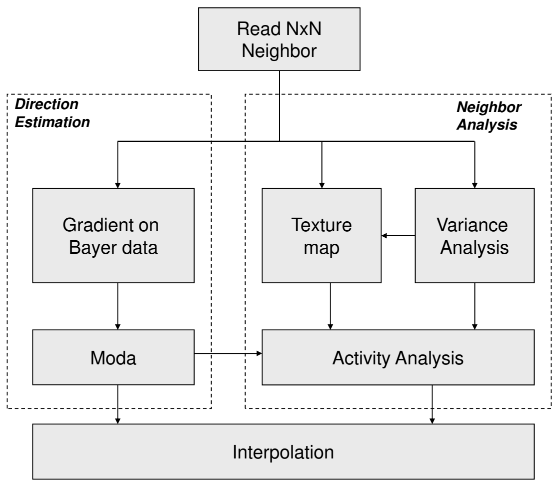


Figure 2.1 : Scheme of the color interpolation algorithm

2. The *neighbor activity analysis* stage selects which interpolation algorithm shall be performed, according to the surrounding features (edge, texture, flat area, etc.);
3. The *interpolation* stage executes a particular demosaicing algorithm, depending on the choice performed by the neighbor activity analysis block; the demosaicing algorithms could use the direction information provided by the direction estimation block for choosing the suitable filter to be applied. Moreover, the available demosaicing algorithms could have different kernel sizes.

Each phase of the algorithm will be extensively described in the following subsections.

2.1.1 Direction Estimation Block

The quality of demosaiced images is clearly dependent on the extracted gradient information from the input CFA image, but the information on mosaic image is not so accurate, as each pixel in the mosaic image only has one color channel, as stated in [37]. The direction estimation block is based on the one proposed in the patent [62], where the derivatives along both horizontal and vertical directions are computed through the classical 3x3 $Sobel_x$ and $Sobel_y$ filters, which are shown in equation (2.1).

$$Sobel_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad Sobel_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \tag{2.1}$$

Although these filters are widely used on full color images, their employment on Bayer images is not a commonly defined procedure. We introduced this technique after fixing the following assumptions. Let Q be the generic 3x3 neighborhood from the Bayer pattern as it is defined in (2.2):

$$Q = \begin{bmatrix} G_{11} & J_{12} & G_{13} \\ H_{21} & G_{22} & H_{23} \\ G_{31} & J_{32} & G_{33} \end{bmatrix} \tag{2.2}$$

where G_i are the green components and H_i and J_i are the generic red and/or blue components. In order to compute the derivatives only on the green channel, we exploit the spectral correlation property. In particular, as already mentioned, Gunturk et al. in [65] demonstrated that high frequency components of the three color planes are highly correlated. For instance, if it is assumed a red central pixel, the green component can be determined as:

$$G(i,j) = G_{LPF}(i,j) + R_{HPF}(i,j) \tag{2.3}$$

Where R_{HPF} is the high frequency content of the R channel, and G_{LPF} and R_{LPF} are the low frequency components of the G and R channels, respectively. Since the high frequency content of the red channel can be calculated as:

$$R_{HPF}(i,j) = R(i,j) - R_{LPF}(i,j) \tag{2.4}$$

The equation (2.3) can be rewritten as follows:

$$\begin{aligned} G(i,j) &= R(i,j) + G_{LPF}(i,j) - R_{LPF}(i,j) \\ &= R(i,j) + \Delta_{GR}(i,j) \end{aligned} \tag{2.5}$$

This implies that a *Q'* neighborhood, which is a matrix containing only *G* samples could be written as:

$$Q' = \begin{bmatrix} G_{11} & J_{12}+\Delta_{12} & G_{13} \\ H_{21}+\Delta_{21} & G_{22} & H_{23}+\Delta_{23} \\ G_{31} & J_{32}+\Delta_{32} & G_{33} \end{bmatrix} \tag{2.6}$$

Therefore the mathematical convolution between *Q'* and, for example, the derivative filter *Sobel*$_x$, becomes:

$$\frac{\partial Q'}{\partial x} = Q' \star Sobel_x = \begin{bmatrix} G_{11} & J_{12}+\Delta_{12} & G_{13} \\ H_{21}+\Delta_{21} & G_{22} & H_{23}+\Delta_{23} \\ G_{31} & J_{32}+\Delta_{32} & G_{33} \end{bmatrix} \star \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} =$$

$$= G_{13} + 2(H_{23}+\Delta_{23}) + G_{33} - G_{11} - 2(H_{21}+\Delta_{21}) - G_{31} \tag{2.7}$$

where Δ_{23} and Δ_{21} are unknown values. Since for real world images the color difference planes (Δ_{GR}=*G-R* and Δ_{GB}=*G-B*) are rather flat over small regions, the difference between Δ_{23} and Δ_{21} could be assumed irrelevant, and thus:

$$\frac{\partial Q'}{\partial x} \approx \frac{\partial Q}{\partial x} = G_{13} + 2H_{23} + G_{33} - G_{11} - 2H_{21} - G_{31} \tag{2.8}$$

A similar condition obviously holds for the computation of *Sobel*$_y$. This means that Sobel filters can be applied directly on Bayer images.

Unfortunately, if the color planes are not correlated, e.g. red to blue transitions (as shown in Fig.2.2), the spectral correlation property is not valid [85], and the just disclosed procedure for computing horizontal and vertical gradients could fail. To solve this issue, we developed a method to establish if the color channels are not correlated, directly acting on the Bayer pattern. Let M be the generic 3x3 window, which is shown in (2.9):

$$M = \begin{bmatrix} M_{NW} & M_N & M_{NE} \\ M_W & M_C & M_E \\ M_{SW} & M_S & M_{SE} \end{bmatrix} \tag{2.9}$$

For each direction (horizontal and vertical) we define two different gradients, external (*ext*) and central (*cnt*), each of which involving a single bayer channel, as it is apparent from equation (2.10):

$$\begin{aligned} &\frac{\partial M}{\partial x}ext = M_{NE} + M_{SE} - M_{NW} - M_{SW};\ \frac{\partial M}{\partial x}cnt = M_E - M_W \\ &\frac{\partial M}{\partial y}ext = M_{NE} + M_{NW} - M_{SE} - M_{SW};\ \frac{\partial M}{\partial y}cnt = M_N - M_S \end{aligned} \tag{2.10}$$

The lack of correlation between color channels is evaluated in two different ways according to the central pixel channel (G or not G). If the central pixel is green, the equation (2.11) is applied:

$$NoCorrelation(M) = \begin{cases} 1 \quad if & \left(\begin{array}{c} SIGN\left(\frac{\partial M}{\partial x}ext\right) \neq SIGN\left(\frac{\partial M}{\partial x}cnt\right)\ OR \\ SIGN\left(\frac{\partial M}{\partial y}ext\right) \neq SIGN\left(\frac{\partial M}{\partial y}cnt\right) \end{array} \right) \\ 0 & otherwise \end{cases} \tag{2.11}$$

Otherwise, if the central pixel is red or blue the equation (2.12) is applied :

$$NoCorrelation\left(M\right)=\begin{cases} 1 \quad if & \left(\begin{array}{c} \left(\begin{array}{c} SIGN\left(\frac{\partial M}{\partial x}ext\right)\neq SIGN\left(\frac{\partial M}{\partial x}cnt\right)\ AND \\ SIGN\left(\frac{\partial M}{\partial y}ext\right)\neq SIGN\left(\frac{\partial M}{\partial y}cnt\right) \end{array}\right) \\ OR \\ \left(\begin{array}{c} \left|\frac{\partial M}{\partial x}ext\right|>2\left|\frac{\partial M}{\partial x}cnt\right|\ AND \\ \left|\frac{\partial M}{\partial y}ext\right|>2\left|\frac{\partial M}{\partial y}cnt\right| \end{array}\right) \end{array}\right) \\ 0 & otherwise \end{cases} \tag{2.12}$$

where

$$SIGN\left(x\right)=\begin{cases} 1 & if\ x\geq 0 \\ -1 & otherwise \end{cases} \tag{2.13}$$

The difference between (2.11) and (2.12) is due to the different arrangement of color channels between G and non-G central pixel cases. In fact, if the central pixel is green the horizontal central derivative has the information about the blue (or red) channel, whereas the vertical central derivative has the information about the red (or blue) channel, so the evaluation of cross-correlation could use only simple sign criteria. In particular, equation (2.11) has the aim of evaluating if at least either green-red or green-blue are opposite correlated. On the contrary, if the central pixel is not green, both horizontal and vertical external derivatives have the information of the same color channel, and hence more complex criteria can be used to properly evaluate cross-correlation. More specifically, in equation (2.12) the first term of the OR expression is used to evaluate if there is an opposite correlation between green and the color channel which has its samples at the corners of the 3x3 mask; the second term is used to evaluate the lack of correlation between green and the other color channel even if both the external and the central gradients have the same sign. This could happen if the green channel is quite flat with respect to the other color channel. In this case the classic Sobel operators could fail in providing a good gradient estimation.

Let *SimpleGrad*$_x$ and *SimpleGrad*$_y$ be two simpler gradient filters, defined in equation (2.14):

$$SimpleGrad_x = \begin{bmatrix} -1 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}; \qquad SimpleGrad_y = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & -1 \end{bmatrix} \tag{2.14}$$

These gradient operators could be used every time a lack of correlation between the three color channels is suspected, because they involve a single color channel, and hence they are not negatively influenced by the different variations of the R, G and B channels. Thus, the gradient for the central pixel of the Q neighborhood could be calculated according to the following equation :

$$\nabla Q = \left(\frac{\partial Q}{\partial x}, \frac{\partial Q}{\partial y}\right) \begin{cases} (Q \star Sobel_x, Q \star Sobel_y) & if\ NoCorrelation(Q) = 0 \\ (Q \star SimpleGrad_x, Q \star SimpleGrad_y) & if\ NoCorrelation(Q) = 1 \end{cases} \tag{2.15}$$

The equations (2.11) and (2.12) have especially the aim to identify the cases where the Bayer channels are not correlated, to avoid the application of Sobel operators, which involve different color channels. On the contrary, even if the simpler gradient operators were applied on a region where the Bayer channels are correlated this would not be a problem, because they provide a quite good gradient estimation, even if Sobel operators are better. Moreover, the final gradient estimation for the central pixel depends on the gradients within its 3x3 neighborhood, as it will be further explained. Once the gradient is computed, its direction and magnitude are calculated by the (2.16) and (2.17), respectively.

$$\psi(\nabla Q) = \begin{cases} arctg\left(\frac{\partial Q/\partial y}{\partial Q/\partial x}\right) & if\, \partial Q/\partial x \neq 0 \\ \frac{\pi}{2} & otherwise \end{cases} \tag{2.16}$$

$$|\nabla Q| = \sqrt{\left(\frac{\partial Q}{\partial x}\right)^2 + \left(\frac{\partial Q}{\partial y}\right)^2} \tag{2.17}$$

It is important to specify that $\psi(\nabla Q)$ and $|\nabla Q|$ represent the direction and magnitude of the gradient of the central pixel of Q. If we set *(x,y)* as the coordinates of the central pixel of Q, the direction and magnitude associated to its gradient could be indicated as:

$$\begin{array}{l} or\,(x,y) = \psi(\nabla Q) \\ mag\,(x,y) = |\nabla Q| \end{array} \tag{2.18}$$

The gradient orientation *or(x,y)* could be quantized into k predefined directions:

$$direction_i = \frac{i \cdot \pi}{k} \quad i \in [0, k-1],\; k \in \aleph \tag{2.19}$$

and, according to the equation (2.19), the gradient orientation $\overline{or}(x,y)$ is set as follows:

$$\overline{or}(x,y) = \{direction_i | direction_i \leq or(x,y) < direction_{i+1}, i \in [0, k-1]\} \tag{2.20}$$

To avoid the influence of noise on the gradient estimation, a “weighted-mode” (WM) operator is applied on each pixel (x,y) computing the prominent direction in its 3x3 neighborhood. For estimating this direction, the magnitudes of the pixels in the neighborhood are firstly accumulated according to their associated directions:

$$Acc\,(x,y,i) = \sum_{u=-1}^{1}\sum_{v=-1}^{1} mag(x+u,y+v)\cdot t(x+u,y+v,i) \tag{2.21}$$

where

$$t\,(x,y,i) = \begin{cases} 1 & if\,\overline{or}\,(x,y) = direction_i \\ 0 & otherwise \end{cases} \tag{2.22}$$

for $i\in$ *[0, k-1], k*$\in$ ℵ

Therefore the WM operator selects the direction associated to the maximum sum of magnitudes in the neighborhood of the considered pixel:

$$WM(x,y) = j \text{ such that } Acc\,(x,y,j) = \max_{i=0..k-1}\left(Acc\,(x,y,i)\right),\, j\in[0,k-1] \tag{2.23}$$

Finally, after the gradient direction is retrieved, through the process already described, the edge direction is obtained by taking the orthogonal direction to the gradient one. The edge direction is provided to the neighbor analysis block for the texture analysis, and to the interpolation block when the directional color interpolation is selected.

2.1.2 Neighbor Analysis

The aim of this part of the proposed method (whose detailed functional scheme is shown in Fig. 2.3) is to evaluate the presence of texture, edges or flat zones in the pixel neighbor. This analysis is performed by estimating the variances very close to the pixel and on a larger surrounding. The comparison of these variances provides useful information about neighbor activity. In case of textured zones, an additional analysis is necessary, in order to validate the texture

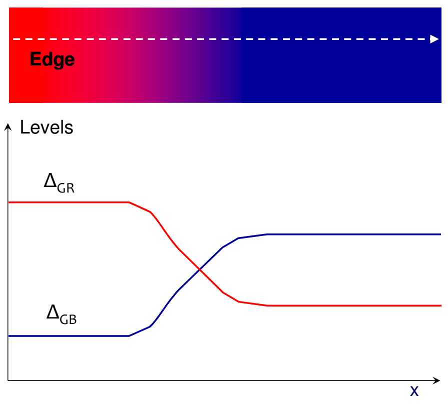


Figure 2.2 : Uncorrelated planes: example of red to blue transitions.

direction estimated by the *Direction Estimation Block*. According to the results of this analysis, the appropriate interpolation scheme is then used. For our purposes we used two kernel sizes: 5x5 for the larger surrounding analysis and 3x3 for the analysis close to the pixel. The variance is computed, as a measure of the activity within a neighborhood. In particular, let P and Q be the generic 5x5 and 3x3 windows from the Bayer pattern, respectively.

$$P = \begin{bmatrix} G_{11} & H_{12} & G_{13} & H_{14} & G_{15} \\ J_{21} & G_{22} & J_{23} & G_{24} & J_{25} \\ G_{31} & H_{32} & G_{33} & H_{34} & G_{35} \\ J_{41} & G_{42} & J_{43} & G_{44} & J_{45} \\ G_{51} & H_{52} & G_{53} & H_{54} & G_{55} \end{bmatrix} \qquad Q = \begin{bmatrix} G_{22} & J_{23} & G_{24} \\ H_{32} & G_{33} & H_{34} \\ G_{42} & J_{43} & G_{44} \end{bmatrix} \tag{2.24}$$

where G_i are the green components and H_i and J_i are the generic red and/or blue components. The mean value and the variance, for a generic neighborhood S of the central pixel, are computed according to the following equations:

$$\begin{aligned}
\mu_X(S) &= \frac{\sum_{(i,j)\in S}\Big(S(i,j)*Mask_X(i,j)\Big)}{\sum_{(i,j)\in S} Mask_X(i,j)} \\
\sigma_X^2(S) &= \frac{\sum_{(i,j)\in S}\Big(S(i,j)*Mask_X(i,j)-\mu_X(S)\Big)^2}{\sum_{(i,j)\in S} Mask_X(i,j)} \\
\sigma^2(S) &= \left(\sigma_G^2(S)+\sigma_H^2(S)+\sigma_J^2(S)\right) \\
S &\in \{P,Q\} \quad X \in \{G,H,J\}
\end{aligned} \tag{2.25}$$

where $Mask_X$ identifies the involved pixels components present in the pattern of the processed color component. Let Λ_G, Λ_H and Λ_J be the set of pixel locations, *(x,y)*, that have the samples of green, red (blue) and blue (red) channels, respectively. The binary mask *MaskX* is defined as:

$$\mathrm{Mask}_\mathrm{X}(x,y) = \begin{cases} 1 & if \ \ (\mathrm{x},\mathrm{y}) \in \Lambda_\mathrm{X} \\ 0 & otherwise \end{cases} \tag{2.26}$$

where $X = G, H\, or\, J$.

By using the equations in (2.25) on the neighborhoods P and Q, defined by the equation (2.24), two values are calculated, $\sigma^2(P)$ and $\sigma^2(Q)$, which represent a measure of the activity in the relative neighborhoods.

Two thresholds are set for distinguishing different conditions. With reference to Fig. 2.3 , if both the variance values are under the lower threshold, the region is considered flat, and hence a low pass filter based interpolation could be used, in order to remove residual noise. This threshold could be either fixed or modulated by the noise sigma (if computed or estimated). It allows to blur more the flat zones, where the Human Visual System is more sensitive to noise.

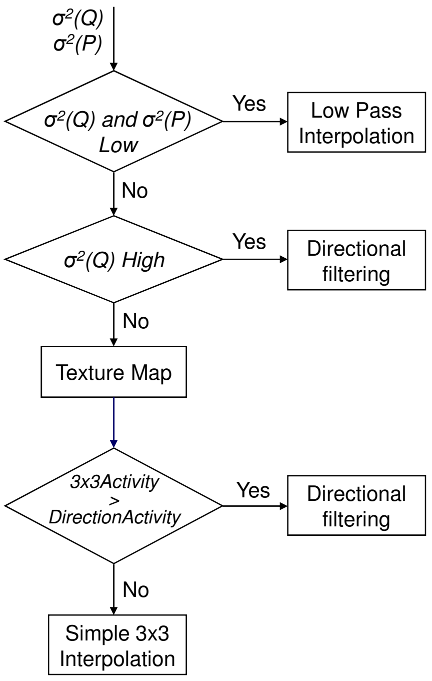


Figure 2.3 : Neighbor analysis block scheme

When the variances have values not too high or low, the evaluation is not straightforward, because it could be due either to a zone nor flat nor textured, or to a zone with a texture without strong edges. In the first case a directional interpolation could introduce artifacts, while in the second one the directional interpolation could highlight the texture.

The mean values and the standard deviations of each color channel, within the P neighborhood, are used to build a texture Mask (we call it TLM, three level mask), which is a matrix having the same size as P and having each element calculated according to the equation (2.27).

$$TLM(i,j) = \begin{cases} 0 & if\ P(i,j) < (\mu_X(P) - \sigma_X(P)) & with\ (i,j) \in \Lambda_X \\ 1 & if\ (\mu_X(P) - \sigma_X(P)) < P(i,j) < (\mu_X(P) + \sigma_X(P)) & with\ (i,j) \in \Lambda_X \\ 2 & if\ P(i,j) > (\mu_X(P) + \sigma_X(P)) & with\ (i,j) \in \Lambda_X \end{cases} \quad X = \{G,R,B\} \tag{2.27}$$

where $\sigma_X(P) = \sqrt{\sigma_X^2(P)}$.

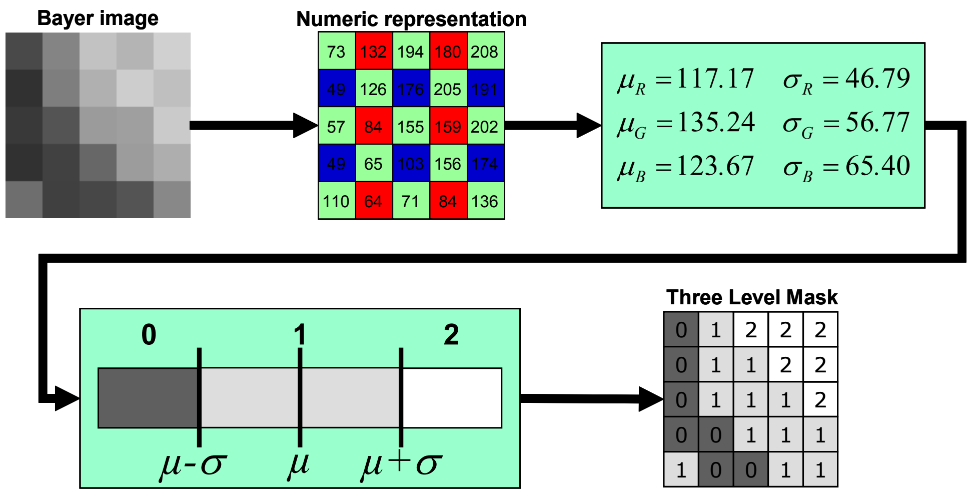


Figure 2.4 : Example of the Three Level Mask (TLM) generation

Each value within the current *P* window is associated, in the TLM, with one of three possible levels: Low (value 0), Medium (value 1) or High (value 2), according to its position with respect to its mean value and its standard deviation. The generation of the TLM is illustrated in Fig. 2.4. In the top left of the figure, a 5x5 crop of a Bayer image is represented, then its numeric representation is shown. The mean values and the standard deviations of each color channel are reported in a box.

Let us define *"Activity"* the sum of absolute differences of contiguous elements of the TLM within a certain working mask.

Once the TLM is generated, two working masks are taken into account and hence two activity values are calculated accordingly. The first working mask is the same for each pixel: it is the squared 3x3 mask centered in the pixel under consideration. The second one depends on the direction provided by the direction estimation block for the current pixel: it includes the pixels along the identified direction. It is obvious that a greater activity within a certain working mask represents a lower uniformity of the image content in it. Fig. 2.5 shows an example

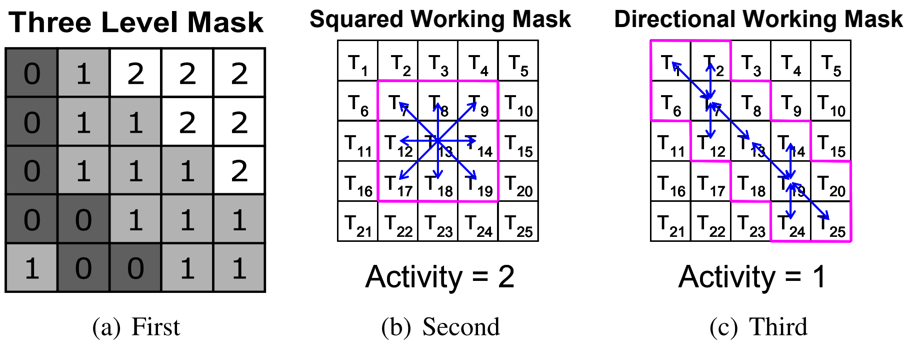


Figure 2.5 : Example of the Activity evaluation assuming that the estimated direction is 135°.

of activity computation. Fig. 2.5(a) illustrates the TLM already presented in fig. 2.4; in fig. 2.5(b) the squared working mask is highlighted in pink. Finally, in this example the edge direction is estimated to be 135°, and the correspondent directional working mask is highlighted in Fig. 2.5(c).

Let T_i represent the generic element of the TLM, the activities within the 3x3 squared mask and within the 5x5 directional mask are calculated through the equations (2.28) and (2.29).

$$\begin{aligned} 3\times 3 Activity &= |T_7-T_{13}|+|T_8-T_{13}|+|T_9-T_{13}|+|T_{14}-T_{13}| \\ &+ |T_{19}-T_{13}|+|T_{18}-T_{13}|+|T_{17}-T_{13}|+|T_{12}-T_{13}| \end{aligned} \tag{2.28}$$

$$\begin{aligned} DirectionActivity(135^\circ) &= |T_1-T_7|+|T_2-T_7|+|T_7-T_{12}|+|T_7-T_{13}| \\ &+ |T_{13}-T_{19}|+|T_{14}-T_{19}|+|T_{19}-T_{24}|+|T_{19}-T_{25}| \end{aligned} \tag{2.29}$$

The 5x5 directional working mask and its related activity computation formula depend on the edge direction, which is provided by the direction estimation block. Finally the activity values computed in equations (2.28) and (2.29) are compared to evaluate if the activity within the 3x3 squared mask is less than the activity along the direction provided by the direction estimation block. If this is the case, a simple 3x3 interpolation could be used. This control allows avoiding the directional 5x5 interpolation in case of a wrong estimated direction, or in case of a region close to an edge, thus reducing the introduction of unpleasant artifacts. In the example of fig. 2.4, the directional interpolation is chosen for the central pixel, because the 3x3 squared mask activity (Fig. 2.5(b)) is greater than the 5x5 directional one (Fig. 2.5(c)).

For the sake of completeness, the activity computation formulas applied in case of edge direction equal to 0° and 22.5° are shown in equations (2.30) and (2.31), respectively. All the other activity computation formulas can be easily derived from the three examples already presented.

$$\begin{aligned} DirectionActivity(0^\circ) &= |T_6 - T_8| + |T_8 - T_{10}| + |T_{11} - T_{12}| + |T_{12} - T_{13}| \\ &+ |T_{13} - T_{14}| + |T_{14} - T_{15}| + |T_{16} - T_{18}| + |T_{18} - T_{20}| \end{aligned} \tag{2.30}$$

$$\begin{aligned} DirectionActivity(22.5^\circ) &= |T_9 - T_{10}| + |T_{11} - T_{12}| + |T_{12} - T_{13}| + |T_{13} - T_{14}| \\ &+ |T_{14} - T_{15}| + |T_{16} - T_{17}| + |T_{13} - T_{10}| + |T_{13} - T_{16}| \end{aligned} \tag{2.31}$$

2.1.3 Interpolation

As already pointed out, according to the analysis of the ”Direction Estimation” and ”Neighbor Analysis”, the appropriate color interpolation scheme is used. in particular we use:

- 5x5 directional filter;
- 3x3 simple interpolation;
- 5x5 omnidirectional low pass filter.

In the following paragraphs each one of the aforementioned algorithms will be described.

2.1.3.1 Directional filtering color interpolation

If a strong edge or a texture is detected, the directional filtering is used. This interpolation is carried out through elliptical shaped Gaussian filters, given by:

$$f(u,v,\alpha) = he^{-\frac{\tilde{u}^2}{2\sigma_u^2}-\frac{\tilde{v}^2}{2\sigma_v^2}} \tag{2.32}$$

where

$$\begin{aligned}\tilde{u} &= u\cos(\alpha) - v\sin(\alpha),\\ \tilde{v} &= u\sin(\alpha) + v\sin(\alpha),\end{aligned} \tag{2.33}$$

and σ_u^2, σ_v^2 are the variances along the two dimensions, h is a normalization constant and α is the orientation angle. Through heuristic experiments $\sigma_u = 8$ and $\sigma_v = 0.38$ have been fixed.

These interpolation kernels can be computed only once, after the number of admissible directions, k, has been set. More specifically, fixing the coordinates *(u,v)* in a given range (i.e., 5x5), the kernel filters DF_i can be generated, for $i \in [0, k-1]$. The generic element of these matrices, *DF_i(u,v)*, is defined as follows:

$$DF_i(u,v) = f\left(u, v, i \cdot \pi/k\right) \tag{2.34}$$

During the directional filtering color interpolation, once the direction j for the central pixel is derived by the direction estimation block, the kernel DF_j is applied to the central pixel to obtain the low pass filter (LPF) color components R_{LPF_DF}, G_{LPF_DF} and B_{LPF_DF}, preserving the image from zigzag effect and partially from false colors.

Let P be the generic 5x5 neighborhood from the Bayer pattern as it is defined in equation (2.24).

The LPF components, for the central pixel (x,y), are computed by assuming the following rules:

$$\begin{aligned} H_{LPF\_DF} &= P \otimes Mask_H \otimes DF_j \\ G_{LPF\_DF} &= P \otimes Mask_G \otimes DF_j \\ J_{LPF\_DF} &= P \otimes Mask_J \otimes DF_j \end{aligned} \tag{2.35}$$

where $Mask_X$, which has been already defined in equation (2.26), identifies the pixels belonging to the processed color component.

As already aforementioned, the high frequency components of the three color channels are highly correlated, and hence any color component can be exploited to reconstruct the high frequencies of the remaining color components. For this reason, the directional LPF component of the central pixel is likewise computed in order to calculate its high pass filter (HPF) component, which will be added to the other two color channels. Assuming a green central pixel, its high pass component can be calculated according to the equation (2.36).

$$G_{HPF} = G - G_{LPF\_DF} \tag{2.36}$$

The resulting G_{HPF} value is added to the unknown values, in this case R and B (e.g. H), to increase their high frequency content:

$$H_{DF} = H_{LPF\_DF} + G_{HPF} \tag{2.37}$$

The central pixel value will be maintained to its original value:

$$G_{DF} = G \tag{2.38}$$

2.1.3.2 Simple 3x3 interpolation

The 5x5 directional filtering color interpolation could produce some artifacts near edges, especially in text images. This problem could arise when a wrong direction is estimated or when a large interpolation window is used for interpolating a small region, enclosed by edges. To overcome this problem a 3x3 interpolation could be performed in regions which are close to edges.

A wide variety of 3x3 interpolation algorithms exists: bilinear interpolation, edge sensing, etc. In order to achieve good quality results near corners and edges, it is possible to use an algorithm which analyzes the 5x5 neighborhood of the central pixel, and then applies a 3x3 interpolation, avoiding the interpolation across edges.

Taking idea from the paper in [32], the proposed 3x3 interpolation algorithm uses a threshold-based variable number of gradients. According to the central pixel channel, four (G case) or eight (R or B case) gradients are computed in a 5x5 neighborhood. Each gradient is defined as the sum of absolute differences of the like-colored pixels in this neighborhood. For each color channel to be interpolated, a proper subset of these gradients is selected to determine a threshold of acceptable gradients. Each missing color channel is then obtained by averaging the color values of the same channel in the 3x3 neighborhood, which are associated with gradients lower than the calculated threshold.

2.1.3.3 Omnidirectional low pass filter

This interpolation is performed on flat regions. A simple 5x5 omnidirectional low pass filter is applied for smoothing the image, removing some residual noise from it. In this case the original sampled values are also modified. The kernels used to achieve the interpolation are shown in equation (2.39). In particular, the *A* kernel is applied in case of G central pixel, whereas the *B* kernel is used in case of R/B central pixel.

$$A = \frac{\begin{bmatrix} 0 & 8 & 4 & 8 & 0 \\ 8 & 8 & 16 & 8 & 8 \\ 4 & 16 & 16 & 16 & 4 \\ 8 & 8 & 16 & 8 & 8 \\ 0 & 8 & 4 & 8 & 0 \end{bmatrix}}{64} \qquad B = \frac{\begin{bmatrix} 0 & 3 & 9 & 3 & 0 \\ 3 & 16 & 10 & 16 & 3 \\ 9 & 10 & 28 & 10 & 9 \\ 3 & 16 & 10 & 16 & 3 \\ 0 & 3 & 9 & 3 & 0 \end{bmatrix}}{64} \tag{2.39}$$

2.2 False Colors Removal

Since the color interpolation step quite well exploits spatial correlation, zipper effect does not often arises. On the contrary, residual false colors can be introduced by the directional filters. For this reason, a postprocessing technique has been developed, which eliminates the residual false colors, thus considerably improving the final image quality.

Many methods have been developed in the past to reduce false colors. Among them, the most interesting techniques median filter the inter-channel differences, thus removing spikes and valley from them, which usually correspond to false colors. Freeman's approach [52] and Lu and Tan's technique [92] have been already presented in the introduction section. The post-processing approach described in [95] not only is based on the color difference model, but also uses an edge-sensing mechanism. Another interesting technique, which is proposed in [84], updates the R, G, B values adaptively, modifying also the original pixel value which could be corrupted, due to the effect of noise. Two updated values for the green channel are calculated using each color difference domain:

$$\begin{array}{l} G^R(i,j) = R(i,j) + v_{GR}(i,j) \\ G^B(i,j) = B(i,j) + v_{GB}(i,j) \end{array} \tag{2.40}$$

where

$$\begin{array}{l} v_{GR}(i,j) = median\{G(k,l) - R(k,l)\,|(k,l) \in A\} \\ v_{GB}(i,j) = median\{G(k,l) - B(k,l)\,|(k,l) \in A\} \end{array} \tag{2.41}$$

And A denotes the support of the 5x5 local window centered in (i,j).

The updated G value is determined by the weighted sum of two updated G^R and G^B values of each color difference domain and original G value. Subsequently, R and B values are updated using the updated G value. This process is expressed as:

$$\begin{array}{l} G'(i,j) = \frac{1}{2}G(i,j) + \frac{1}{2}\{(1 - a(i,j))\,G^R(i,j) + a(i,j)\,G^B(i,j)\} \\ R'(i,j) = \frac{1}{2}R(i,j) + \frac{1}{2}\{G'(i,j) - v_{GR}(i,j)\} \\ B'(i,j) = \frac{1}{2}B(i,j) + \frac{1}{2}\{G'(i,j) - v_{GB}(i,j)\} \end{array} \tag{2.42}$$

where $a(i,j)$ is a weight, expressed as:

$$a(i,j) = \frac{\sigma^2_{(G-R)}(i,j)}{\sigma^2_{(G-R)}(i,j) + \sigma^2_{(G-B)}(i,j)}, 0 < a(i,j) < 1 \tag{2.43}$$

$\sigma^2_{(G-R)}$ and $\sigma^2_{(G-B)}$ represent the variances of interchannel differences.

As it is apparent from the equation (2.42), the color correction algorithm proposed in [84], thanks to the variance information, weights more the flatter color difference domain than the other. Moreover, the initial interpolated value is not totally exchanged by the updated value, but it is equally weighted for correction. Subsequently, the color values of the central pixel are replaced by R', G' and B' so that they will be involved in filtering the updating pixels.

The local statistics are effectively estimated from a running square window, through the formulas in (2.44):

$$\begin{array}{l} E\left[A\left(i,j\right)\right]=\frac{\sum_{k,l\in A}e(k,l)\cdot A(k,l)}{\sum_{k,l\in A}e(k,l)} \\ \sigma_A^2\left(i,j\right)=\frac{\sum_{k,l\in A}e(k,l)\cdot\left(A(k,l)-E[A(i,j)]\right)^2}{\sum_{k,l\in A}e(k,l)} \\ e\left(k,l\right)=1-\left(A\left(i,j\right)-A\left(k,l\right)\right) \end{array} \tag{2.44}$$

The just described technique has the disadvantage of weighting the unfiltered values together with the filtered ones, so false colors are reduced, without being completely removed. If the unfiltered values were not weighted in the color correction process, false colors would be removed much better but true colors could be removed as well. In fact, it is well known [148] that a window width $2k+1$ median filter can only preserve details lasting more than $k+1$ points. To preserve smaller details in signals, a smaller window width median filter must be used. Unfortunately, the smaller the filter window is, the poorer its false color reduction capability becomes.

To overcome this problem, we have developed a new solution which takes into account both a 3x3 window median filter and a 5x5 window median filter. In particular, local variances of interchannel differences are evaluated through the equation (2.44) in both 3x3 and 5x5 square windows, thus obtaining $\sigma^2_{(G-R)3\times3}$,$\sigma^2_{(G-B)3\times3}$,$\sigma^2_{(G-R)5\times5}$,$\sigma^2_{(G-B)5\times5}$. Similarly, four median values (v_{GR3x3},v_{GB3x3} ,v_{GR5x5} ,v_{GB5x5}) are computed.

Due to the Bayer arrangement and to a quite good interpolation achieved by the previous demosaicing step, the green plane is less affected by false colors than the R and B planes, and hence it is left unmodified. The color correction on R

and B planes is performed according to the following rules:

$$\begin{aligned} R'(i,j) &= G(i,j) - \left[(1 - kr_{ratio}(i,j)) \cdot v_{GR3\times3}(i,j) + kr_{ratio}(i,j) \cdot v_{GR5\times5}(i,j)\right] \\ B'(i,j) &= G(i,j) - \left[(1 - kb_{ratio}(i,j)) \cdot v_{GB3\times3}(i,j) + kb_{ratio}(i,j) \cdot v_{GB5\times5}(i,j)\right] \end{aligned} \tag{2.45}$$

Where

$$\begin{aligned} kr_{ratio}(i,j) &= \frac{\sigma^2_{(G-R)3\times3}(i,j)}{\sigma^2_{(G-R)3\times3}(i,j)+\sigma^2_{(G-R)5\times5}(i,j)} \\ kb_{ratio}(i,j) &= \frac{\sigma^2_{(G-B)3\times3}(i,j)}{\sigma^2_{(G-B)3\times3}(i,j)+\sigma^2_{(G-B)5\times5}(i,j)} \end{aligned} \tag{2.46}$$

By this way, the greatest weight is assigned to the median value associated to the neighborhood having the lowest variance. So, if the pixel being processed belongs to an edge, the 3x3 variances of interchannel differences are likely to be greater than the correspondent 5x5 variances, and hence the filtering action will be stronger. Vice versa, if the 3x3 neighborhood is flatter than the 5x5 neighborhood, the 3x3 median is weighted more than the 5x5 median, and hence details lasting 2 pixels at least are preserved.

Fig.2.6 shows a schematic representation of the proposed approach.

Since variances could be used as measures of the flatness of a region, it is possible to perform the color correction process only where it is needed, thus reducing the power consumption. In the color difference domain, a flat color difference neighborhood is characterized by: expectation values close to the central pixel values, low variance values. Defined $Diff_{GR}(i,j) = G(i,j) - R(i,j)$ and $Diff_{GB}(i,j) = G(i,j) - B(i,j)$, these two conditions can be expressed with the following formulas:

$$\begin{gathered} \left|E\left[Diff_{GR}(i,j)\right] - Diff_{GR}(i,j)\right| < Mean_{Threshold} \\ \left|E\left[Diff_{GB}(i,j)\right] - Diff_{GB}(i,j)\right| < Mean_{Threshold} \\ \sigma^2_{(G-R)3\times3}(i,j) < Variance_{Threshold} \\ \sigma^2_{(G-B)3\times3}(i,j) < Variance_{Threshold} \end{gathered} \tag{2.47}$$

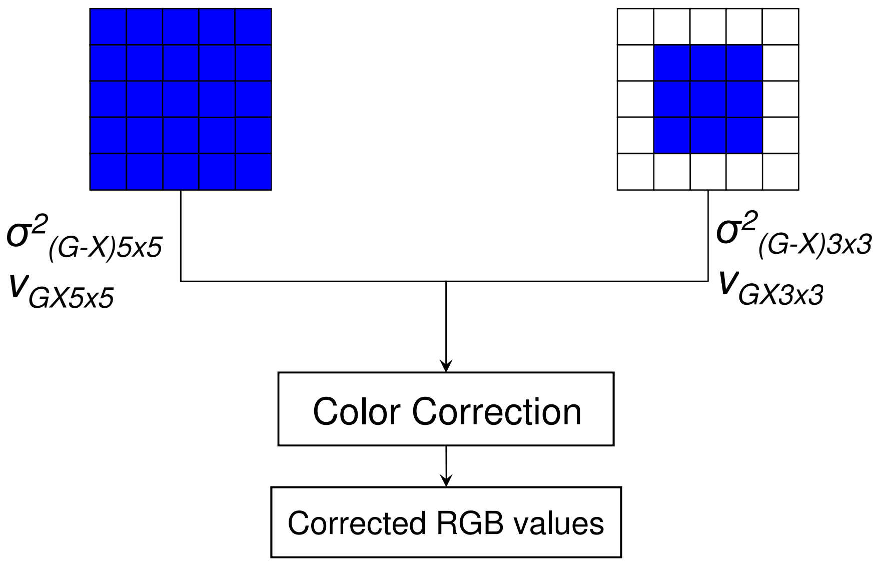


Figure 2.6 : Schematic representation of the proposed approach

where *MeanThreshold* and *VarianceThreshold* are two thresholds, which can be set to 16 for 8 bits images.

If the conditions stated above are satisfied, the region is considered flat, thus the central pixel can be left unchanged.

According to the expectation value and to the variance, a map of homogeneous (black) vs. inhomogeneous (white) regions can be achieved, as it is apparent from Fig.2.7.

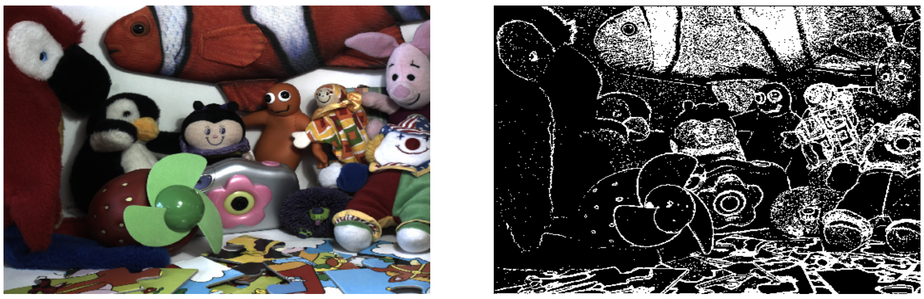

(a) Processed image (b) Map of homogeneous vs. inhomogeneous regions

Figure 2.7 : Map of homogeneous vs. inhomogeneous regions

Homogeneous regions (black) can be left unchanged or can be low pass filtered. Inhomogeneous regions (white) are processed by the color correction algorithm.

2.3 Experimental Results

In this section, we will show the experimental results obtained using the proposed algorithm compared with other demosaicing techniques. The results are highligthed both visually and numerically. For coherence with the most of papers on demosaicing, which use the Kodak image test set to make comparisons with other techniques, we downloaded this database from [51] and then we processed these images through the proposed approach. These images contains a lot of edges and textures, thus they are useful for highlight how the various algorithms handle the high frequency content. Since the Kodak image test set is full color, CFA images are simulated by subsampling the color channels according to the Bayer Pattern. This means that the algorithms are not applied on genuine Bayer data (derived from a real sensor), but on images that have already had matrix and gamma applied, so their gamut is far wider than that of data coming from a sensor. Moreover they are immune from noise and other impairments related to the sensor technology. For these reasons we also applied the proposed algorithm to the kodak images, to which we applied a gaussian noise with $\sigma = 8, \sigma = 12, \sigma = 25$ In our experiments, we have used the PSNR (Peak signal to Noise Ratio) for evaluating the quality of the images (only for Kodak database). Given two $N \times M$ images A and B, the PSNR is expressed as:

$$PSNR = 20 \times \log_{10}\left(\frac{255}{\sqrt{\frac{1}{N\times M}\sum_{i=1}^{N\times M}(A_i - B_i)^2}}\right) \tag{2.48}$$

Higher values (expressed in decibel) of the PSNR generally imply better quality.

For demonstrating the competitiveness of the proposed algorithm, we compared it with other thirteen demosaicing algorithms, whose PSNR are reported in Table 2.2. The Table reports, for readability reason, only part of the measures available in the Table I of the survey [88]. For precision the techniques used are :

1. Lu&Tan's method (LT) [92];
2. Alternating projection (AP) [65];
3. Adaptive homogeneity-directed (AHD) [74];
4. Successive approximation (SA) with edge-weighted improvement [138];
5. Lukac's CCA method [97] with post-processing [96];

6. Frequency-domain (FD) demosaicing [10];
7. Variance of color-difference (VCD) [36];
8. Directional Linear Minimum Mean Square-Error Estimation (DLMMSE) [153];
9. Local polynomial approximation (LPA) [115];
10. Adaptive filtering (AF) for demosaicing. [89];
11. Gunturk's method [65];
12. The directional interpolations proposed by Hirakawa et al. in [75];
13. Menon et al. in [103].

The PSNRs have been computed separately for each color channel. However, as it is well known, the PSNR is not always representative of the visual quality of an image. Two images having the same PSNR could appear very different from the visual standpoint, because the artifacts on edge or in flat zones are differently perceived by the human visual system, due to the contrast masking effect. Moreover, the PSNR approach is a full-reference metric which cannot be applied on real sensor data. For these reasons, a subjective image quality analysis has been also achieved. Figures 2.9 and 2.10 illustrate the ROIs (region of interest) relative to the "hats" test image (kodim03) and to the "mountain" test image (kodim13), respectively. They were obtained cropping a detail from the originals and interpolated images. In these figures, (a) represents the ROI of the original image, (b) is the result of the bilinear interpolation, (c) is the result of the edge sensing algorithm, (d) is obtained applying the interpolation by Gunturk, (e) is the result of the Hirakawa's approach, (f) is the output of the technique proposed by Menon *et al.* and, finally, (g) is obtained using the proposed algorithm. The

ROIs show how our algorithm reduces both the zipper and false color defects. Fig. 2.9.b presents a great amount of false colors and blur. Edge sensing algorithm quite well interpolates along edges but introduces many false colors (see fig. 2.9.c). Figs. 2.9.d and 2.9.f show that both Gunturk's and Menon's approach still maintain few zipper artifacts near edges, although for this image the PSNR of the Menon's technique is comparable to ours and higher than the Hirakawa's approach, which outputs an image free from zipper effects and false colors (see fig. 2.9.e) like our demosaicing technique. Looking at fig. 2.10, it is possible to see that the proposed technique (fig. 2.10.g) outperforms all the others in removing false colors, providing an image where also edges are effectively interpolated and sharpness is maintained. Both bilinear (fig. 2.10.b) and edge sensing (2.10.c) techniques produce images heavily affected by false colors. Gunturk's (fig. 2.10.d) and Menon's (2.10.f) approaches also introduce some visible false colors. Finally, Hirakawa's technique still maintains a minimum amount of false colors in comparison with our approach.

Another test which is usually performed to compare different demosaicing algorithm is the interpolation of a colored resolution chart, which allows to see how each algorithm resolves color details. For this purpose, the image shown in fig. 2.11.a was taken into consideration. Looking at fig. 2.11.d it is quite evident that the proposed technique resolves color details much better than the other two approaches (see figs. 2.11.b and 2.11.c). This is due to the proposed direction estimation block which evaluates the correlation among different Bayer colors, to determine the direction to be used in the interpolation phase.

As it has been already discussed, images coming from real sensors are often affected by artifacts due to noise and green imbalance, so it is important to design an interpolation algorithm able to not exalt and even to reduce these kinds of issues, which would be further enhanced by the successive sharpening step. As the proposed technique low pass filters flat regions, green imbalance and noise are heavily reduced while preserving the details on edges and textures, thus producing high quality images. Looking at fig. 2.12.c it is possible to notice that residual noise is quite well reduced by the proposed technique, whereas edges are still well interpolated. Both figs. 2.12.a and 2.12.b are still affected by noise. In order to have a numeric measure of the noise immunity of the presented demosaicing technique, we added three different amounts of Gaussian noise (having standard deviations equal to 8, 12 an 25, respectively) to the original Kodak images, and then interpolated them with some state of the art color interpolation approaches and the proposed technique. Afterwards, we calculated the PSNR with respect to the original kodak images. The PSNR values are reported in Table 2.1, where the proposed approach is compared with the algorithms proposed in [103], [87] and [145]. From this table it is evident how the proposed demosaicing algorithm has the greatest PSNR values for almost all the processed images. Moreover, the improvement is greater especially in images having large flat areas, where the human visual system is more sensitive to noise. The noise immunity of the algorithm depends on the threshold used to identify flat regions. In the aforementioned experiment it was set to 300 . Fig. 2.8 is a map showing which interpolation is chosen for each pixel of the 'lighthouse' image, where gaussian noise having $\sigma = 12$ was added. In particular, black regions are interpolated through the 5x5 omnidirectional low pass filter, red regions are interpolated by the directional approach and green regions are interpolated through the simple 3x3 technique. It is quite evident that strong edges are interpolated by the directional technique, whereas regions near edges and textures are reconstructed by the 3x3 approach. Noise in flat regions is reduced by the omnidirectional

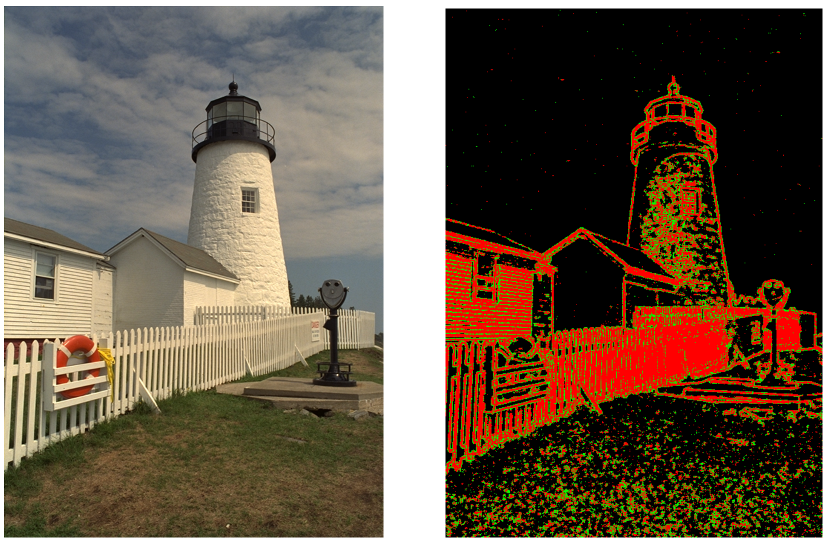

(a) Original image (b) Map of interpolation

Figure 2.8 : Map of interpolation method

low pass filter. A ROI of the 'lightouse' image is shown in fig. 2.13 where the original kodak image is compared with the noisy image (with Gaussian noise, $\sigma = 12$) interpolated by the Menon's technique [103], the Li's approach [87], the Wu and Zhang's algorithm [145] and the proposed technique.

Table 2.1 : PSNR(dB) performance comparison of different demosaicing methods in noisy conditions, where s8, s12 and s25 means $\sigma = 8$, $\sigma = 12$ and $\sigma = 25$ rispectively.

Image		Kodim05	Kodim15	Kodim19	Kodim23	**Average**
Menon	s8	29.37	29.90	29.68	29.91	29.72
	s12	26.33	26.77	26.40	26.54	26.51
	s25	20.53	21.07	20.33	20.43	20.59
Li	s8	29.47	30.08	29.92	30.03	29.88
	s12	26.38	27.00	26.66	26.75	26.70
	s25	20.61	21.16	20.48	20.54	20.70
Wu Zhang	s8	29.74	30.40	30.04	30.44	30.16
	s12	26.80	27.31	26.85	27.10	27.02
	s25	21.07	21.61	20.83	20.98	21.12
Proposed	s8	29.42	32.12	30.82	34.00	31.59
	s12	27.32	29.90	28.53	30.80	29.14
	s25	21.43	22.40	21.34	21.54	21.68

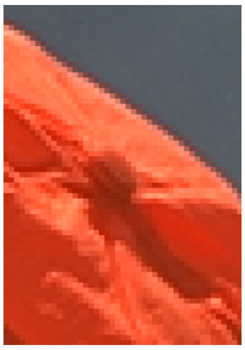
Finally, one more visual comparison is presented to validate the effectiveness of the proposed post-processing false colors removal algorithm with respect to other state of the art approaches. In particular, the images from the Kodak data set were interpolated through the demosaicing step proposed in Section II, then they were processed by the Freeman's technique [52], the Lu's algorithm [92], the post-processing approach proposed by Lukac *et al.* in [95], the color correction technique presented in [84] and the proposed algorithm, already disclosed in Section III. Fig. 2.14 is a ROI from the "hotel" test image (kodim08). In particular, (a) represents the ROI of the original image, (b) is the result of the color interpolation disclosed in Section II, (c) is obtained processing (b) with the Freeman's approach, (d) is the result of the Lu's technique, (e) is the output of the Lukac's post-processing technique, (f) is the result of [84] and, finally, (g) is obtained processing (b) with the algorithm presented in Section III. It is quite evident that the proposed color interpolation technique (fig. 2.14.b) very well exploits spatial correlation, but some false colors are introduced, and hence we apply a post-processing aliasing removal. Freeman's technique (fig. 2.14.c) not only does not remove many false colors but also introduces annoying zipper artifacts. Lu's approach does not introduce zipper effects, but considerably blurs the image. Lukac's algorithm is neither able to remove all the color artifacts. Kim's approach, weighting the initially interpolated values to produce the corrected ones, is not able to effectively reduce the false colors. The proposed technique (fig. 2.14.g) quite well reduces false colors without introducing zipper effects. The complexity in terms of operations is, obviously, dependent on the image content. To figure out, somehow, the complexity of the approach, each single algorithm phases will be analyzed. The direction estimation for each pixel needs 32 sums, 18 differences, 18 multiplications and 56 comparisons. This phase is mainly composed by MAC operations. The Variance analysis needs 20 differences and 2 divisions. The Activity analysis needs 20 sums, 25 diffs, 1 division and 4 comparisons. The 5x5 filtering needs 25 MAC operations.

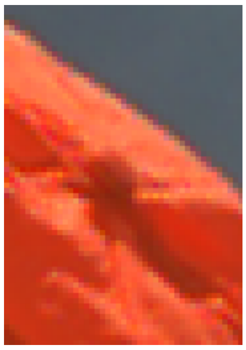
2.4 Conclusion

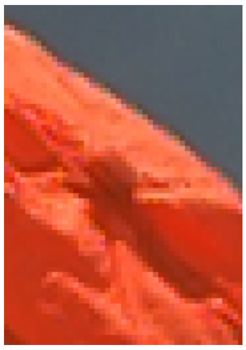
We have presented a color interpolation method based on edge and texture analysis. According to this analysis we proposed the usage of different interpolation approaches, where the novel proposed directional filtering exploits spatial-spectral correlation. The method allows to perform effective reconstructions also in case of noisy images. A post-processing algorithm effectively removes residual impairments introduced by the color interpolation step. This paper compares demosaicing performance of our improved method with the state-of-the-art demosaicing methods. The cascading structure of the proposed method has a merit of the pipelined real-time processing for the hardware implementation, while some other methods require iterative computations.

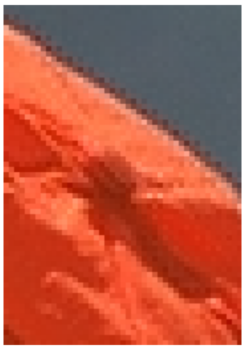
Table 2.2 : PSNR(dB) performance comparison of different demosaicing methods on Kodak PhotoCD image set selected from the first eight images used in Table I of Li. et al [88].

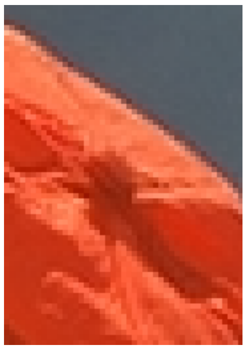
Image		1	2	3	4	5	6	7	8	**Average**
	R	42.20	37.69	41.29	34.27	40.98	38.42	40.25	38.14	39.16
Gunturk	G	43.48	41.29	43.34	38.08	44.10	41.03	44.47	42.62	42.30
	B	38.64	34.00	36.93	32.59	38.74	36.83	35.44	38.44	36.45
	R	38.96	37.58	40.06	33.21	40.31	36.63	41.36	37.61	38.22
Hirakawa	G	43.27	39.11	42.66	35.35	43.01	39.01	42.86	39.50	40.60
	B	39.57	36.45	39.01	33.24	40.64	37.46	40.55	37.86	38.10
	R	39.97	39.29	41.39	34.66	41.25	38.33	42.89	38.78	39.57
Menon	G	45.08	41.06	44.09	37.33	44.39	40.93	44.59	41.23	42.34
	B	39.92	37.91	40.22	34.76	42.94	39.02	41.98	39.98	39.59
	R	42.90	38.00	43.33	34.88	42.85	38.81	41.01	40.11	40.24
LT	G	46.24	39.82	45.88	37.15	44.90	40.88	42.88	41.95	42.46
	B	43.06	37.59	42.44	34.98	41.90	39.35	40.55	39.46	39.92
	R	42.07	39.06	42.53	35.20	42.50	39.08	42.18	40.02	40.33
AP	G	45.33	42.75	44.91	39.66	45.30	42.78	45.75	43.86	43.79
	B	42.69	38.97	41.51	35.58	42.31	40.02	41.70	39.95	40.34
	R	41.42	38.58	41.13	34.17	41.76	37.64	42.37	39.16	39.53
AHD	G	45.16	40.08	43.67	36.14	44.06	39.89	43.77	40.77	41.69
	B	42.23	38.05	40.33	34.35	41.08	38.41	41.57	38.62	39.33
	R	41.90	40.21	42.53	35.96	42.93	39.18	43.40	40.98	40.89
SA	G	46.32	43.48	44.52	40.37	46.12	43.29	46.42	44.33	44.36
	B	42.96	39.54	40.48	36.78	42.33	40.53	42.24	40.33	40.65
	R	41.14	39.58	41.74	35.65	42.76	39.03	42.45	40.66	40.38
CCA	G	45.56	43.03	45.21	39.77	45.98	43.32	45.72	43.96	44.07
	B	42.13	38.96	40.87	36.22	41.94	40.46	41.62	39.95	40.27
	R	38.86	37.40	39.35	32.28	39.93	36.88	40.10	37.12	37.74
FD	G	44.16	42.30	43.49	37.58	43.60	42.74	44.84	42.05	42.60
	B	41.41	38.23	41.03	32.95	41.11	39.23	40.62	37.90	39.06
	R	42.97	40.93	42.92	36.57	43.70	39.73	44.47	41.10	41.55
VCD	G	46.74	43.83	45.58	40.25	46.71	43.33	47.03	44.09	44.70
	B	43.50	40.37	41.85	37.10	43.10	40.82	43.55	40.60	41.36
	R	42.90	41.39	42.95	36.33	43.71	39.98	44.75	41.69	41.71
DLMMSE	G	47.56	43.69	46.26	39.63	46.68	43.19	46.82	44.14	44.75
	B	43.86	40.44	41.80	36.66	42.93	40.96	43.54	40.88	41.38
	R	43.86	42.10	43.77	37.42	44.26	40.44	44.91	42.18	42.37
LPA	G	47.75	44.81	46.53	41.15	47.01	43.85	47.15	44.72	45.37
	B	44.46	41.07	42.65	37.85	43.42	41.39	43.77	41.46	42.01
	R	42.93	38.74	43.48	35.51	43.16	39.38	41.61	40.58	40.67
AF	G	46.98	41.67	46.66	38.88	46.29	42.63	44.53	43.49	43.89
	B	43.65	38.18	42.41	35.63	42.69	39.80	40.99	39.92	40.41
	R	41.16	37.16	40.64	33.50	40.87	37.05	40.29	37.86	38.57
Proposed	G	44.79	40.92	43.77	37.93	43.97	41.43	43.95	41.69	42.31
	B	41.76	36.73	40.19	33.67	40.39	38.38	39.91	38.39	38.68

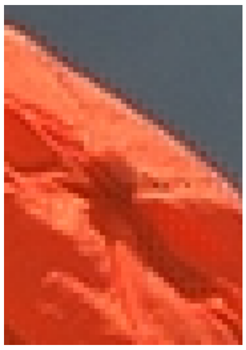

(a)


(b)


(c)


(d)


(e)


(f)

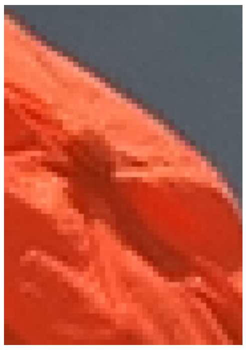

(g)

Figure 2.9 : Example of ROI visual comparison relative to the ”hats” test image (kodim03). (a) Original image; (b) Bilinear Interpolation; (c) Edge sensing [35]; (d) Gunturk [65]; (e) Hirakawa [75]; (f) Menon [103]; (g) Proposed solution.

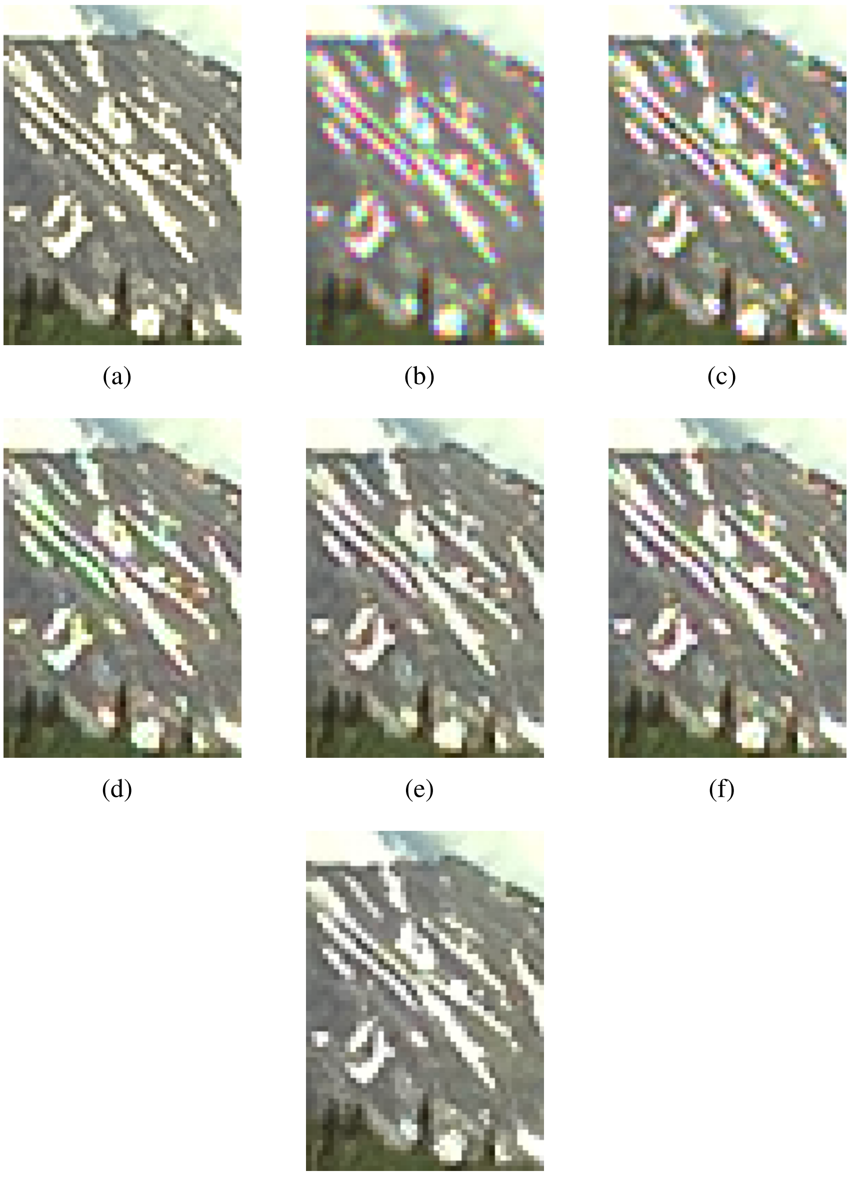

(a) (b) (c) (d) (e) (f) (g)

Figure 2.10 : Example of ROI visual comparison relative to the ”mountains” test image (kodim13). (a) Original image; (b) Bilinear Interpolation; (c) Edge sensing [35]; (d) Gunturk [65]; (e) Hirakawa [75]; (f) Menon [103]; (g) Proposed solution.

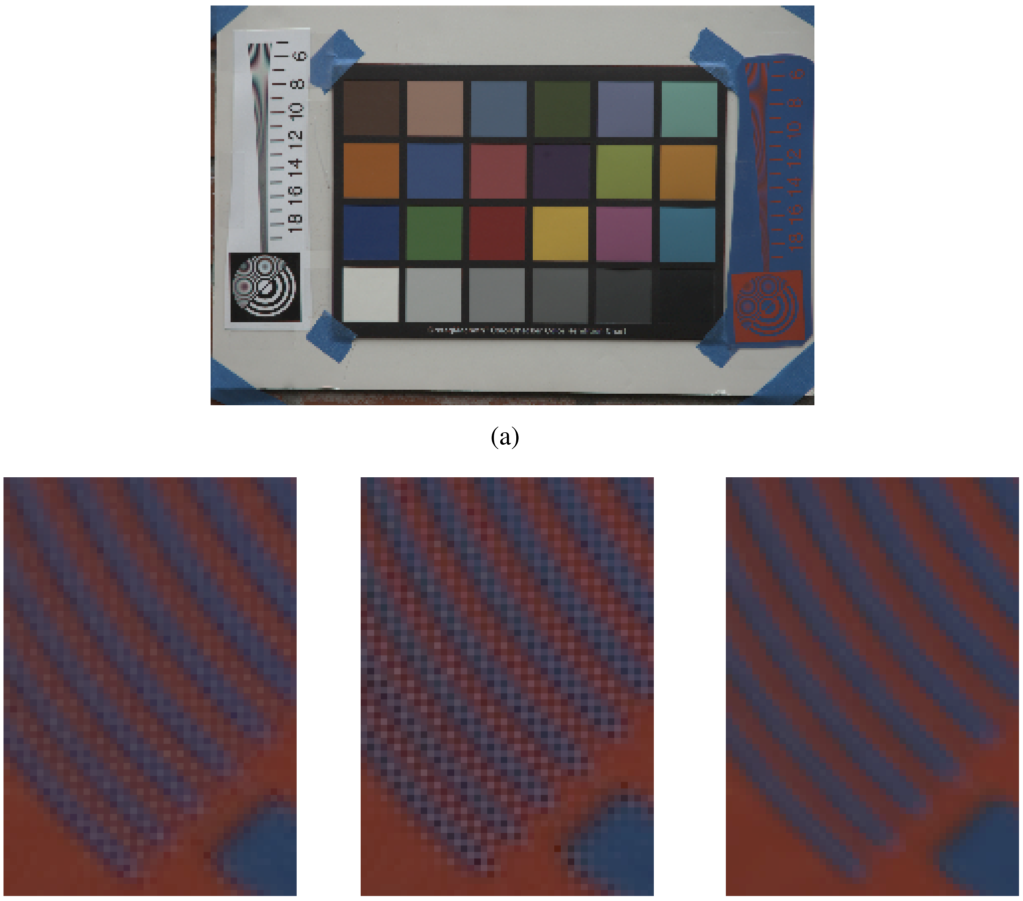


Figure 2.11 : Example of visual comparison on colored resolution chart.(a) Real sensor image; (b) ROI by Hirakawa [75]; (c) ROI by Menon [103]; (d) ROI by proposed solution.

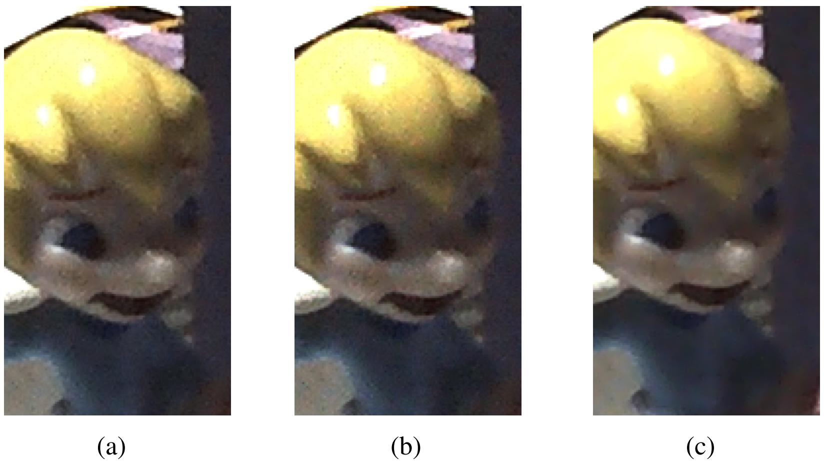


Figure 2.12 : Example of visual comparison on noise reduction effects.(a) ROI by Hirakawa [75]; (b) ROI by Menon [103]; (c) ROI by proposed solution.

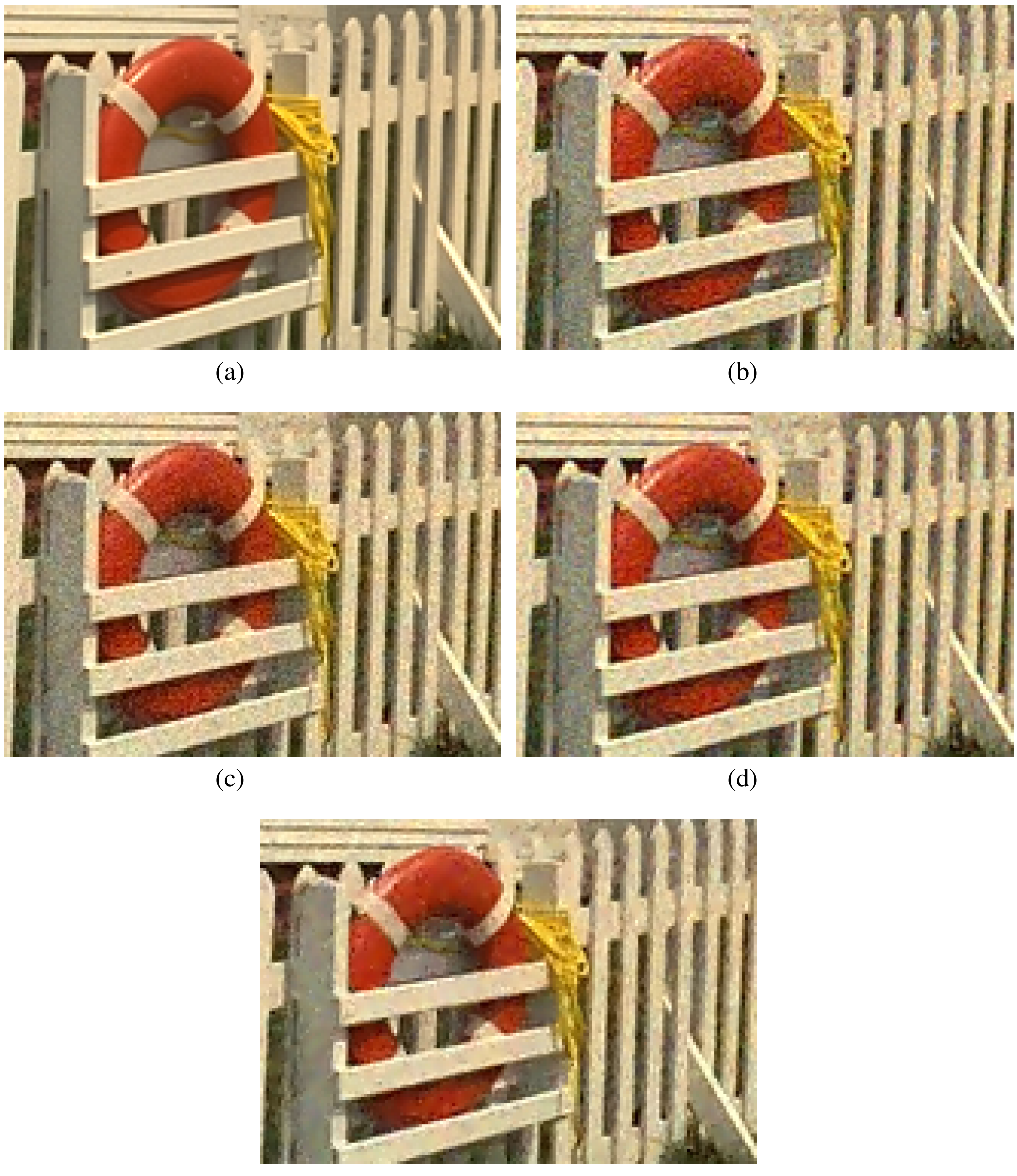
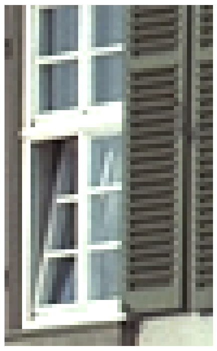
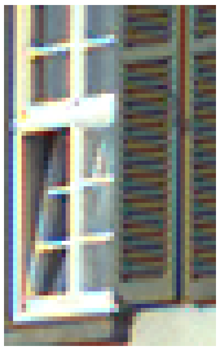
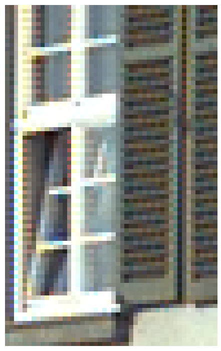
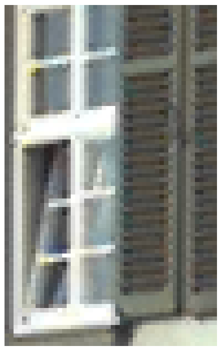


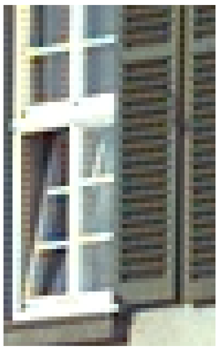
Figure 2.13 : Example of ROI visual comparison relative to the ”lighthouse” test image (kodim19) corrupted by gaussian noise with sigma=12. (a) Original image; (b) Menon [103]; (c) Li [87]; (d) Wu and Zhang [145]; (e) Proposed solution.

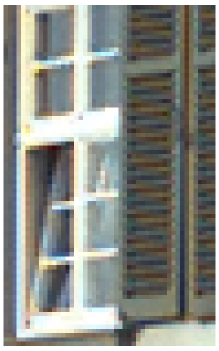

(a)


(b)


(c)


(d)


(e)


(f)

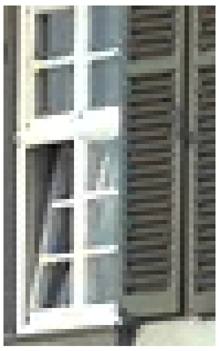

(g)

Figure 2.14 : Example of ROI visual comparison relative to the ”hotel” test image (kodim08). (a) Original image; (b) Interpolation through proposed approach (Sec.II); (c) image (b) filtered by Freeman [52]; (d) image (b) filtered by Lu [92]; (e) image (b) filtered by Lukac [95]; (f) image (b) filtered by Kim [84]; (g) Proposed solution.

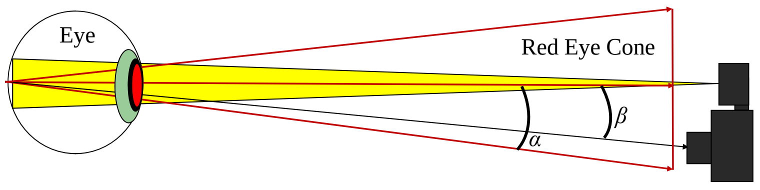
Part II

Image Analysis and Enhancement

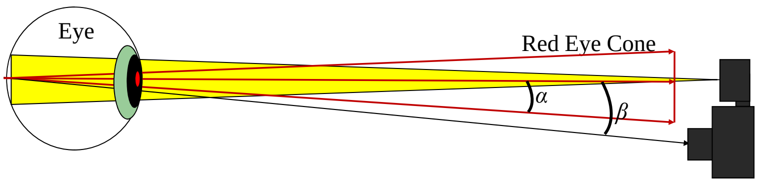
3. Red Eyes Removal

3.1 Introduction

Red eye artifacts are a well-known problem in digital photography . They are caused by direct reflection of light from the blood vessels of the retina through the pupil to the camera objective. When taking flash-lighted pictures of people, light reflected from the retina forms a cone, whose angle α depends on the opening of the pupil. Be β the angle between the flash-gun and the camera lens (centered on the retina), the red eye artifact is formed if the red light cone hits the lens, that is, if α is greater than or equal to β (see Fig.(**3.1**)). Small compact devices and point-and-click usage, typical of non-professional photography, greatly increase the likelihood for red eyes to appear in acquired images.



(a)



(b)

Figure 3.1 : The red eye is caused by the reflection of the flash off the blood vessels of the retina. The camera will record this red hue if the angle β is not greater than α **(a)**, otherwise the red eye is not recorded **(b)**.

High-end cameras often feature a separate flash with an extensible and steerable bracket, which allows for more distance between the flash and the lens, thus reducing the probability for red eyes to appear. One preventive measure suitable to both high-end and low-end devices is to make additional flashes before actually taking the photograph (pre-flash). This method, first proposed by Kodak [108], gives time to pupils to shrink in order to reduce the reflectance surface, thus making red eyes less likely. It is important that enough time elapses between flashes to account for the response time of the pupils (see Fig.(**3.2**)). This approach is effective, but it has the disadvantage of greatly increasing power consumption, which may be problematic for power-constrained mobile devices. Also, the additional flashing may sometimes be uncomfortable to people.

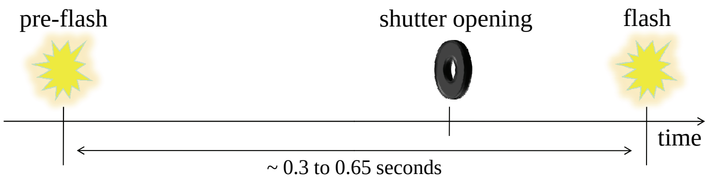


Figure 3.2 : Timeline explaining the pre-flash approach. Before the actual acquisition, a flash is fired. After a short time, the shutter opens and light enters the sensor. At the end of the exposure time the “true” flash is fired. Time between flashes is such that the pupils have time to react and shrink.

Red eye prevention methods reduce the probability of the phenomenon but don’t remove it entirely. Most of the times, then, the picture must be corrected during post-processing. Red eye removal is a very challenging task: red eyes may vary in shape and color, and may also differ in position and size relative to the whole eye. Sometimes light is reflected on a part of the retina not covered with blood vessels, yielding a yellow or white reflection (golden eyes). Some examples of the phenomenon are showed in Fig.(**3.3**). Designing a system which can effectively address all the possible cases is very difficult.

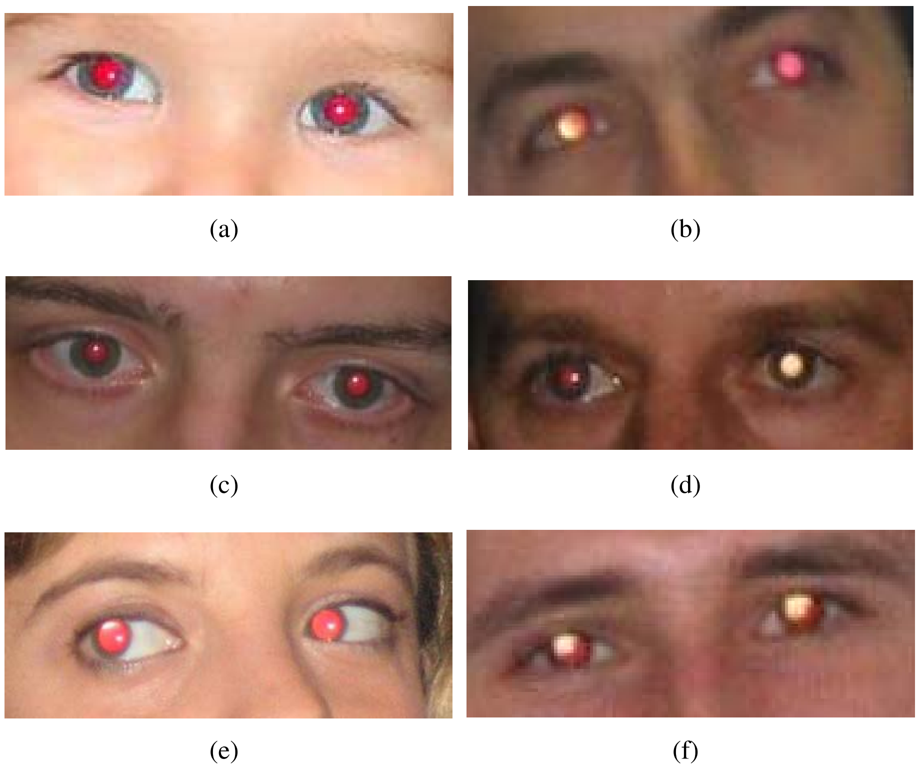


Figure 3.3 : Examples of the variability of the red eye phenomenon. Golden eyes are also visible.

For red eyes to be removed, they must be first reliably detected then properly corrected. Detection methods are divided into semi-automatic methods, which ask the user to manually localize and point the red eyes, and automatic methods, which detect the red eyes themselves. In the first case the eyes are manually selected using a visual interface (e.g., Adobe Photoshop [7], Corel Paint Shop Pro [39], ACDSee [4], etc.). This is feasible because eyes are easy to localize for men, but requiring manual intervention for every picture taken is unsuitable for non-professional usage; moreover, it may be difficult to have such an interface on a mobile device.

Automatic methods attempt to find red eyes on their own. Since they don't require user intervention, they are easier to use and more appealing, thus suitable for embedded devices. However, automatic detection of red eyes is a very challenging task, due to the variability of the phenomenon and the general difficulty in reliably discerning the shape of eyes from other details.

Red eye correction techniques, on the other hand, may be more or less invasive. Generally speaking, "easier" cases may be addressed with a softer correction, while sometimes a stronger intervention is needed. Since the aim is to provide a corrected image which looks as natural as possible, a less invasive correction is preferred when the natural aspect of the eye is reconstructible from the acquired image.

This Chapter aims to provide an overview of well-known automatic red eye detection and correction techniques, pointing out working principles, strengths and weaknesses of the various solutions. For further information about red eye removal, see recent surveys by Gasparini *et al.* on academic papers [56], on patents [57] and some of our personal works [17–19].

This Chapter is organized as follows. Section 3.2.1 explores red eye detection. Section 3.2.2 describes methods for red eye correction, while section 3.2.3 gives an insight into the problem of unwanted and improper corrections, showing their side effects. Lastly, Section 3.2.4 provides criteria to evaluate the quality of the results.

3.2 Prior Art

3.2.1 Eye Detection

The main difficulty in detection of red eyes is their great degree of variability. In the easier cases, the pupil has a normal shape and size and differs from a regular one only by its color. However, it is not uncommon for the red reflection to spread over the iris generating an unnatural luminance distribution. Usually a small white glint is also present, representing the direct reflection of the flash on the surface of the eye and giving the eye much more naturalness.

Typical red eye detection approaches involve extraction of red zones combined with skin extraction, shape template matching, and/or face detection. Some approaches also make use of classifiers to further refine their results.

3.2.1.1 Color Based

Color based approaches are the simplest ones. They are based on detecting red zones which may correspond to red eye artifacts. As a typical constraint for the position of the red eyes, they also detect the human skin, then consider some criteria about the relative position of the red eyes and the skin (usually, the eyes must be almost completely surrounded by nearby skin). Some color based approaches also detect the sclera (the white part of the eye), distinguishing it from the skin. Possible constraints may be imposed about the geometry of the red zones, such as discarding candidates too much elongated to represent a red pupil. This kind of approaches is quite simple, but does not take into account more complex features like, e.g., the presence of the various parts of the eye or the detection of the face.

One of the biggest problems of color-based techniques is characterizing exactly the colors to look for. Usually, interesting portions of the color space (corresponding to red, skin color, etc.) are delimited by hard thresholds, but they may also delimited by soft margins, yielding a fuzzy probability for the color to belong to the region. However, finding proper boundaries for the regions is a challenging task. The color of red eyes is heavily influenced by the type of flash used, the sensor and the processing pipeline. While this is not a big issue, since the thresholds may be fine-tuned to adapt to the acquisition system, there are external factors which may influence the color of the eyes, including (but not limited to) the age of the person, the opening of the pupils, the distance from the camera, and the angle between the eyes and the flash. The variability is so high that even the same subject in one picture may have two different colored red eye artifacts, or a red eye and a regular one (see Fig.(**3.4**)). Moreover, if the flash is not very strong (as is often the case with mobile devices), the external illuminant may produce a noticeable color cast on the picture, which adds another degree of variability to the colors. Similar considerations apply to the color of the skin and of the sclera.

The red color region may be defined in different color spaces. In the RGB space,

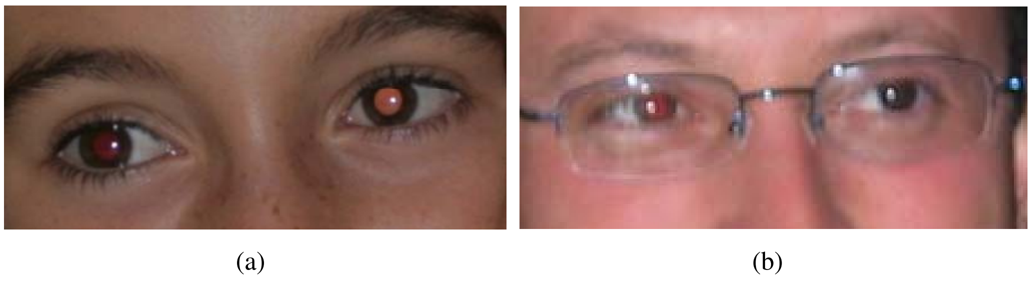


Figure 3.4 : Picture **(a)** shows two very different red eyes; picture **(b)** shows one red eye along with a regular one.

a possible definition is [152]:

$$\begin{cases} R > 50 \\ R/(R+G+B) > 0.40 \\ G/(R+G+B) < 0.31 \\ B/(R+G+B) < 0.36 \end{cases} \tag{3.1}$$

Often, instead of hard thresholds, a *Redness* function is provided. This function is an estimate of how well the color of each pixel resembles a red eye artifact, and is used as a way to define soft margins for the red color region. Some possible redness functions [55, 58, 72, 133] are:

$$Redness = (R - \min\{G, B\}) \tag{3.2}$$

$$Redness = \frac{R^2}{(G^2 + B^2 + 14)} \tag{3.3}$$

$$Redness = \frac{\max\{0, (R - \max\{G, B\})\}^2}{R} \tag{3.4}$$

$$Redness = \max\left\{0, \frac{2R - (G+B)}{R}\right\}^2 \tag{3.5}$$

As an alternative to select an interesting portion of the color space, it is possible to compare a redness function with a luminance function, discarding pixels whose luminance is more noticeable than the redness [144]:

$$Redness = R - (G + B)/2 \tag{3.6}$$

$$Luminance = 0.25R + 0.6G + 0.15B \tag{3.7}$$

$$RedLum = \max\{0, 2 \cdot Redness - Luminance\} \tag{3.8}$$

Search for red regions may also be performed in color spaces different from RGB, such as YCC [117] or HSL [26].

Given a particular choice for the red color region, it is possible to convert each image to a representation which shows whether each pixel belongs to the region. Such representations are called redness maps. According to the employed definition for the red color region (hard-thresholded or soft-delimited), the redness map is a black-and-white or full-grayscale image (in the latter case, the redness function is adjusted to the possible maximums and minimums of the redness function, or to the maximums and minimums over each particular image). Fig.(**3.5**) and Fig.(**3.6**) show redness maps computed using the above formulas.

Skin extraction may be performed in a similar way as red color extraction.

Other color based information useful to detect red eyes may be gained searching for the sclera [143] and selecting the zones where the flash has noticeably

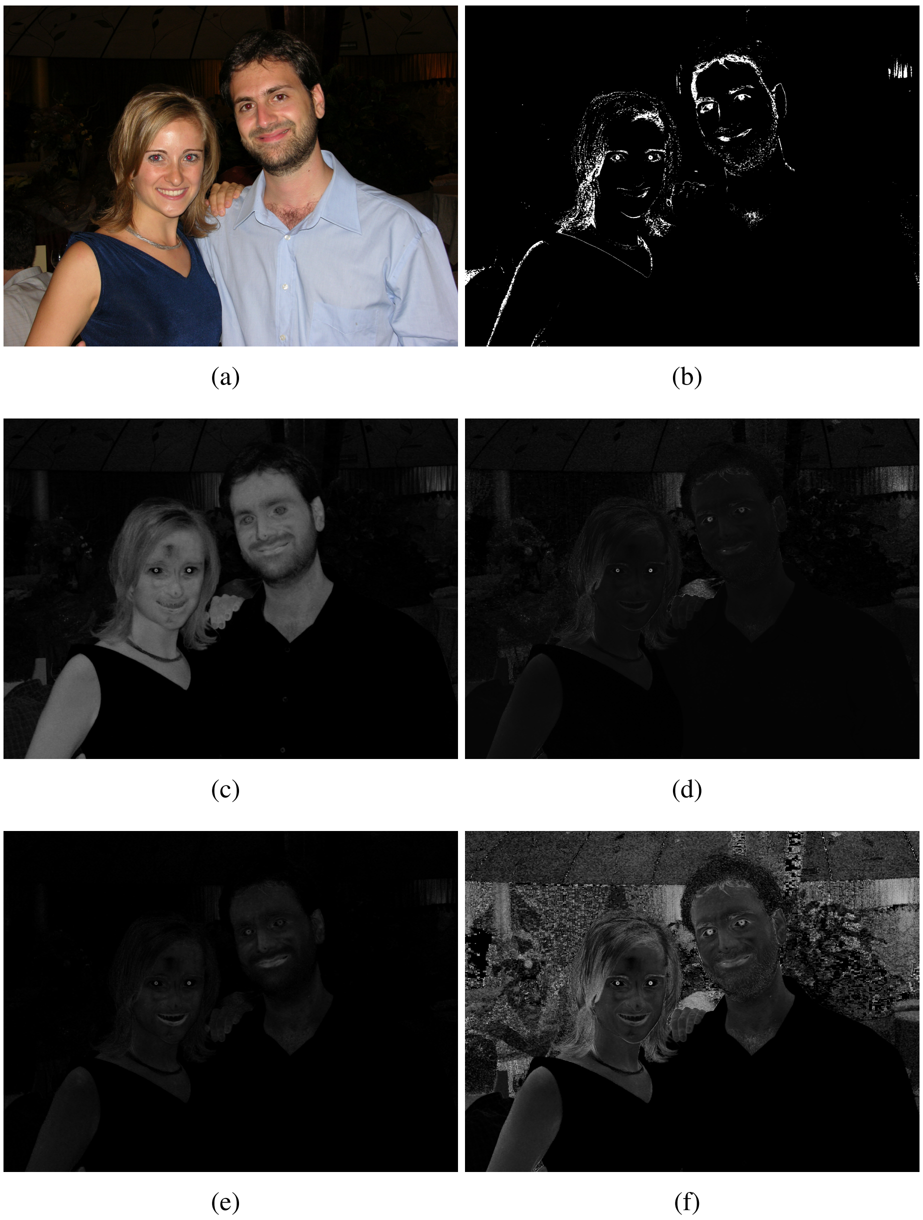
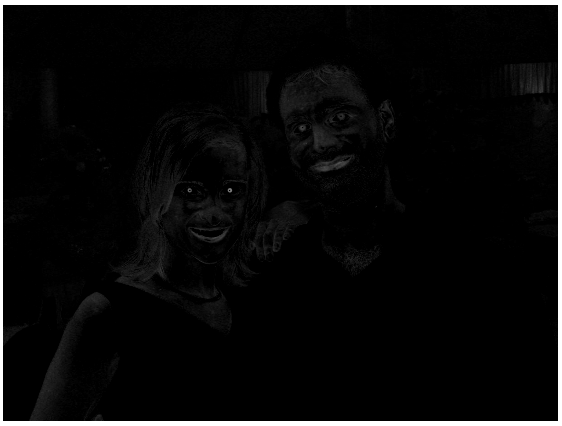
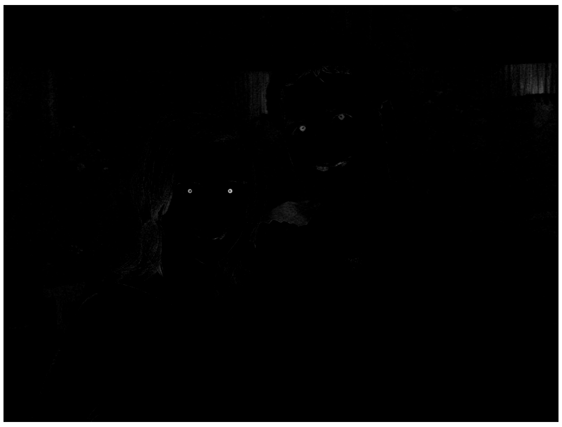


Figure 3.5 : Examples of redness maps. **(a)** Original image; **(b-f)** redness maps obtained from (3.1), (3.2), (3.3), (3.4), (3.5), respectively.



(a)



(b)

Figure 3.6 : (a) Redness map obtained from (3.6); **(b)** redness vs. luminance map computed according to (3.8).

affected the image (discarding, e.g., a distant background) [50]. Using thresholding and morphological operators to combine different masks, it is possible to effectively extract red pupils.

3.2.1.2 Shape Based

Shape based approaches attempt to find eyes exploiting simple information about their shape. They typically use templates which are matched at different positions and resolutions, in order to search the image for shapes which may correspond to eye features. The region of interest is then restricted to zones where the response of the templates is stronger. Using simple circular or square templates it is possible to recognize, e.g., the difference in intensity between the inner pupil and the outer skin and sclera. Slightly more complex templates may be useful in locating the other parts of the eye, which helps to effectively assess the presence or the absence of a red eye [98].

Edge detection filters may also be useful to extract information about shape. It is possible to use them in conjunction with color tables to make advantage of both spatial and chromatic information [128].

3.2.1.3 Pairing Verification

One of the most obvious constraints which can be used to filter out false detections is eye pairing verification [131]. It is based on the assumption that every eye found must be paired with the other one on the same subject's face. The two eyes must have the same size, and they must be in a certain range of distances (possibly proportional to the size, in order to account for the distance of the subject from the camera) from each other, in a horizontal or almost horizontal direction. If an eye can't be paired because it has no suitable match, it is discarded, since it is most probably a false detection.

This approach is effective, since it is very unlikely for two false positives to satisfy the pairing criteria, but it presents a major drawback: if a face is partially occluded, so that only one eye is visible in the picture, and that eye is red, it will not be corrected, since it can't be matched to the other one. The same problem will occur when both eyes are visible but only one is red, or when both are red but only one is detected, possibly due to a difference in color (see Fig.(**3.7**)).

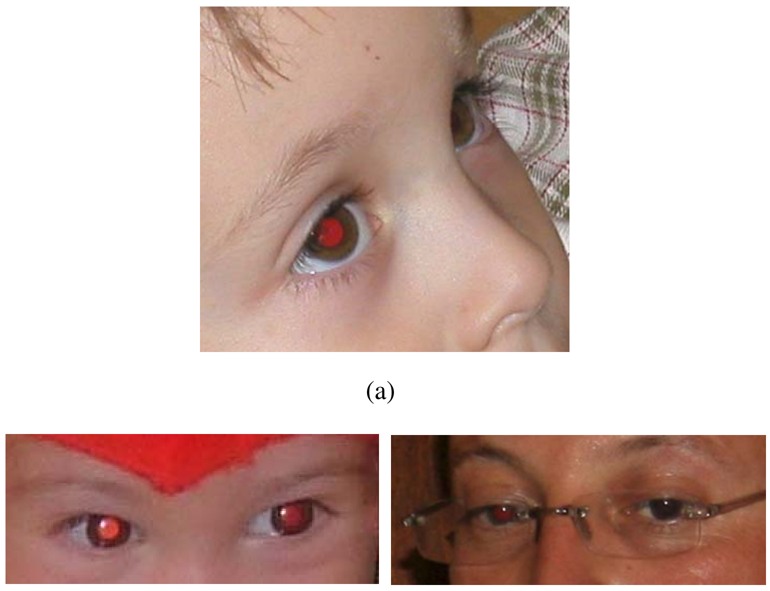


Figure 3.7 : In picture **(a)**, only one of the eyes is visible; the red eyes in picture **(b)** are very different, and in most cases only one of them will be properly detected; in picture **(c)** only one of the eyes is affected by the red eye phenomenon. In all these cases, the pairing verification will fail.

3.2.1.4 Face Detection

The most sophisticated kind of approach to red eye detection is based on face detection [58]. Restricting the search region to the zones where faces are detected, it is possible to discard a great number of false positives.

In details, a face detection system determines the locations and sizes of human faces in arbitrary digital images by making use in some cases of ad-hoc facial features. The localization is done by considering a bounding box that encloses the region of interest. The detection problem if often achieved as a binary pattern classification task; the content of a given part of an image is transformed into features used to train a classifier on example faces able to decide whether that particular region of the image is a face, or not. For practical situations is very common to employ a sliding-window technique just using the classifier on small portions of an image (usually squared or rectangular), at all locations and scales, as either faces or non-faces. In the more general case the face localization is achieved regardless of position, scale, in-plane rotation and orientation, pose (out-of-plane rotation) and illumination. Further source of problems are the presence or absence of structural components (e.g., beards, mustaches and glasses), the facial expression that has a great impact over the face appearance and the occlusions that occur when faces may be partially occluded by other objects.

To implement a robust face detector it is fundamental to fix some points relative to the specific application. In particular it is important to decide the facial representation, the involved pre-processing, the particular ”cues” (e.g., colors, shape, etc.) and the classifier design.

In literature a lot of approaches have been published with different capabilities, advantages and limitations. Of course, the implementation of a face detector system inside an embedded device system requires ad-hoc peculiarities due the limited available resources. The constrained domain imposes to consider methods

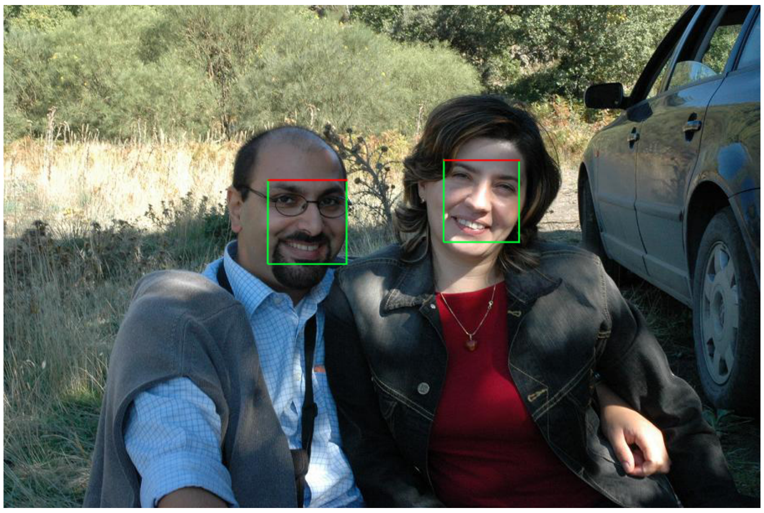


Figure 3.8 : Face Detection Example.

able to guarantee a reasonable trade-off between robustness and computational issues. For this reason it is out of the scope of this Section to provide a detailed review of all related technologies. See [79, 147] for more specific details.

One of the most popular algorithms in the field is due to Viola and Jones [142]. For this reason we have decided to describe it in a more details just to give also some useful suggestions for a practical implementation. The authors introduced for the first time the concept of ”integral image”, a way to compute efficiently local features in an incremental way just to find in a suitable way the underlying scale of the face to be localized. The integral image $Int(x,y)$ of a given image I at location (x,y) is defined as follows:

$$Int(x,y) = \sum_{(i \leq x,\ j \leq y)} I(i,j) \tag{3.9}$$

It is computed using just a single pass over the image I as described in [142]. By proper managing such image it is possible to compute any processing over rectangular patches in a very efficient manner. The corresponding rectangular

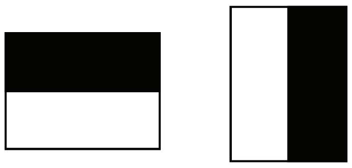


Figure 3.9 : Some examples of rectangular features managed by the algorithm of Viola and Jones. Each feature can be computed in a very efficient way just using the concept of integral image.

features can then be computed as simple differences between adjacent rectangular sums. Although the similarity with the Haar features [91] are pretty evident, the proposed strategy is able to obtain effective results in a more efficient way.

The remaining important contribution of the seminal work of Viola and Jones was the introduction of an ad-hoc classifier making use of a learning approach based on Adaboost [53]. This classifier is able to discriminate, among a large set of potential features, a smaller number of elements without lacking too much of accuracy. The system is able to select, among others, a small number of features just considering a boosting approach [130] that, given an exhaustive set of positive and negative examples, with a greedy algorithm, decides the best set of features to be considered both in terms of robustness and fast detection rate (see Fig.(**3.9**) an example of rectangular features.).

Finally, they described a way to combine efficiently, in a cascade approach, the output of different classifiers just to speed-up the overall process. With respect to the former approaches this method was the first able to work with sufficient accuracy in real time application.

Of course, the underlying ingredients of a face detector can be improved in several ways just providing to the overall flow further information to be processed.

In [78] is proposed a learning based face detector able to find human faces in a very fast way. To further speed up the process, a face rejection cascade is constructed to remove most of negative samples while retaining all the face samples. To do that a series of skin color features [123] are used as useful prior.

The face localization can be thought as the pre-processing step needed to recognize the person which the face belong to. Person identification is, of course, a more difficult task especially if implemented in a so constrained domain such as imaging devices. Also in this field there is a huge number of papers published over the past decades, specifically devoted to biometric world. By the way, personal photos have an associated context, often already available to the user of the photo management system. In newer systems the combination of user feedback with EXIF meta information (and, if available, with GPS location) can give an effective improvement to the unsupervised recognition [112].

First commercial products are available also to be used in novel application context such as social network [44, 61], etc.

The quality of the detection greatly depends on the quality of the face detector. Sometimes it is limited to frontal upright faces, while red eye artifacts may be located in profile or three-quarter views of subjects (especially when taking snapshots). Therefore, face detectors with such limitations are not suitable for red eye detection. Another important degree of variability is the age of the subject: children are difficult to detect, since their faces have a different shape and different features than those of adults. Nonetheless, they have a higher chance to present red eye artifacts, since their pupils are usually more open. Thus, it is important for face detection to be robust both to the angle of view and to the age of the subject.

Another important issue related to face detection is that it doesn't help discard false detections on the face, which are usually critical. An additional constraint which may be imposed is to only accept eyes located in the upper half of the detected face. This helps filter out some false detections (e.g., lips or tongue) but it keeps the ones near the eyes (e.g., details of glasses or pimples on the forehead).

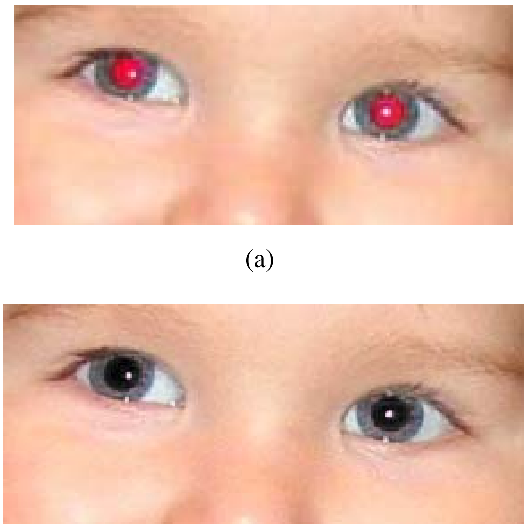
3.2.2 Red Eye Correction

The goal of red eye correction is to modify the image in such a way that it looks as natural as possible, given the assumption that there are red eye artifacts in the detected zones (according to the eye detector, the assumption may be given for sure or with a certain degree of probability).

According to the extent to which the artifact has corrupted the image, the correction algorithm may need to adjust the hue, brightness, luminance distribution, and/or even the shape and size of the pupil. Since naturalness of the image is the goal, it is best to use a minimally invasive technique to correct each case. This also means that a way to distinguish the gravity of each artifact (either in the detection phase or at the very beginning of the correction phase) is to be preferred, in order to be able adapt the correction method on a case-by-case basis [101].

3.2.2.1 De-saturation

In the simplest cases, the eye has its regular shape, and the artifact only consists in the wrong color of the pupil. In these cases, the optimal solution is equally simple: the red eye is desaturated, that is, its chrominance is (totally or partially) suppressed, while its luminance is left intact or only slightly lowered (see Fig.(**3.10**)).



(a)

(b)

Figure 3.10 : In the simplest cases, pupil desaturation produces good results.

One simple way of desaturating red pupils is to replace each pixel with a gray shade at 80% of original pixel luminance [117]. An adaptive desaturation may be performed in the CIELAB color space by stretching the lightness values of the pupil so that its darkest point becomes black [67]:

$$\begin{aligned} L^*_{corrected} &= \frac{\max L^*}{(\max L^* - \min L^*)}\left(L^* - \min L^*\right) \\ a^*_{corrected} &= 0 \\ b^*_{corrected} &= 0 \end{aligned} \tag{3.10}$$

Desaturation may suffer from a boundary effect: the transition between the corrected and uncorrected area may be noticeable and unpleasant. Moreover, some pixels outside the pupil may be incorrectly considered to be part of the red eye artifact and desaturated. To overcome these problems, a smoothing (usually Gaussian) mask may be used to modulate the strength of the correction. For each pixel (i, j) in the red eye artifact area, be $c_{original}(i, j)$ its color in the uncorrected image, $c_{target}(i, j)$ the target color of the correction and $m(i, j)$ the value of the smoothing mask; the final corrected color $c_{corrected}(i,j)$ is then:

$$c_{corrected}(i,j) = c_{target}(i,j) \cdot m(i,j) + c_{original}(i,j) \cdot (1 - m(i,j)) \qquad (3.11)$$

3.2.2.2 Inpainting

In the hardest cases, a more invasive correction is needed. Often, the distribution of reflected light is influenced by the direction of the flash with respect to the face. Sometimes eyes present the "washed out" effect, where the reflected light spreads off the pupil onto the iris. In these cases a simple desaturation may yield incorrect and unnatural results (see Fig.(**3.11**)).

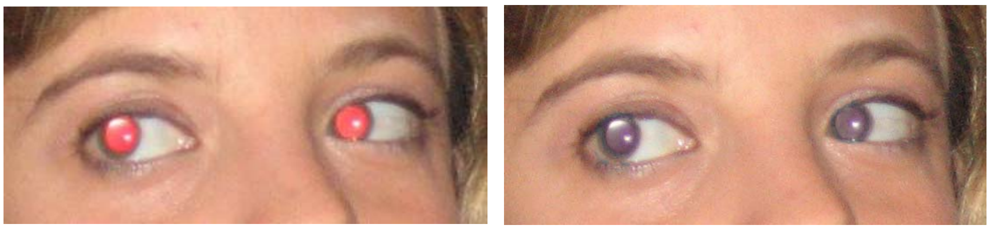
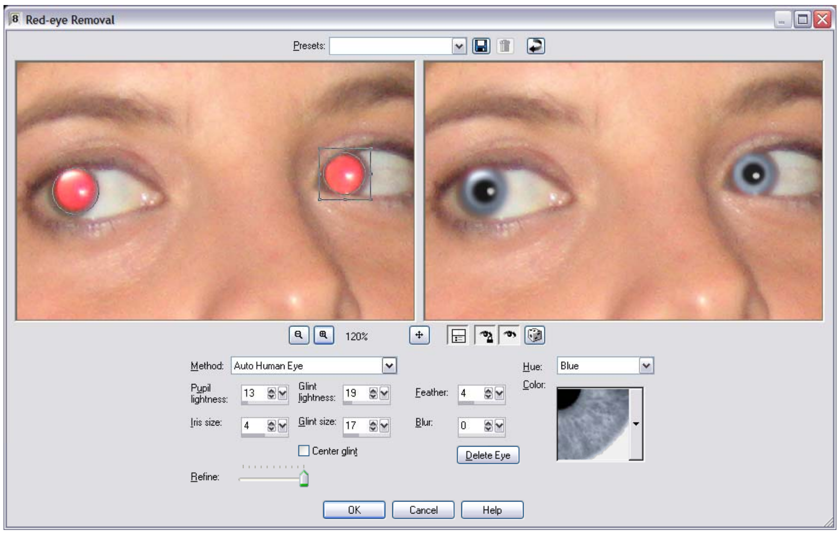

(a) (b)

Figure 3.11 : When reflected light spreads over the iris, simple desaturation gives unnatural results.

It is then necessary to use a more complex method to reconstruct a realistic image of the eye. Inpainting may vary from an adaptive recoloring of red pixels to a complete redrawing of iris and pupil [150]. The results, however, tend to be unrealistic, up to the point that they sometimes resemble glass eyes (see Fig.(**3.12**)[1]).

[1]Corel Paint Shop Pro Red-eye Removal tool.



Method: Auto Human Eye
Pupil lightness: 13
Glint lightness: 19
Feather: 4
Iris size: 4
Glint size: 17
Blur: 0
Center glint [ ]
Refine:
Hue: Blue
Color:

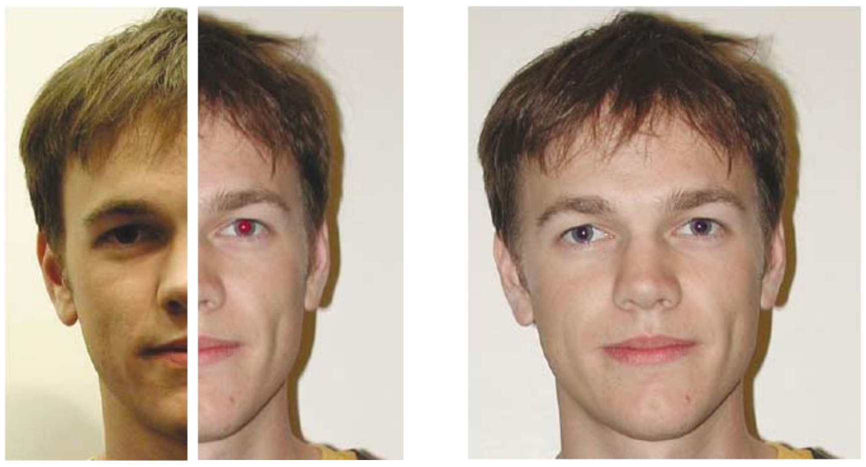
Figure 3.12 : Correction of washed-out red eyes with an inpainting technique.

3.2.2.3 Flash/no-Flash

Another way of obtaining simultaneous detection and correction of red eye artifacts is the "flash/no-Flash" technique [107], which aims to combine the advantages of taking a non-flashed picture and a flashed one. The main idea is to take a high-quality flashed picture and a low-quality non-flashed one, which is used to detect the red eyes and recover the natural colors of the affected zones (see Fig.(**3.13**)[2]).

The method works as follows: two pictures are taken in quick succession. The first one is shot without flash with high sensitivity, large lens aperture and with a short (for a non-flashed picture in low light conditions) exposure time. This yields a dark and noisy picture with small depth of focus, but still suitable to help recover the unaltered colors of the eyes. The second one is a regular flashed picture, which represents the "real" picture to correct. It is important that the two pictures are taken with the same focal length and that very little time elapses

[2]Picture taken from Petschnigg *et al.* [121].



(a) (b) (c)

Figure 3.13 : Flash/no-Flash technique. **(a)** Dark non-flashed picture used to recover the correct color of the eyes; **(b)** high-quality picture affected by red eye artifacts; **(c)** corrected picture.

in between, in order to prevent misalignment. Search for red eye artifacts is performed in a luminance-chrominance color space, usually CIELAB. The a^* channel is used as a measure of redness. Pixels whose a^* component exceeds a certain threshold are considered red. Among such pixels, those whose difference between the a^* channel in the flashed image and the same channel in the non-flashed image is larger than another threshold are marked as possible red eye pixels. Morphological operators are used to cluster them into blobs, discarding isolated pixels or very narrow regions as noisy results.

To correct red eyes using information from the non-flashed picture, it is important to first compensate differences in color cast between the two images. To this end, for each of the chroma channels a^* and b^*, the difference between the two images is averaged over all non-red eye pixels, thus obtaining a color compensation term. Correction of red eye artifacts is then performed by substituting the chrominance of affected pixels in the flashed image with the chrominance of the corresponding pixels in the non-flashed image, then adding the color compensation term.

The approach is quite simple and theoretically effective, but it presents a number of drawbacks. First of all, the memory and processing requirements double, since there are two pictures being taken in place of one. Moreover, the images may suffer from registration problems, or they may simply be misaligned due to movements of the hand or of the subjects. This makes this method especially unsuitable for snapshots, where people may be caught while moving. Another important issue of this approach is uneven illumination, which is recorded by the non-flashed image but not by the flashed one: a dark shadow on a red detail (such as the shadow of the nose projected on the lips) may trigger a false detection, which in turn causes image degradation (especially if the chrominance of the shaded part is not correctly perceived due to insufficient illumination).

3.2.3 Correction Side Effects

3.2.3.1 False Positive

One of the biggest issues in red eye removal is false positives in the detection phase. Correcting a red detail falsely detected as a red eye artifact may have a much more displeasing effect than leaving an artifact uncorrected. For this reason, getting as few false positives as possible is more important than catching as many red eyes as possible. Examples of image degradation resulting as correction of false positives are shown in Fig.(**3.14**).

False positives can be classified according to the severity of the associated degradation risk, as discussed in Section 3.2.4.

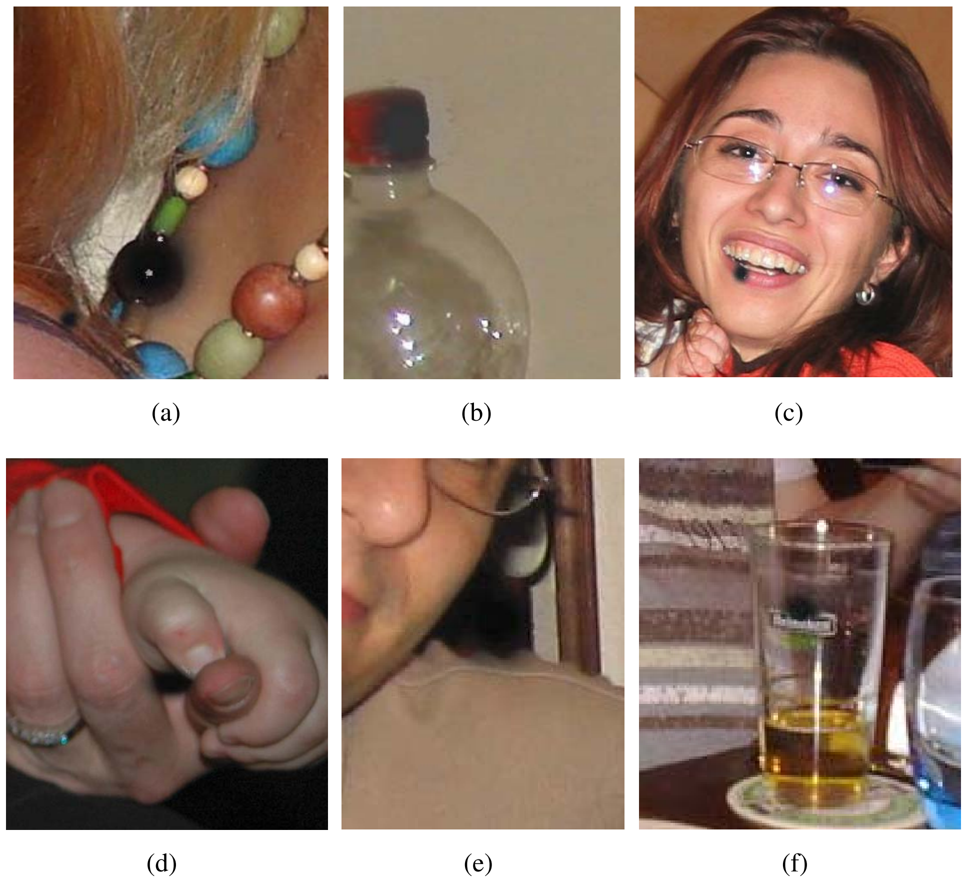


Figure 3.14 : Examples of corrections of false positives. Some are barely noticeable, while others are totally unacceptable.

3.2.3.2 Partial Detection/Correction

Sometimes eyes are properly detected, but wrongly corrected. In such cases unnatural corrections appear in the final image. Unnatural corrections, like false positives, are very undesirable, since they are often more evident and displeasing than untouched red eyes. One type of unnatural correction is partial correction, caused by an incorrect segmentation of the red eye zone (possibly due to a difference in hue or luminance between the detected and the undetected parts).

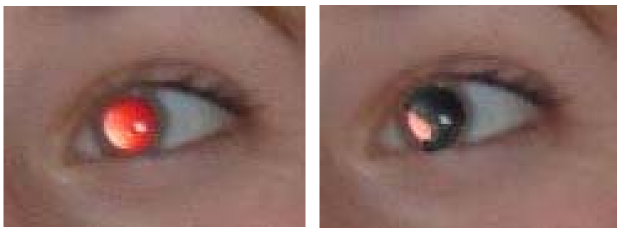


Figure 3.15 : Partial red eye correction, where the brighter area was not considered to belong to the red pupil.

Noisy Correction

Noisy correction is another kind of unnatural correction. Noisy corrections appear when, in presence of heavy image noise, red pixels are present around the pupil. In this case, the detector may assume that such pixels belong to the red eye, and correction may spread over the iris, giving a strange and unnatural look to the corrected eye.

It is worthwhile to note that a strong lossy image compression (e.g., low-bitrate JPEG) may cause the same phenomenon: however, in the context of automatic algorithms which act just after the picture is taken, it is reasonable to assume

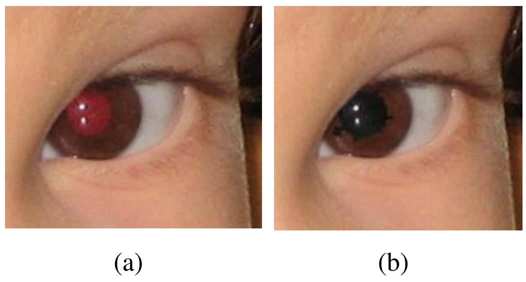


Figure 3.16 : Correction of the red pupil extends over the iris, due to red pixels caused by image noise.

that red eye removal is performed before image compression (to improve red eye detection and to avoid compressing twice).

Dead Eye

Sometimes red eyes are properly detected, but the corrected image just doesn't look natural. This may happen when a wrong luminance distribution, caused by reflected light, is kept through the correction and is evident in the resulting image. This may also happen when the color of the corrected pupils isn't quite natural, possibly because the correction isn't strong enough. Finally, the absence of the glint, which may be due to inpainting or excessive correction, may cause the eye to look "dead".

3.2.4 Quality Criteria

The formulation of a quality metric allows to choose the best solution and to adjust parameters of the algorithm in the best way. To achieve a quality control on red eye removal algorithm is a challenging issue. Usually the quality of the algorithm is estimated considering the ratio between corrected eyes and false

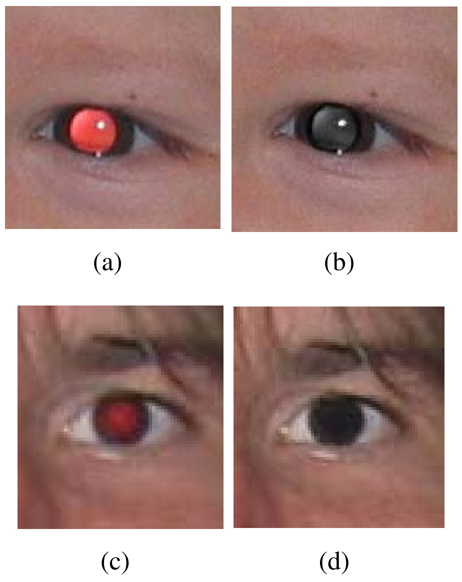


Figure 3.17 : In some cases, an unnatural luminance distribution is visible in the corrected image **(b)**. Sometimes, instead, the eye has a "dead" look, due to the absence of the glint **(d)**.

positives. Obviously this is strictly related to the nature of the database and the quantity of images. Safonov [128] introduced an interesting quality metric that permit to remove correlation between quantity and quality.

First of all the author enumerated all possible cases, further he prioritized them using Analytic Hierarchy Process (AHP) [127]. Obviously a representative set of photos affected by red eye defect should be used for calculation of these unwanted cases. Furthermore good solutions must have low False Negatives (FN) and False Positives (FP), ideally FN and FP are equal to zero. However the severity of the False Positives differs significantly. Almost indistinguishable small FP on foreground is undesirable but sometimes allowable. Visible FP on foreground especially on human faces and bodies is absolutely not allowable; such FP artifacts damage photo more than red eyes. Therefore he divided False Positives in two classes: FP_c is the number of critical FP and FP_n is the number non-critical

FP.

A similar situation is described for the False Negatives. Several red eye regions are relatively large and well distinguishable; other regions are small and have low local contrast. Detection of the first red eyes is defined as mandatory by Safonov, whereas detection of the second regions is desirable. Accordingly to such hypotheses he divided all FN in two groups: FN_m is defined as the number of regions which are mandatory for detection; FN_d is the number of regions which are desirable for detection.

One more unwanted situation is the correction of only one eye from pair. For semi-automatic approaches it is not so crucial because users have possibility to correct the second eye manually, but for embedded implementations it is quite unpleasant. N_P is then defined as the number of faces with one corrected eye from pair of red eyes.

The retouching quality is important too. Regarding correction Safonov distinguished two cases: if the corrected eye looks worse of the original red eye, for example only part of the red region is corrected, it is an irritating case; it is noticeable that eye has been corrected but it does not irritate strongly. Accordingly C_I is the number of irritating cases and C_n is the number of situations when retouching is noticeable.

As described above Safonov uses prioritization of the factors through AHP table (see 3.1) according to observers opinions. The simplest way for filling the table is: if left item is more important than top then cell is assigned to 5; if severity of the two items are the same then cell is set to 1; if top item is more important than left then cell is set to 1/5. Taking into account weights from AHP table, and taking into consideration a global weight of 10, for all the features, Safonov

Table 3.1 : Analytic Hierarchy Process Table. Where the coefficients a_i, that refer to the assigned importance values in each row, are used to estimate the Geometric Mean. The weight is estimated in percentage from the sum of the Geometric Means (=9.60).

Req. Quality	FN_m	FN_d	FP_c	FP_n	N_p	C_i	C_n	$\sqrt[7]{\prod_{i=1}^{7} a_i}$	Weight %
FN_m	1.00	5.00	0.20	5.00	1.00	0.20	5.00	1.26	13.13%
FN_d	0.20	1.00	0.33	5.00	0.20	0.20	5.00	0.68	7.08%
FP_c	5.00	3.00	1.00	3.00	5.00	5.00	5.00	3.43	35.73%
FP_n	0.20	0.20	0.33	1.00	0.20	0.20	1.00	0.34	3.54%
N_p	1.00	5.00	0.20	5.00	1.00	1.00	5.00	1.58	16.46%
C_i	5.00	5.00	0.20	5.00	1.00	1.00	5.00	1.99	20.72%
C_n	0.20	0.20	0.20	1.00	0.20	0.20	1.00	0.32	3.33%

proposes the following quality criterion:

$$Q_c = \frac{N_t - 1.3 \cdot FN_m - 0.7 \cdot FN_d - 3.6 \cdot FP_c - 0.4 \cdot FP_n - 1.6 \cdot N_p - 2.1 \cdot C_i - 0.3 \cdot C_n}{N_t} \tag{3.12}$$

where N_t is total number of red eyes.

3.3 A New Red-Eyes Detection and Correction Technique

The proposed red-eyes removal pipeline uses three main steps to identify and remove red-eyes artifacts. First candidates red-eyes patches are extracted, then classified to distinguish between eyes and non-eyes patches. Finally, correction is performed on detected red-eyes. The details of the three steps involved in the proposed pipeline are detailed in the following subsections.

3.3.1 Red Patch Extraction

To extract the red-eyes candidates, we first built a color model from the training set to detect pixels belonging to possible red-eyes artifacts. We constructed red-eye-pixel and non-red-eye-pixel histogram models using a set of pixels of the training images. Specifically, for each image of the training set, the pixels belonging to red-eyes artifacts have been labeled as red-eye-pixels (*REP*), whereas the surrounding pixels within a windows of fixed size have been labeled as non-red-eye-pixels (*NREP*). The labeled pixels (in both RGB and HSV spaces) have been mapped in a three dimensional space $C_1 \times C_2 \times C_3$ obtained taking into account the first three principal components of the projection through principal component analysis [42]. By using the principal component analysis the original six-dimensional space of each pixel considered in both RGB and HSV color domains, is transformed into a reduced three-dimensional space maintaining as much of the variability in the data as possible. This is useful to reduce the computational complexity related to the space dimensionality. We used a 3D histogram with $64 \times 64 \times 64$ bins in the $C_1 \times C_2 \times C_3$ space. Since most of the sample pixels of the training set lies within three standard deviations of the mean, each component C_i has been uniformly quantized in 64 values taking into account the range $[-3\lambda_i, +3\lambda_i]$, where λ_i is standard deviation of the i^{th} principal component (i.e., the i^{th} eigenvalue). The probability that a given pixel belongs to the classes *REP* and *NREP* is computed as follows:

$$P(C_1,C_2,C_3|REP) = \frac{h_{REP}[C_1,C_2,C_3]}{T_{REP}} \tag{3.13}$$

$$P(C_1,C_2,C_3|NREP) = \frac{h_{NREP}[C_1,C_2,C_3]}{T_{NREP}} \tag{3.14}$$

where $h_{REP}[C_1,C_2,C_3]$ is the red-eye-pixels count contained in bin $C_1 \times C_2 \times C_3$ of the 3D histogram, $h_{NREP}[C_1,C_2,C_3]$ is the equivalent count for non-red-eye-pixels, T_{REP} and T_{NREP} are the total counts of red-eye-pixels and non-red-eye-pixels respectively. We derive a red-eye-pixel classifier through the standard likelihood ratio approach. A pixel is labeled red-eye-pixel if

$$P(C_1,C_2,C_3|REP) > \alpha P(C_1,C_2,C_3|NREP) \tag{3.15}$$

where α is a threshold which is adjusted to maximize correct detection and minimize false positives. Note that a pixel is assigned to *NREP* class when both probability are equal to zero.

Employing such filtering, a binary map with the red zones is derived. To remove isolated red pixels, a morphology operation of closing is applied to this map. In our approach we have used the following 3×3 structuring element:

$$m = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix} \tag{3.16}$$

Once the closing operation has been accomplished, a search of the connected components is achieved using a simple scanline approach. Each group of connected pixels is analyzed making use of simple geometric constraints. As in [143], the detected regions of connected pixels are classified as possible red-eye candidates if the geometrical constraints of *size* and *roundness* are satisfied. Specifically, a region of connected red pixels is classified as possible red-eye candidate if the following constraints are satisfied:

- the size S_i of the connected region i is within the range $[Min_s, Max_s]$, which defines the allowable size for eyes.
- the binary roundness constraint R_i, of the connected region i is verified:

$$R_i = \begin{cases} True & \rho_i \in [Min_\rho, Max_\rho];\ \eta_i \leq Max_\eta;\ \xi_i \gg 0 \\ False & otherwise \end{cases} \tag{3.17}$$

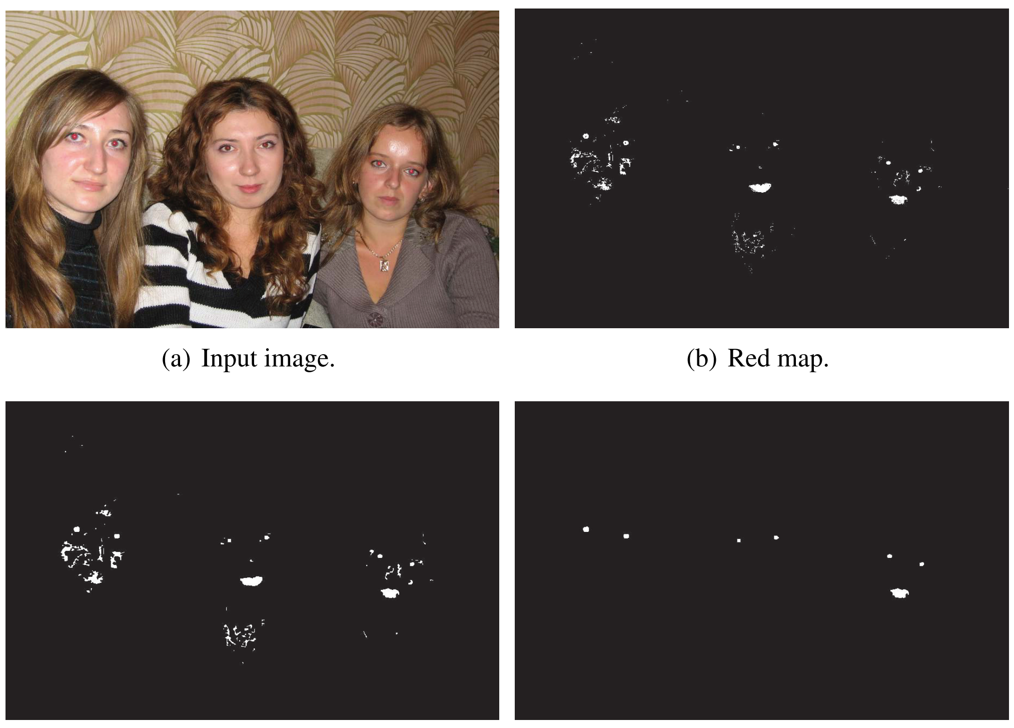
  where

$\rho_i = \frac{4\pi \times A_i}{P_i^2}$ is the ratio between the estimated area A_i and the perimeter P_i of the connected region; the more this value is near 1 the more the shape will be similar to a circle.

$\eta_i = \max\left(\frac{\Delta_{x_i}}{\Delta_{y_i}}, \frac{\Delta_{y_i}}{\Delta_{x_i}}\right)$ is the distortion of the connected region along the axes.

$\xi_i = \frac{A_i}{\Delta_{x_i}\Delta_{y_i}}$ is the filling factor, the more this parameter is near 1 the more the area is filled.

The parameters involved in the aforementioned filtering pipeline have been set through a learning procedure as discussed in Section 3.3.5.



(a) Input image. (b) Red map.

(c) Closing operation. (d) Final candidates.

Figure 3.18 : Filtering pipeline on a input image.

In Figure 3.18 all the involved steps in filtering pipeline are shown. The regions of connected pixels which satisfy the geometrical constraints are used to extract

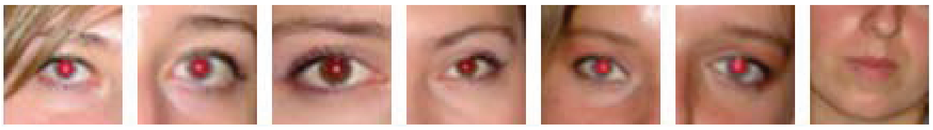


Figure 3.19 : Examples of possible candidates after red patches extraction.

the red-eyes patches candidates from the original input image (Figure 3.19). The derived patches are resembled to a fixed size (i.e., 30×30 pixels) and converted into gray code [60] for further classification purpose (Figure 3.20). Gray code representation allows to have a natural way (e.g., no strong transaction between adjacent values) to pick-up the underlying spatial structures of a typical eye.

The gray levels of an m-bit gray-scale image (i.e., a color channel in our case) is represented in the form of the base 2 polynomial

$$a_{m-1}2^{m-1} + a_{m-2}2^{m-2} + \cdots + a_1 2^1 + a_0 2^0 \tag{3.18}$$

Based on this property, a simple method of decomposing the image into a collection of binary images is to separate the m coefficients of the polynomial into m 1-bit bit planes. The m-bit Gray Code $(g_{m-1} \ldots g_2, g_1, g_0)$ related to the polynomial in Equation 3.18 can be computed as follows:

$$g_i = \begin{cases} a_i \bigoplus a_{i+1} & 0 \leq i \leq m-2 \\ a_{m-1} & i = m-1 \end{cases} \tag{3.19}$$

where $\bigoplus$ denotes the exclusive OR operation. This code has the unique property that successive code words differ only one bit position. Thus, small changes in gray level are less likely to affect all m bit planes.

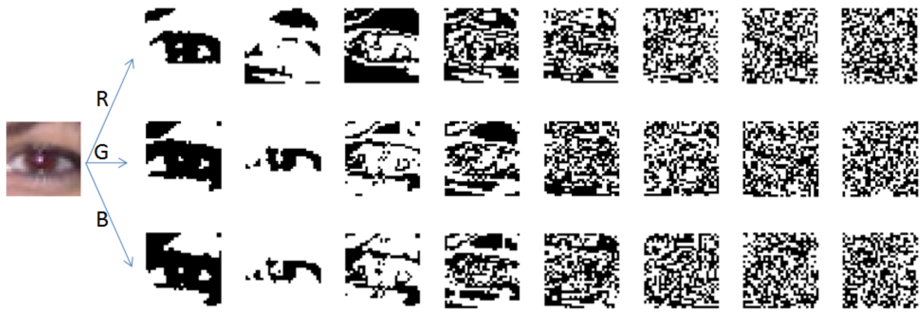


Figure 3.20 : Example of gray code planes on the three RGB channels of a red-eye patch.

3.3.2 Red Patch Classification

The main aim of the classification stage is the elimination of false positive red-eyes in the set of patches obtained performing the filtering pipeline described in Section 3.3.1.

At this stage we deal with a binary classification problem. Specifically, we want discriminate between eye vs. non-eye patches. To this aim we employ an automatic learning technique to make accurate predictions based on past observations. The approach we use can be summarized as follows: start by gathering as many examples as possible of both eyes and non-eyes patches. Next, feed these examples, together with labels indicating if they are eyes or not, to a machine-learning algorithm which will automatically produce a classification rule. Given a new unlabeled patch, such a rule attempts to predict if it is eye or not.

Building a rule that makes highly accurate predictions on new test examples is a challenging task. However, it is not hard to come up with rough weak classifiers that are only moderately accurate. An example of such a rule for the problem under consideration is something like the following: “If the pixel p located in the sclera region of the patch under consideration is not white, then predict it is non-eye”. In this case such a rule is related the knowledge that the white region corresponding to the sclera should be present in an eye patch. On the other hand, such a rule will cover all possible non-eyes cases; for instance, it is correct to say nothing about what to predict if the pixel p is white. Of course, this rule will make predictions that are significantly better than random guessing. The key idea is to find many weak classifiers and combine them in a proper way deriving a single strong classifier.

Among other, Boosting [54, 129, 130] is one of the most popular procedure for combining the performance of weak classifiers in order to achieve a better classifier. We use a boosting procedure on patches represented as gray codes to build a strong classifier useful to distinguish between eye and non-eye patches. Specifically, boosting is used to select the positions $\{p_1, \ldots, p_n\}$ corresponding to n gray code bits that best discriminate between the classes eye vs. non-eye, together with n associated weak classifiers of the form:

$$h_i(\mathbf{g}) = \begin{cases} a_i & g_{p_i} = 1 \\ b_i & g_{p_i} = 0 \end{cases} \tag{3.20}$$

where $\mathbf{g} = [g_1, g_2, \ldots, g_D]$ is the gray code vector ($g_i \in \{0,1\}$) of size $D = 30 \times 30 \times 3 \times 8$ corresponding to a 30×30 patch extracted as described in previous Section. The parameters a_i and b_i are automatically learned by *Gentleboost* procedure [54] as explained in Section 3.3.3. The classification is obtained considering the sign of the learned additive model:

$$H(\mathbf{g}) = \sum_{i=1}^{n} h_i(\mathbf{g}) \tag{3.21}$$

where $n \ll D$ indicates the number of weak classifiers involved in the strong classifiers H.

The rationale beyond the use of gray code representation is the following. In the gray code space just a subset of all possible bit combinations are related the eyes patches. We wish to select those bits that usually differ in terms of binary value between eye and non-eye patches. Moreover, by using gray code representation rather than classic bit planes decomposition we reduce the impact of small changes in intensity of patches that could produce significant variations in the corresponding binary code [60].

In Figure 3.21 an example of $n = 1000$ gray code bits selected with *Gentleboost* procedure is reported. Selected bits are shown as black or white points on the different gray code planes. This map indicates that a red-eye patch should have 1 in the position coloured in white and 0 in the positions colored in black. Once gray code bits and the corresponding weak classifiers parameters are learned, a new patch can be classified by using the sign of Equation 3.21.

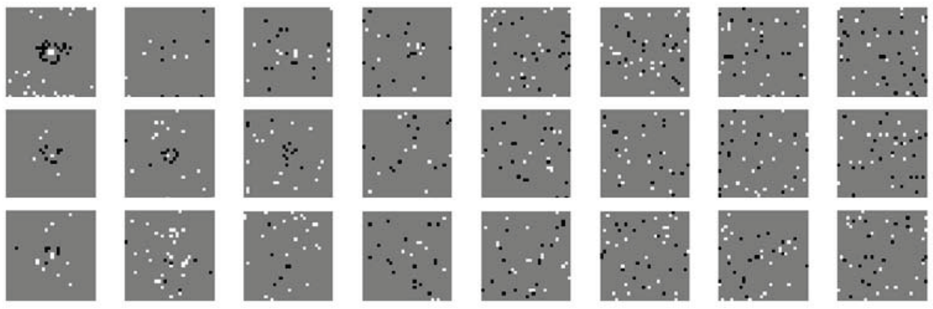


Figure 3.21 : Selected gray code bits.

The approach described above does not take into account spatial relationship between selected gray code bits. Spatial information is useful to make stronger the classification task (e.g., pupil is surrounded of sclera). To overcome this problem we coupled the gray codes bits selected at the first learning stage using *xor* operator to obtain a new set of n^2 binary features. We randomly select a subset containing m of these features and performed a second round of *Gentleboost* procedure to select the most discriminative spatial relationship among the *m* randomly selected. This new classifier is combined with the one learned previously to perform final eye and non-eye patches classification.

Due to the multi-modally nature of the patches involved in our problem (i.e., colours, orientation, shape, etc.), a single discriminative classifier could fail during classification task. To get through this weakness we propose to perform first a clustering of the input space and then apply the two stage boosting approach described above on each cluster. More specifically, during the learning phase, the patches are clustered by using K-means [42] in their original color space producing the subsets of the input patches with the relative prototypes; hence the two stage of boosting described above are performed on each cluster. During the classification stage, a new patch is first assigned to a cluster according to the closest prototype and then classified taking into account the two additive models properly learned for the cluster under consideration.

Experimental results reported in Section 3.3.5 confirm the effectiveness of the proposed strategy.

3.3.3 Boosting for binary classification exploiting gray codes

Boosting provides a way to sequentially fit additive models of the form in Equation 3.21 optimizing the following cost function [54]:

$$J = E[e^{-yH(\mathbf{g})}], \tag{3.22}$$

where $y \in \{-1,1\}$ is the class label associated to the feature vector $\mathbf{g}$. In this work *y=1* is associated to the *eye* class, whereas *y=-1* is the label associated to the *non-eye* class. The cost function in the Equation 3.22 can be thought as a differentiable upper bound of the misclassification rate [130].

There are many ways to optimize this function. A simple and numerical robust way to optimize this function is called *Gentleboost* [54]. This version of boosting procedure outperforms other boosting variants for computer vision tasks (e.g., face detection) [90]. In *Gentleboost* the optimization of Equation 3.22 is performed minimizing a weighted squared error at each iteration [141]. Specifically at each iteration i the strong classifier H is updated as $H(\mathbf{g}) := H(\mathbf{g}) + h_{best}(\mathbf{g})$ where the weak classifier h_{best} is selected in order to minimize the second order approximation of the cost function in the Equation 3.22:

$$h_{best} = \arg\min_{h_d} J(H(\mathbf{g}) + h_d(\mathbf{g})) \simeq \arg\min_{h_d} E[e^{-yH(\mathbf{g})}(y - h_d(\mathbf{g}))^2] \tag{3.23}$$

Defining as $w_j = e^{-y_j H(\mathbf{g}_j)}$ the weight for the training sample j and replacing the expectation with an empirical average over the training data, the optimization reduces in minimizing the weighted squared error:

$$J_{wse}(h_d) = \sum_{j=1}^{M} w_j (y_j - h_d(\mathbf{g}_j))^2, \tag{3.24}$$

where M is the number of samples in the training set.

The minimization of J_{wse} depends on the specific form of the weak classifiers h_d. Taking into account the binary representation of samples (i.e., the gray code of each patch), in the present proposal we define the weak classifiers as follows:

$$h_d(\mathbf{g}) = \begin{cases} a_d & if\, g_d = 1 \\ b_d & if\, g_d = 0 \end{cases} \tag{3.25}$$

In each iteration the optimal a_d and b_d for each possible h_d can be obtained through weighted least squares as follows:

$$a_d = \frac{\sum_{j=1}^{M} w_j y_j \delta(g_d = 1)}{\sum_{j=1}^{M} w_j \delta(g_d = 1)} \tag{3.26}$$

$$b_d = \frac{\sum_{j=1}^{M} w_j y_j \delta(g_d = 0)}{\sum_{j=1}^{M} w_j \delta(g_d = 0)} \tag{3.27}$$

The best weak classifier h_{best} is hence selected in each iteration of the boosting procedure such that the cost of Equation 3.24 is the lowest:

$$h_{best} = \arg\min_{h_d} J_{wse}(h_d) \tag{3.28}$$

Finally, before a new iteration the boosting procedure makes the following multiplicative update to the weights corresponding to each training sample:

$$w_j := w_j e^{-y_j h_{best}(\mathbf{g}_j)} \tag{3.29}$$

This update increases the weight of samples which are misclassified (i.e., for which $y_j H(\mathbf{g}_j) < 0$), and decreases the weight of samples which are correctly classified.

The procedures employed for learning and classification on the proposed representation are summarized in Algorithm 1 and Algorithm 2. In the learning stage we initialize the weights corresponding to the elements of the training set such that the number of the samples within each class is taken into account. This is done to overcome the problems that can occur due to the unbalanced number of training samples within the considered classes.

3.3.4 Red-Eyes Correction

Once the red-eyes have been detected the correction step is performed. Usually the red-eye artifact consists of a red pupil with a white glint. This area is devoted to absorb light and thus should be dark. To transform the red pupil to a dark

Algorithm 1: Learning

```
Input: A set of gray code vectors $\mathbf{G} = \{\mathbf{g}_1, \dots, \mathbf{g}_M\}$, and corresponding labels
        $\mathbf{Y} = \{y_1, \dots, y_M\}$
Output: A strong classifier $H(\mathbf{g}) = \sum_{i=1}^{n} h_i(\mathbf{g})$
begin
    $C^+ := \{j | y_j = 1\}$
    $C^- := \{j | y_j = -1\}$
    $w_{j \in C^+} := \frac{1}{2|C^+|}$
    $w_{j \in C^-} := \frac{1}{2|C^-|}$
    for $i = 1, 2, \dots, n$ do
        for $d = 1, 2, \dots, D$ do
            $a_d^* := \frac{\sum_{j=1}^{M} w_j y_j \delta(g_d=1)}{\sum_{j=1}^{M} w_j \delta(g_d=1)}$
            $b_d^* := \frac{\sum_{j=1}^{M} w_j y_j \delta(g_d=0)}{\sum_{j=1}^{M} w_j \delta(g_d=0)}$
            $h_d^*(\mathbf{g}) := \begin{cases} a_d^* & if\, g_d = 1 \\ b_d^* & if\, g_d = 0 \end{cases}$
            $J_{wse}(h_d^*) := \sum_{j=1}^{M} w_j (y_j - h_d^*(\mathbf{g}_j))^2$
        end
        $p_i := \arg\min_d J_{wse}(h_d^*)$
        $a_i := a_{p_i}^*$
        $b_i := b_{p_i}^*$
        $h_i(\mathbf{g}) := \begin{cases} a_i & if\, g_{p_i} = 1 \\ b_i & if\, g_{p_i} = 0 \end{cases}$
        $w_j := w_j e^{-y_j h_i(\mathbf{g}_j)}$
    end
    $H(\mathbf{g}) := \sum_{i=1}^{n} h_i(\mathbf{g})$
end
```

Algorithm 2: Classification

```
Input: The strong classifier $H$, and a new gray code sample $\mathbf{g}$ to be classified
Output: The inferred class $y \in \{-1, 1\}$
begin
    $y := sign(H(\mathbf{g}))$
end
```

region, a de-saturation and a brightness reduction is accomplished [56,106]. The region of connected red pixels is used to fix the area that must be desaturated. To prevent unpleasant transition from the iris to the pupil, red-eye artifact is replaced by a mask with equal dimensions where each value is used as weighted brightness/de-saturation reduction factor. The correction mask M is based on a 32×32 fixed point LUT with Gaussian shape (Figure 3.22). The mask is resized through a bilinear resampling to fit the dimension of the region of connected red pixels under consideration.

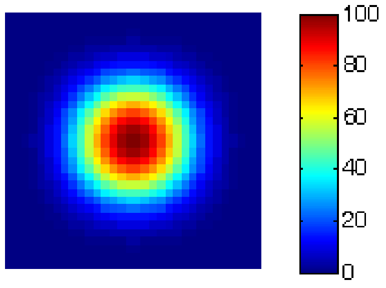


Figure 3.22 : Brightness-Saturation Mask

Let I_c^r the channel $c \in \{R,G,B\}$ of a region of interest r within the image I. For each channel $c \in \{R,G,B\}$ the pixels (x,y) belonging to the region I^r are corrected as follows:

$$I_c^r(x,y) = \begin{cases} I_c^r(x,y) & [I_R^r(x,y), I_G^r(x,y), I_B^r(x,y)] \in W \\ \frac{I_G^r(x,y)}{M(x,y)} & otherwise \end{cases} \quad (3.30)$$

where W is a surrounding of the “white” color which can slightly vary in terms of lightness, hue and saturation. This means that, to prevent glint from disappearing only red pixels are de-satured (the whitish pixels are excluded from the brightness processing).

3.3.5 Experimental Settings and Results

The proposed red-eye removal pipeline has been tested on a dataset of 390 images in which 1049 red-eyes have been manually labeled. The dataset has been collected from various sources, including digital single-lens reflex (DSLR) cameras, compact cameras, personal collections and Internet photos. Single red-eyes, as well as high variability of red-eyes colors, poses and shapes have been considered in building the dataset. In order to accurately assess the proposed approach, the size of the eyes to be detected in the collected images must be small enough to ensure that also the smallest red eyes can be detected and corrected. The basic requirement considered in our experimental phase is that the red eyes must be accurately detected and corrected up to three meter distance from the camera. Table 3.2 presents the estimated eye sizes, in pixels, for XGA image size (1024×768), with the assumption that the average eye is directed to the camera. In this paper the collected images have been considered with a XGA image resolution and the minimum and maximum estimated pupil diameter (Table 3.2) have been taken into account in building the dataset for testing purposes.

Table 3.2 : Estimated eye sizes taking into account the distance from the camera.

Distance (m)	0.20	0.40	0.60	0.80	1.00	1.20	1.40	1.60	1.80	2.00	2.20	2.40	2.60	2.80	3.00
Diameter (pixels)	52	26	17	13	10	9	7	7	6	5	5	4	4	4	3

For each image of the dataset, the pixels belonging to red eyes artifacts have been manually labeled as red-eye-pixels. The parameters Min_h, Max_h, t_s, Min_s, Max_s, Min_ρ, Max_ρ, and Max_η involved in the first stage of the proposed approach (see Section 3.3.1) have been learned taking into account the true and false red-eyes-pixels within the labeled dataset. To this aim, a full search procedure on a grid of equispaced points in the eight dimensional parameters' space was employed. For each point of the grid, the correct detection and false positives rates of the true red-eyes-pixels within the dataset was obtained. The tuple of parameters with the best trade-off between correct detection and the false positives have been used to perform the final filtering pipeline. A similar procedure was employed to determine the subspace W of the RGB space involved in the correction step to identify pixels belonging to the glint area.

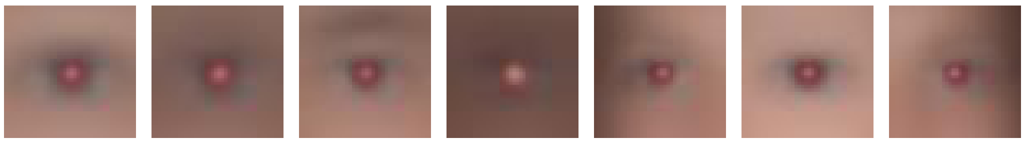


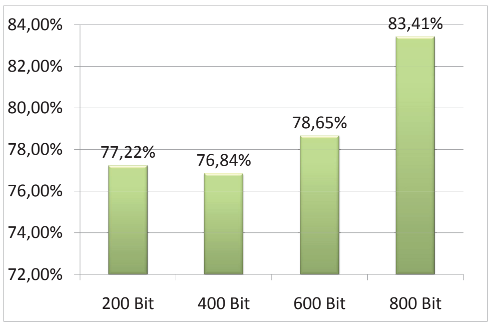
Figure 3.23 : Example of clusters prototypes obtained in a LOOCV run.

In order to evaluate the classification performance of the proposed method, the leave-one-out cross validation procedure (LOOCV) have been employed. Each run of LOOCV has involved a single image as test, and the remaining images as training data. This is repeated to guarantee that each input image is used once as test image. At each run of LOOCV the parameters of the filtering pipeline have been set to maximize correct detection and minimize false positives. At each run of LOOCV the training images have been clustered and then the two stage boosting approach described in Section 3.3.2 have been performed on each cluster. Seven clusters (Figure 3.23) and 800 binary features for the additive classifiers corresponding to the clusters have been used on each LOOCV run. The maximum number of iterations used by boosting procedure to obtain the 800 binary feature was 1400. The final results have been obtained averaging on the results of the overall LOOCV runs.

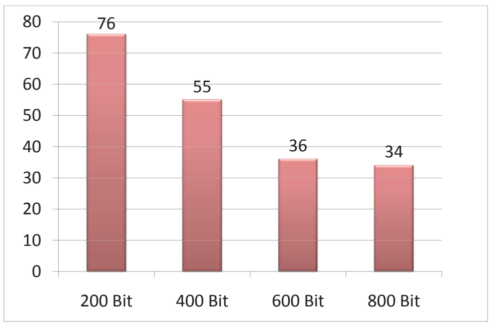
Taking into account both, the filtering and the classification stages, the hit-rate of the proposed red-eyes detector is 83.41%. This means that 875 red-eyes have been correctly detected with respect to the 1049 red-eyes of the 390 input images, whereas only 34 false positives have been introduced. In Figure 3.24 the training ability increasing the number of bits is shown in terms of Hit Rates (Figure 3.24(a)) and False Positives (Figure 3.24(b)).

In Figure 3.25 two examples of misclassified patches are reported. Fig. 3.25 (a) a “golden” eye is depicted (another possible artifact due to similar acquisition problem). The underlying structure in Fig. 3.25 (b) is probably the main reason of misclassification.

In order to point out the usefulness of the proposed cluster-based boosting, as

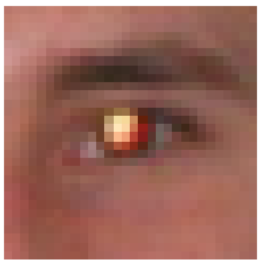


(a) Hit Rates



(b) False Positives

Figure 3.24 : Performances increasing number of bits.


(a) False Negative.

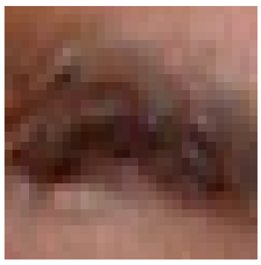

(b) False Positive.

Figure 3.25 : Examples of misclassified patches.

well as the usefulness of the spatial relationship introduced by using *xor* operation on gray codes bits, we have repeated tests considering different configurations. Results reported in Table 3.3 confirm the effectiveness of the rationale beyond the proposed method.

Configuration	**Hit Rate**	**False Positives**
Gray Codes	75.98%	47
Gray Codes + Clustering	77.51%	44
Gray Codes + XOR	79.31%	36
Gray Codes + Clustering + XOR	**83.41%**	**34**

Table 3.3 : Comparison of different configurations.

To properly evaluate the overall red-eyes removal pipeline, the qualitative criterion (see Section 3.2.4) was adopted to compare the proposed solution with respect to existing automatic solutions.

The proposed pipeline has been compared with respect to the following automatic (mainly commercial) solutions: Volken [143], NikonView V6.2.7, KodakEasyShare V6.4.0, StopRedEye! V1.0, HP RedBot, Arcsoft PhotoPrinter V5, Cyberlink MediaShow. Experiments have been done using effective commercial software and the implementation of [143] provided by the authors. NikonView approach is mainly based on [38].

Method	$\mathbf{FN_m}$	$\mathbf{FN_d}$	$\mathbf{FP_c}$	$\mathbf{FP_n}$	$\mathbf{N_p}$	$\mathbf{C_i}$	$\mathbf{C_n}$	$\mathbf{Q_c}$	**Hit Rate**
Cyberlink MediaShow	270	86	40	19	39	122	**61**	0.1423	66.06%
Volken et alii [143]	179	117	150	1540	83	**17**	79	-0.5851	71.78%
KodakEasyShare V6.4.0	194	99	5	20	**5**	104	100	0.4243	72.07%
HP RedBot	174	109	26	45	85	99	150	0.2345	73.02%
NikonView V6.2.7	143	116	6	29	88	124	129	0.2944	75.31%
StopRedEye! V1.0	124	125	8	12	83	81	91	0.4161	76.26%
Arcsoft PhotoPrinter V5	132	103	10	78	80	89	82	0.3800	77.60%
Battiato et alii [19]	122	85	**2**	**2**	60	20	64	**0.6346**	80.26%
Proposed Pipeline	**114**	**60**	9	25	46	34	79	0.6174	**83.41%**

Table 3.4 : Quality score of different red-eyes removal approaches.

As reported in Table 3.4, the proposed approach has obtained the best performances in terms of both, hit rate and quality criterion. Moreover, the proposed approach outperforms the method we have presented in [19] also in terms of computational complexity. Despite the complete set of images used in the experiments is not publicly available, since most of the photos are taken from private collections, some examples with corresponding results are available at the following web address:
http://iplab.dmi.unict.it/download/EurasipSpecialIssue2010.

3.3.5.1 Computational Complexity

To evaluate the complexity, a deep analysis has been performed by running the proposed pipeline on an ARM926EJ-S processor instruction set simulator. We have chosen this specific processor because it is widely used in embedded mobile platforms. The CPU run at 300 MHz, and both data and instruction caches have been fixed to 32 KB. The bus clock has been set to 150 MHz, and the memory read/write access time is 9 ns. The algorithm has been implemented using bitwise operators to work on colour maps and fixed point operations. Due to the dependence of the operations to the number of red clusters found in the image, we have analyzed a mid case, that is an image containing around 40 potential red eye zones, but only 2 of them are real eyes to be corrected.

Table 3.5 contains a report of the performances of the main steps of the proposed pipeline, assuming to work on a XGA version (scaled) of the image: the redness detection (Color Map), the processing on the generated maps (Morphological Operations), the candidate extraction, the classification step and finally the correction of the identified eyes. The performances information reported in Table 3.5 are related to the following computational resources:

Instructions: counts the executed ARM instructions;

Core cycles: core clock ticks needed to make the Instructions;

Data (D$): Read/Write Hits and Misses, cache memory hits and misses;

Seq and Non Seq: sequential and non-sequential memory accesses;

Idle: represents bus cycles when the instruction bus and the data bus are idle, that is when the processor is running;

Busy: counts busy bus cycles, that is when the data are transferred from the memory into the cache;

Wait States: the number of bus cycles introduced when waiting for accessing the RAM (is an indicator of the impact of memory latencies);

Total: is the total number of cycles required by the specific function, expressed in terms of bus cycles;

Milliseconds: time required by the specific function expressed in milliseconds.

Table 3.5 : Performances of the main steps of the proposed pipeline.

	Color Map	**Morph. Oper.**	**Candidate Extr.**	**Classification**	**Correction**
Instructions	19.845.568	22.990.051	9.418.650	4.446.349	1.698.946
Core Cycles	28.753.276	30.489.180	16.407.293	5.668.496	2.390.279
D$ R Hits	4.722.760	2.903.178	2.504.092	945.959	205.188
D$ W Hits	97.636	261.213	428.924	135.634	94.727
D$ R Misses	75.495	6.293	5.666	3.450	244
D$ W Misses	2	193.891	3.290	24.069	1.133
SEQ	538.136	17.486.089	48.539	40.177	4.100
NON-SEQ	77.321	122.234	9.841	22.366	1.533
IDLE	16.282.401	7.325.256	10.345.379	3.203.188	1.372.407
Wait States	615.457	253.103	58.380	62.543	5.633
Total	17.513.316	16.208.789	10.462.139	3.328.274	1.383.673
Milliseconds	**117**	**108**	**70**	**22**	**9**

The overall time achieved on this mid-case is 326 ms. The table highlights the efficiency of the classifier, because it is mainly based on bit comparisons. Considering patches scaled at 32×32 before the classification stage, the classifier is essentially a comparison of 32×32 bit words for each channel with complexity in the range of one operation per pixel. For this reason it is very fast and light. Also the correction is very light because, as explained in Section 3.3.4, it is based on the resampling of a precomputed Gaussian function. The impact on memory is valuable only on the map processing, where data are processed several times, whereas in the remaining steps of the pipeline the weight of the instructions determines the main part of process timing.

We cannot compare the performances and complexity of our methodology with other methods because the other proposed methods are commercial ones, hence the related codes are not available for the analysis.

3.4 Conclusion and Future Works

In this work an advanced red-eyes removal pipeline has been discussed. After an image filtering pipeline devoted to select only the potential regions in which red-eye artifacts are likely to be, a cluster-based boosting on grey codes based features is employed for classification purpose. Red-eyes are then corrected through de-saturation and brightness reduction. Experiments on a representative dataset confirm the real effectiveness of the proposed strategy which also allows to properly managing the multi-modally nature of the input space. The obtained results have pointed out a good trade-off between overall hit-rate and false positives. Moreover, the proposed approach has shown good performance in terms of quality measure. Future works will be devoted to include the analysis of other eyes artifacts (e.g., “golden eyes”).

4. Exposure Correction Feature Dependent

The problem of the proper exposure settings for image acquisition is of course strictly related with the dynamic range of the real scene. In many cases some useful insights can be achieved by implementing ad-hoc metering strategies. Alternatively, it is possible to apply some tone correction methods that enhance the overall contrast of the most salient regions of the picture. The limited dynamic range of the imaging sensors doesn't allow to recover the dynamic of the real world. In this Chapter we present a brief review of automatic digital exposure correction methods trying to report the specific peculiarities of each solution. Starting from exposure metering techniques, which are used to establish the correct exposition settings, we describe automatic methods to extract relevant features and perform corrections.

4.1 Introduction

One of the main problems affecting image quality, leading to unpleasant pictures, comes from improper exposure to light. Beside the sophisticated features incorporated in today's cameras (i.e., automatic gain control algorithms), failures are not unlikely to occur. Digital consumer devices make use of ad-hoc strategies and heuristics to derive exposure setting parameters. Typically such techniques are completely blind with respect to the specific content of the involved scene. Some techniques are completely automatic, cases in point being represented by those based on average/automatic exposure metering or the more complex matrix/intelligent exposure metering. Others, again, accord to the photographer a certain control over the selection of the exposure, thus allowing space for personal taste or enabling him to satisfy particular needs. In spite of the great variety of methods for regulating the exposure and the complexity of some of them, it is not rare for images to be acquired with a nonoptimal or incorrect exposure. This is particularly true for handset devices (e.g., mobile phones) where several factors contribute to acquire bad-exposed pictures: poor optics, absence of flashgun, not to talk about difficult input scene lighting conditions, and so forth.

There is no exact definition of what a correct exposure should be. It is possible to abstract a generalization and to define the best exposure that enables one to reproduce the most important regions (according to contextual or perceptive criteria) with a level of gray or brightness, more or less in the middle of the possible range. In any case if the dynamic range of the scene is sensibly ”high” there is no way to acquire the overall involved details. One of the main issues of this Chapter is devoted to give an effective overview of the technical details involved in:

- Exposure settings of imaging devices just before acquisition phase (i.e., preprocessing phase) [99];
- Content-dependent enhancement strategies applied as post-processing [13];
- Advanced solution where multi-picture acquisition of the same scene with different exposure time allows to reproduce the Radiance map of the real world [16].

.

The rest of the Chapter is organized as follows. The initial Section will discuss in details traditional and advanced approaches related to the pre-processing phase (i.e., the sensor is read continuously and the output is analyzed in order to determine a set of parameters strictly related with the quality of the final picture [99]). The role of exposure setting will be analyzed also considering some case studies where, by making use of some assumptions about the dynamic range of the real scene, it is possible to derive effective strategies. Section 4.3 will describe the work presented in [13] where by using post-processing techniques, an effective enhancement has been obtained just analyzing some content dependent features of the picture.

4.2 Exposure Metering Techniques

Metering techniques built into the camera are getting much better with the introduction of computer technology but limitations still remain. For example taking a picture on a snow scene or trying to photograph a black locomotive without overriding the camera calculated metering is very difficult. The most important aspect of the exposure duration is to guarantee that the acquired image falls in a good region of the sensors sensitivity range. In many devices, the selected exposure value is the main processing step for adjusting the overall image intensity that the user will see. Many of the first digital cameras used a separate metering system to set exposure duration, rather than using data acquired from the sensor chip. Integrating exposure-metering function into the main sensor (through-the-lens, or TTL, metering) may reduce system cost. The imaging community uses a measure called *Exposure Value* (EV) to specify the relationship between the f-number[1], F, and exposure duration, T :

$$EV = \log_2(\frac{F^2}{T}) = 2\log_2(F) - \log_2(T) \tag{4.1}$$

The exposure value (4.1) becomes smaller as the exposure duration increases, and it becomes larger as the f-number grows. Most auto-exposure algorithms work in this way:

1. Take a picture with a pre-determined exposure value (EV_{pre});
2. Convert the RGB values to luminance, L;
3. Derive a single value L_{pre} (like center-weighted mean, median, or more complicated weighted method as in matrix-metering) from the luminance picture;
4. Based on linearity assumption and equation (4.1), the optimum exposure value EV_{opt} should be the one that permits a correct exposure. The picture taken at this EV_{opt} should give a number close to a pre-defined ideal value L_{opt}, thus:

$$EV_{opt} = EV_{pre} + \log_2(L_{pre}) - \log_2(L_{opt}) \tag{4.2}$$

The ideal value L_{opt} for each algorithm is typically selected empirically. Different algorithms mainly differ in how the single number L_{pre} is derived from the picture. In Fig.(**4.1**) an example Table of EVs, which take into consideration different exposure times and f-numbers, is reported.

[1]f-numbers, or aperture values, are measurement of the size of the hole that the light passes through the rear of the lens, relative to the focal length. The smaller the f-number, the more light gets through the lens.

EV	*f-number*												
	1.0	**1.4**	**2.0**	**2.8**	**4.0**	**5.6**	**8.0**	**11**	**16**	**22**	**32**	**45**	**64**
−6	60	2 m	4 m	8 m	16 m	32 m	64 m	128 m	256 m	512 m	1024 m	2048 m	4096 m
−5	30	60	2 m	4 m	8 m	16 m	32 m	64 m	128 m	256 m	512 m	1024 m	2048 m
−4	15	30	60	2 m	4 m	8 m	16 m	32 m	64 m	128 m	256 m	512 m	1024 m
−3	8	15	30	60	2 m	4 m	8 m	16 m	32 m	64 m	128 m	256 m	512 m
−2	4	8	15	30	60	2 m	4 m	8 m	16 m	32 m	64 m	128 m	256 m
−1	2	4	8	15	30	60	2 m	4 m	8 m	16 m	32 m	64 m	128 m
0	1	2	4	8	15	30	60	2 m	4 m	8 m	16 m	32 m	64 m
1	1/2	1	2	4	8	15	30	60	2 m	4 m	8 m	16 m	32 m
2	1/4	1/2	1	2	4	8	15	30	60	2 m	4 m	8 m	16 m
3	1/8	1/4	1/2	1	2	4	8	15	30	60	2 m	4 m	8 m
4	1/15	1/8	1/4	1/2	1	2	4	8	15	30	60	2 m	4 m
5	1/30	1/15	1/8	1/4	1/2	1	2	4	8	15	30	60	2 m
6	1/60	1/30	1/15	1/8	1/4	1/2	1	2	4	8	15	30	60
7	1/125	1/60	1/30	1/15	1/8	1/4	1/2	1	2	4	8	15	30
8	1/250	1/125	1/60	1/30	1/15	1/8	1/4	1/2	1	2	4	8	15
9	1/500	1/250	1/125	1/60	1/30	1/15	1/8	1/4	1/2	1	2	4	8
10	1/1000	1/500	1/250	1/125	1/60	1/30	1/15	1/8	1/4	1/2	1	2	4
11	1/2000	1/1000	1/500	1/250	1/125	1/60	1/30	1/15	1/8	1/4	1/2	1	2
12	1/4000	1/2000	1/1000	1/500	1/250	1/125	1/60	1/30	1/15	1/8	1/4	1/2	1
13	1/8000	1/4000	1/2000	1/1000	1/500	1/250	1/125	1/60	1/30	1/15	1/8	1/4	1/2
14		1/8000	1/4000	1/2000	1/1000	1/500	1/250	1/125	1/60	1/30	1/15	1/8	1/4
15			1/8000	1/4000	1/2000	1/1000	1/500	1/250	1/125	1/60	1/30	1/15	1/8
16				1/8000	1/4000	1/2000	1/1000	1/500	1/250	1/125	1/60	1/30	1/15

Figure 4.1 : Example of fixed exposure times.

4.2.1 Classical Approaches

The metering system in typical imaging device measures the amount of light in the scene and calculates the best-fit exposure value based on the metering mode explained below. Automatic exposure is a standard feature in all digital cameras. After having selected the metering mode, it is requested just pointing the camera and pressing the shutter release. The metering method defines which information of the scene is used to calculate the exposure value and how it is determined. Cameras generally allow the user to select between spot, center-weighted average, or multi-zone metering modes.

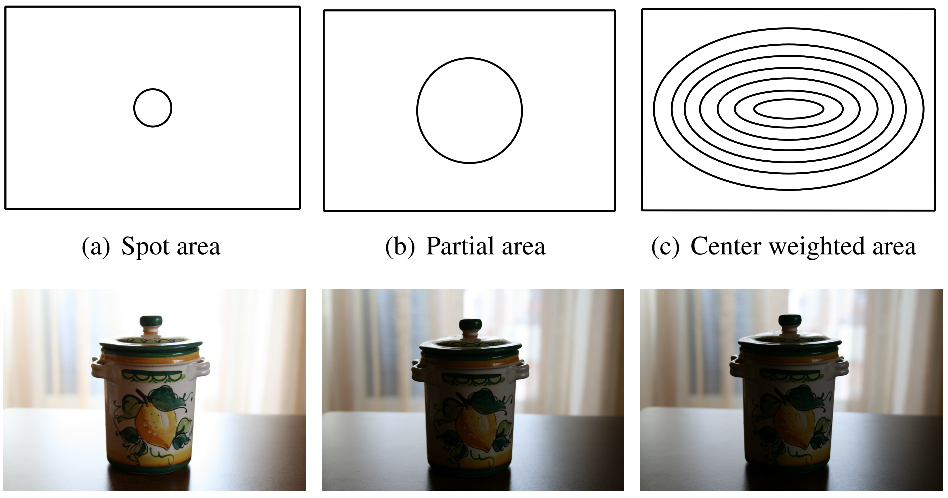
4.2.1.1 Spot Metering

Spot metering allows user to meter the subject in the center of the frame (or on some cameras at the selected AutoFocus (AF) point). Only a small area of the whole frame (between 1-5% of the viewfinder area) is metered while the rest of the frame is ignored. In this case L_{pre} (4.2) is the mean of the center area (see Fig.(**4.2(a)**)). This will typically be the effective center of the scene, but some cameras allow the user to select a different off-center spot, or to recompose by moving the camera after metering. A few models support a Multi-Spot mode which allows multiple spot meter readings to be taken of a scene that are averaged. Both of those cameras and others also support metering of highlight and shadow areas. Spot metering is very accurate and is not influenced by other areas in the frame. It is commonly used to shoot very high contrast scenes. For example (see Fig.(**4.2(d)**)), if the subject's back is being hit by the rising sun and the face is a lot darker than the bright halo around the subject's back and hairline (the subject is ”backlit”), spot metering allows the photographer to measure the light bouncing off the subject's face and expose properly for that, instead of the much brighter light around the hairline. The area around the back and hairline will then become over-exposed. Spot metering is a method upon which the Zone System depends[2].

4.2.1.2 Partial Area Metering

This mode meters a larger area than spot metering (around 10-15% of the entire frame), and is generally used when very bright or very dark areas on the edges of the frame would otherwise influence the metering unduly. Like spot metering, some cameras can use variable points to take readings from (in general autofocus points), or have a fixed point in the center of the viewfinder. In Fig.(**4.2(e)**) an

[2]The Zone System is a photographic technique for determining optimal film exposure and development, formulated by Ansel Adams and Fred Archer in 1941. The Zone System provides photographers with a systematic method of precisely defining the relationship between the way they visualize the photographic subject and the final results. Although it originated with black and white sheet film, the Zone System is also applicable to roll film, both black and white and color, negative and reversal, and to digital photography.


(a) Spot area (b) Partial area (c) Center weighted area

(d) Spot example (e) Partial area example (f) Center weighted example

Figure 4.2 : Metering examples.

example of partial metering on a backlight scene is shown; this method permits to equalize much more the global exposure.

4.2.1.3 Center-weighted Average Metering

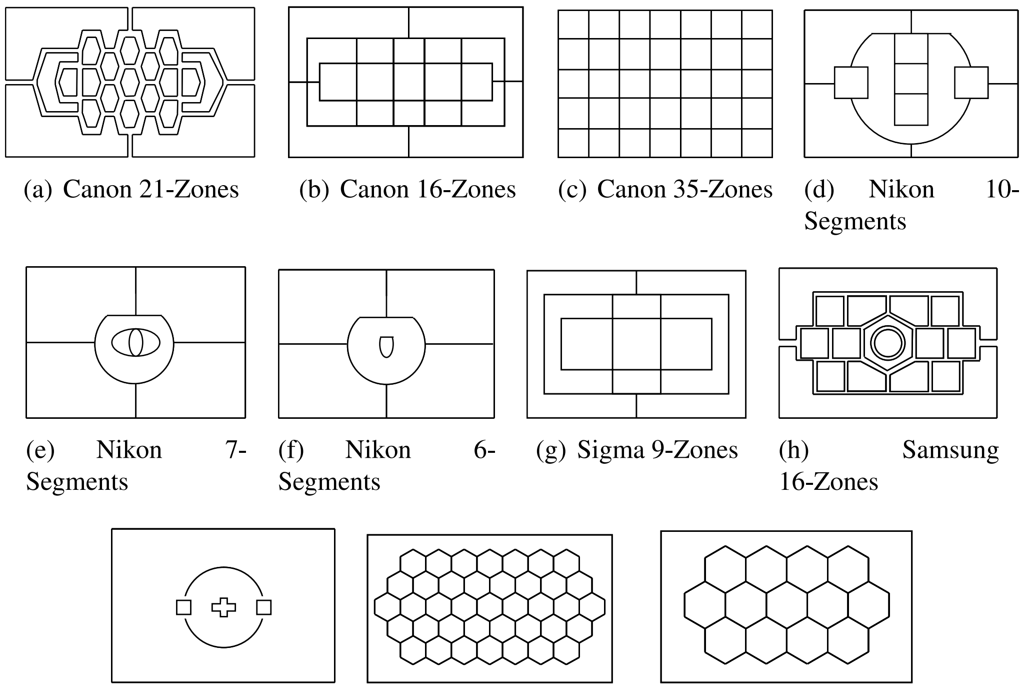
This method is probably the most common metering method implemented in nearly every digital camera: it is also the default for those digital cameras which don't offer metering mode selection. In this system, as described in Fig.(**4.2(c)**), the meter concentrates between 60 to 80 percent of the sensitivity towards the central part of the viewfinder. The balance is then "feathered" out towards the edges. Some cameras allow the user to adjust the weight/balance of the central portion to the peripheral one. One advantage of this method is that it is less influenced by small areas that vary greatly in brightness at the edges of the viewfinder; as many subjects are in the central part of the frame, consistent results can be obtained. Unfortunately, if a backlight is present into the scene the central part results darker than the rest of the scene (Fig.(**4.2(f)**)), and unpleasant underexposed foreground is produced.

4.2.1.4 Average Metering

In this mode the camera will use the light information coming from the entire scene and averages for the final exposure setting, giving no weighting to any particular portion of the metered area. This metering technique has been replaced by Center-Weighted metering, thus is only obsolete and present in older cameras only.

4.2.2 Advanced Approaches

4.2.2.1 Matrix or Multi-zone Metering



(a) Canon 21-Zones (b) Canon 16-Zones (c) Canon 35-Zones (d) Nikon 10-Segments

(e) Nikon 7-Segments (f) Nikon 6-Segments (g) Sigma 9-Zones (h) Samsung 16-Zones

(i) Olympus ESP (j) Konica Minolta 40 zone honeycombs (k) Konica Minolta 14 zone honeycombs

Figure 4.3 : Examples of different kind of Multi-zone Metering mode used by several cameras manufacturers.

This mode is also called matrix, evaluative, honeycomb, segment metering, or esp (electro selective pattern) metering on some cameras. It was first introduced by the Nikon FA, where it was called Automatic Multi-Pattern metering. On a number of cameras, this is the default/standard metering setting. The camera measures the light intensity in several points of the scene, and then combines the results to find the settings for the best exposure. How they are combined/calculated deviates from device to device. The actual number of zones used varies wildly, from several to over a thousand. However performance should not be concluded on the number of zones alone, or the layout. As shown in Fig.(**4.3**) the layout can change drastically from a manufacturer to another, also within the same company the use of different multi-zone metering can change due to several reasons (e.g., the dimension of the final pixel matrix).

Many manufacturers are less than open about the exact calculations used to determine the exposure. A number of factors are taken into consideration, these include: AF point, distance to subject, areas in-focus or out-of-focus, colors/hues of the scene, and backlighting. Multi-zone tends to bias its exposure towards the autofocus point being used (while taking into account other areas of the frame too), thus ensuring that the point of interest has been properly exposed (it is also designed to avoid the need to use exposure compensation in most situations). A database of many thousands of exposures is pre-stored in the camera, and the processor can use a selective pattern to determine what is being photographed. Some cameras allow the user to link (or unlink) the autofocus and metering, giving the possibility to lock exposure once AF confirmation is achieved, AEL (Auto Exposure Lock). Using manual focus, and on many compacts cameras, the AF point is not used as part of the exposure calculation, in such instances it is common for the metering to default to a central point in the viewfinder, using a pattern based off of that area. Some users have problems with wide angle shots in high contrast, due to the large area which can vary greatly in brightness, it is important to understand that even in this situation, the focus point can be critical to the overall exposure.

4.2.3 Exposure Control-System

In many conventional digital cameras, such as Digital Single Lens Reflex (DSLR), the exposure control systems are implemented using mechanical devices. Such mechanical devices include a mechanical iris and/or a mechanical shutter wheel. The most common implementation, the mechanical iris, varies the rate at which the sensor receives photons. The mechanical shutter varies the amount of time during which the sensor receives photons.

Since mechanical devices have a relatively low reliability, slow response time, and increase the size and the cost of lenses, mobile devices have been fitted up with exposure control systems which take into account integration time and multiplication gain factors (see Fig.(**4.4**)).

The exposure control system performs a first lecture of pixels values, directly on Bayer data coming from the sensor and analyzes the pixel values through a statistic processing block. The algorithm involved into this statistical analysis can be a simple statistical (weighted) mean brightness (see Section 4.2.1.3) or a more sophisticated metering (see Section 4.2.2).

The viewfinder pipeline is used to estimate the correct exposure. This pipeline is a simplified version of the image generation pipeline, and is used to show a preview of captured image on the embedded display. The exposure control system, described in Fig.(**4.4**), performs the following steps:

1. Initial integration time consideration: a first image is captured and is given to the system, which perform the statistical analysis.

2. Calculate ideal gain: to achieve a first correction an ideal multiplication gain is is used to accordingly set both analog and digital gains.

3. Determine analog gain: the analog gain is estimated through statistical measures.

4. Finalize integration time: the integration time is then fixed.

5. Determine digital gain: the digital gain is estimated through statistical measures.

6. Determine final exposure: the final exposure is thus estimated and the image capture is achieved.

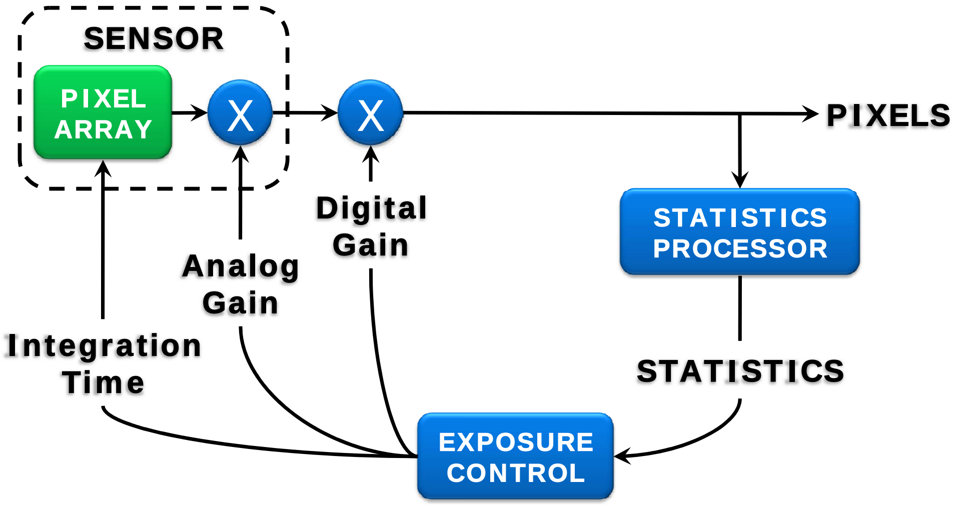


Figure 4.4 : Exposure Control-System pipeline.

4.3 Exposure Correction

As explained in Section 4.2, a good exposure should be able to reproduce the most important regions (according to contextual or perceptive criteria) with a level of gray or brightness, more or less in the middle of the possible range. After acquisition phase typical post-processing techniques try to realize an effective enhancement by using global approaches: histogram specification, histogram equalization and gamma correction to improve global contrast appearance [11, 60] only by stretching the global distribution of the intensity. More adaptive criterions are needed to overcome such drawback. The exposure correction technique [13] described in this Section is designed essentially for mobile sensors applications. This new element, present in newest mobile devices, is particularly harmed by "backlight" when the user utilizes a mobile device for video phoning. The detection of skin characteristics in captured images allows selection and properly enhancement and/or tracking of regions of interest (e.g., faces). If no skin is present in the scene the algorithm switches automatically to other features (such as contrast and focus or indoor/outdoor) tracking for visually relevant regions. This implementation differs from the algorithm described in [27] because the whole processing can also be performed directly on Bayer pattern images [25], and simpler statistical measures were used to identify *information carrying* regions. The algorithm is defined as follows:

1. Luminance extraction. If the algorithm is applied on Bayer data, in place of the three full color planes, a sub-sampled (quarter size) approximated input data (Fig.(**4.8**)) is used.
2. Using a suitable features extraction technique the algorithm fixes a value to each region. This operation permits to seek visually relevant regions (for

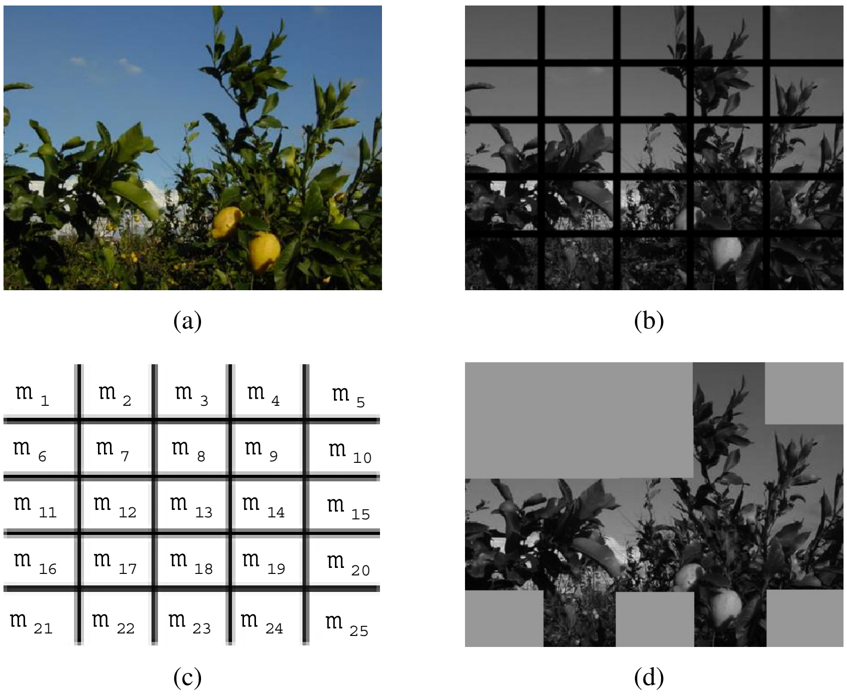


Figure 4.5 : Features extraction pipeline (for focus and contrast with $N = 25$). Visual relevance of each luminance block **(b)** of the input image **(a)** is based on relevance measures **(c)** able to obtain a list of relevant blocks **(d)**.

contrast and focus, or indoor/outdoor, the regions are block based, for skin recognition the regions are associated to each pixel).

3. Once the 'visually important' pixels are identified (e.g., the pixels belonging to skin features) a global tone correction technique is applied using as main parameter the mean gray level of the relevant regions.

4.3.1 Feature Extraction

As aforementioned, in the following we will briefly describe three techniques able to extract useful information to correct the image exposure.

4.3.1.1 Focus and Contrast

To be able to identify regions of the image that contain more information, the luminance plane is subdivided in N blocks of equal dimensions. For each block, statistical measures of "contrast" and "focus" are computed. Contrast refers to the range of tones present in the image. A high contrast leads to a higher number of perceptual significance regions inside a block. Focus characterizes the sharpness or edgeness of the block and is useful in identifying regions where high frequency components (i.e., details) are present.

The contrast measure is computed by simply building a histogram H_x for each block x of the N blocks and then calculating its deviation (4.3) from the mean value (4.4):

$$C_x = \frac{\sum_{i=0}^{255} |i - M_x| \cdot H_x[i]}{\sum_{i=0}^{255} H_x[i]} \tag{4.3}$$

where M is the mean value:

$$M_x = \frac{\sum_{i=0}^{255} i \cdot H_x[i]}{\sum_{i=0}^{255} H_x[i]} \tag{4.4}$$

A high deviation value denotes good contrast and vice versa.

The focus measure is computed by convolving each block with a simple 3x3 Laplacian filter. In order to discard irrelevant high frequency pixels (mostly noise), the outputs of the convolution at each pixel are thresholded. The mean focus value of each block is computed as:

$$F_x = \frac{\sum_{i=1}^{M} thresh[lapl(i), Noise]}{M} \tag{4.5}$$

where M is the number of pixels and the $thresh()$ operator discards values lower than a fixed threshold $Noise$. Once the values F_x and C_x are computed for all blocks, relevant regions will be classified using a linear combination of both values. Features extraction pipeline is illustrated in Fig.(**4.5**).

Therefore it is assumed that well focused or high-contrast blocks are more relevant compared to the others.

- Contrast refers to the range of tones present in the image. A high contrast leads to a higher number of perceptual significance regions inside a block.
- Focus characterizes the sharpness or edginess of the block and is useful in identifying regions where high frequency components (i.e., details) are present.

If the aforementioned measures were simply computed on highly underexposed images, then regions having better exposure would always have higher contrast and edginess compared to those that are obscured.

In order to perform a visual analysis revealing the most important features regardless to lighting conditions, a new "visibility image" is constructed by pushing the mean gray level of the input green Bayer pattern plane (or the luminance channel for color images) to 128. The push operation is performed using the same function that is used to adjust the exposure level and it will be described later. Furthermore to remove irrelevant peaks, the histogram is slightly smoothed by replacing each entry with its mean in a ray 2 neighborhood. Thus, the original histogram entry is replaced with the gray-level $\tilde{H}_x[i]$:

$$\tilde{H}_x[i] = \frac{(H_x[i-2] + H_x[i-1] + H_x[i] + H_x[i+1] + H_x[i+2])}{5} \tag{4.6}$$

Once the values F_x and C_x, for each block x of the N blocks, are computed, relevant regions will be classified using a linear combination of both values.

4.3.1.2 Skin Recognition

As before a “visibility image” obtained forcing the mean gray level of the luminance channel to be about 128 is built. Most existing methods for skin color detection usually threshold some sort of measure of the likelihood of skin colors for each pixel and treat them independently. Human skin colors form a special category of colors, distinctive from the colors of the most other natural objects. It has been found that human skin colors are clustered in various color spaces ([122], [151]). The skin color variations between people are mostly due to intensity differences. These variations can therefore be reduced by using chrominance components only. Yang *et al.* [146] have demonstrated that the distribution of human skin colors can be represented by a two-dimensional Gaussian function on the chrominance plane. The center of this distribution is determined by the mean vector $\vec{\mu}$ and its shape is determined by the covariance matrix Σ; both values can be estimated from an appropriate training data set. The conditional probability $p\left(\vec{x}|s\right)$ of a block belonging to the skin color class, given its chrominance vector $\vec{x}$ is then represented by:

$$p\left(\vec{x}|s\right) = \frac{1}{2\pi}\left|\Sigma\right|^{-\frac{1}{2}} \exp\left\{\frac{-\left[d(\vec{x})\right]^2}{2}\right\} \tag{4.7}$$

where $d(\vec{x})$ is the so-called Mahalanobis distance from the vector $\vec{x}$ to the mean vector $\vec{\mu}$ and defined as:

$$\left[d(\vec{x})\right]^2 = (\vec{x}-\vec{\mu})'\Sigma^{-1}(\vec{x}-\vec{\mu}) \tag{4.8}$$

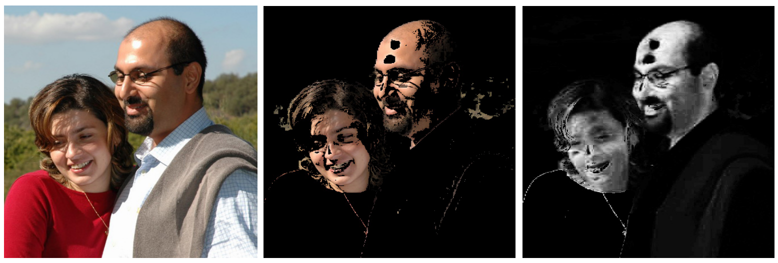


Figure 4.6 : Skin recognition examples on RGB images: **(a)** original images compressed in JPEG format; **(b)** simplest threshold method output; and **(c)** probabilistic threshold output.

The value $d(\vec{x})$ determines the probability that a given block belongs to the skin color class. The larger the distance $d(\vec{x})$, the lower the probability that the block belongs to the skin color class s.

Due to the large quantity of color spaces, distance measures, and two-dimensional distributions, many skin recognition algorithms can be used. The skin color algorithm can be applied in different ways (as shown in Fig.(**4.6**)):

1. By using the input YCbCr image and the conditional probability (4.7), each

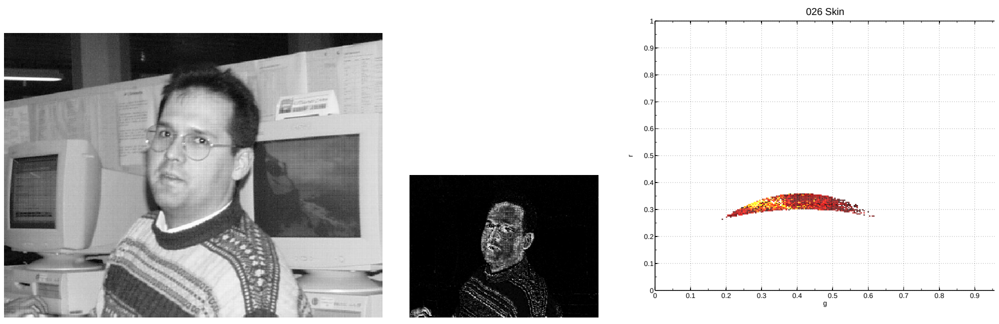


Figure 4.7 : Skin recognition examples on Bayer pattern image: **(a)** original image in Bayer data; **(b)** recognized skin with probabilistic approach; and **(c)** threshold skin values on $r-g$ bidirectional histogram (*skinlocus*).

pixel is classified as belonging to a skin region or not. Then a new image with normalized grayscale values is derived, where skin areas are properly highlighted (Fig.(**4.6(c)**)). The higher the gray value the bigger the probability to compute a reliable identification.

2. By processing an input RGB image, a 2D chrominance distribution histogram (r, g) is computed, where $r=R/(R+G+B)$ and $g=G/(R+G+B)$. Chrominance values representing skin are clustered in a specific area of the (r,g) plane, called "*skin locus*" (Fig.(**4.7(c)**)), as defined in [134]. Pixels having a chrominance value belonging to the skin locus will be selected to correct exposure.

3. For Bayer data, the skin recognition algorithm works on the RGB image created by sub-sampling the original picture, as described in Fig.(**4.8**).

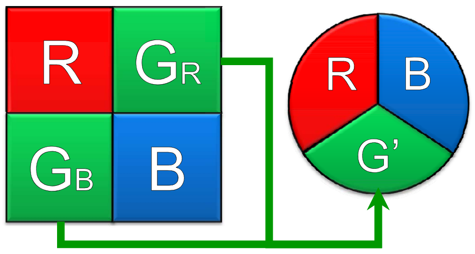


Figure 4.8 : Bayer data sub-sampling generation, where $G' = \frac{G_R+G_B}{2}$.

4.3.1.3 Indoor/Outdoor

Another technique reported in [132] uses a two-stage classifier exploiting two features: colors and textures. Then through a Support Vector Machine (SVM) classifier [30], the algorithm independently classifies image blocks according to features using wavelets coefficients. The classified blocks are then used by the second stage SVM classifier to determine a final indoor/outdoor decision.

To be able to extract color information the image is by first converted into the LST color space. This color space is based on Otha Color Space [113] except for the normalization factors:

$$\begin{array}{rcl} L & = & \frac{k}{\sqrt{3}}(R+G+B) \\ S & = & \frac{k}{\sqrt{2}}(R-B) \\ T & = & \frac{k}{\sqrt{6}}(R-2G+B) \end{array} \tag{4.9}$$

where $k = 225/\max(R,G,B)$. Thus L represents the luminance channel, S and T the chrominances one. This color space is used to de-correlate color channels in the original RGB image. Using a 16 bin histogram for each channel a 48 dimensions feature vector is computed for each block [132] and classified independently.

The texture features are extracted, from the L channel, through a two level wavelets decomposition, using Daubechies' 4-taps filters [40].

Let $c_2, c_3, c_4, c_5, c_6, c_7$ and c_8 represents the subband coefficients of the two level wavelets decomposition, as described in Fig.(**4.9**). The texture features are obtained by first filtering the low-frequency coefficients c_5 using the Laplacian filter and then by taking into consideration the subband energy:

$$e_x = \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}|c_x(i,j)|^2, \quad x = 2,3,...8 \tag{4.10}$$

where M and N are the image dimensions.

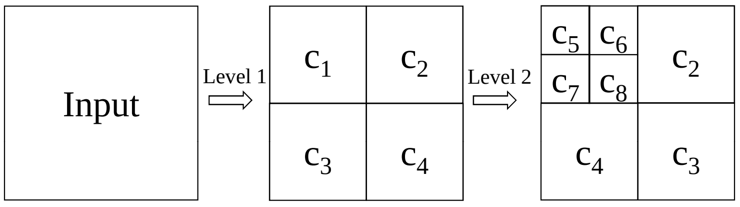


Figure 4.9 : Coefficients of the two-level wavelets decomposition.

For classifying each block, using calculated features, the SVM is applied with a radial basis kernel function, as described by Efimov *et al.* in [43].

By taking into account the information, collected on block basis, a similar approach as described in 4.3.1.1 can be achieved. In this case the correction is adaptively performed considering the number of indoor versus the number of outdoor blocks. In particular:

1. if the number of indoor blocks is larger than the number of outdoor blocks then the scene has probably been taken into a room and the exposure must be correct in function of such blocks.
2. if the number of outdoor blocks is larger than the number of indoor blocks then the scene has probably been taken outside the exposure must be correct in function of such blocks.
3. finally if the number of blocks of each class is equal or more or less equal then the exposure must be corrected taking into consideration the global statistics.

4.3.2 Exposure Correction

Once the visually relevant regions are identified, the exposure correction is carried out using the mean gray value of those regions as reference point. A simulated camera response curve is used for this purpose. This function can be expressed by using a simple parametric closed form representation:

$$f(q) = \frac{255}{\left(1+e^{-(Aq)}\right)^{C}} \tag{4.11}$$

where q represents the 'light' quantity and the final pixel value is obtained by means of the parameters A, and C used to control the shape of the curve. q is supposed to be expressed in 2-based logarithmic unit (usually referred as "stops"). These parameters could be estimated, depending on the specific image acquisition device or chosen experimentally, as better specified below The offset from the ideal exposure is computed using the f curve and the average gray level of visually relevant regions *avg*, as:

$$\Delta = f^{-1}(Trg) - f^{-1}(avg) \tag{4.12}$$

where *Trg* is the desired target gray level. *Trg* should be around 128 but its value could be slightly changed especially when dealing with Bayer Pattern data where some post processing is often applied.

The luminance value *Y(x,y)* of a pixel *(x,y)* is modified as follows:

$$Y'(x,y) = f(f^{-1}(Y(x,y)) + \Delta) \tag{4.13}$$

Note that all pixels are corrected. Basically all processing steps could be implemented through a LUT (Lookup Table) transform.

4.3.3 Results

The described technique has been tested using a large database of images acquired at different resolutions, with different acquisition devices, both in Bayer and RGB format. In the Bayer case the algorithm was inserted in a real-time framework, using a CMOS-VGA sensor on "*STV6500 - E01*" Evaluation Kit equipped with "*502 VGA sensor*" [137]. Examples of skin detection by using

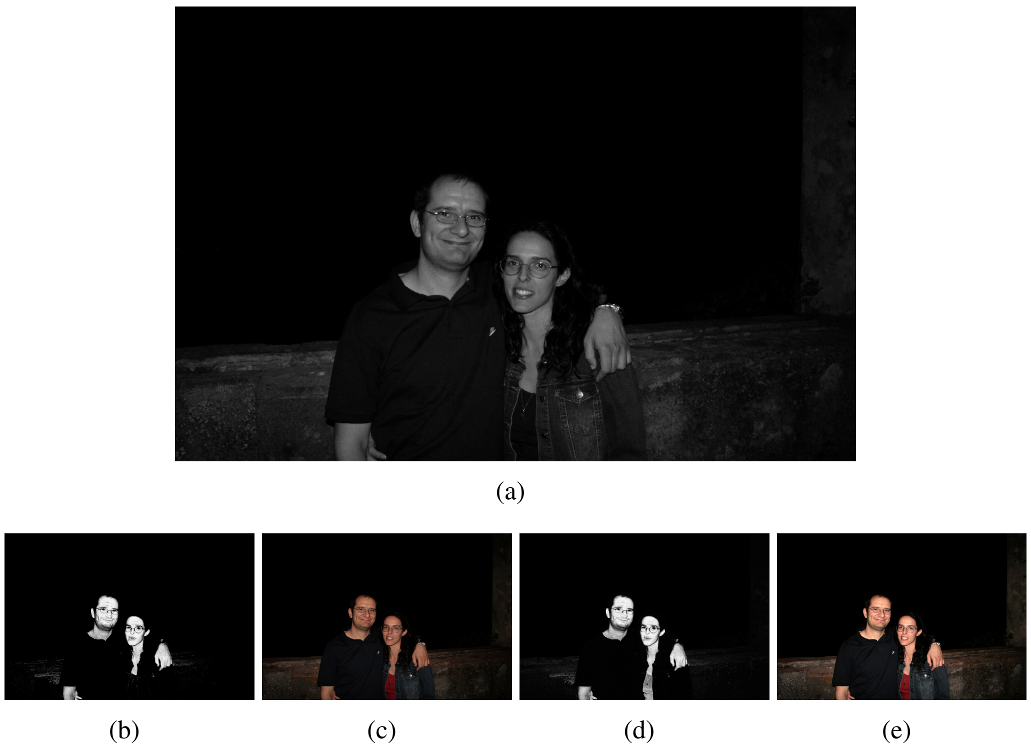
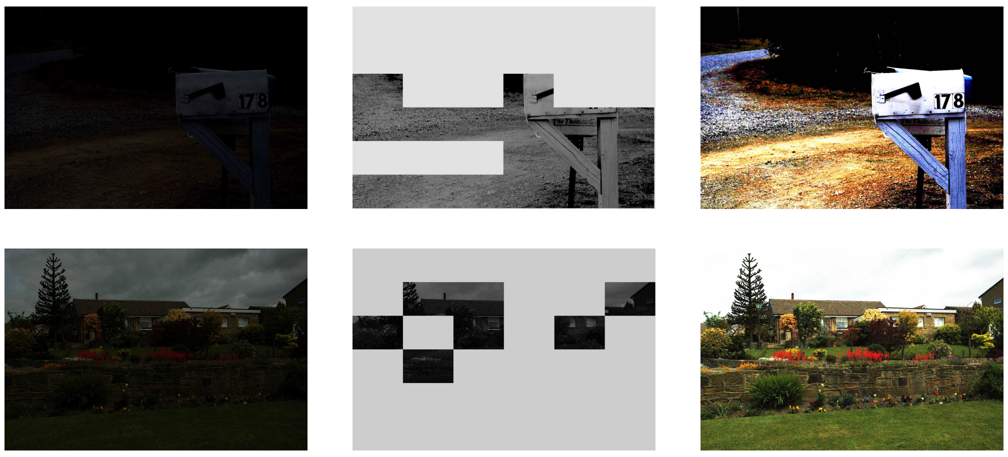


Figure 4.10 : Exposure Correction results by real-time and post processing: **(a)** original Bayer input image; **(b)** Bayer skin detected in real-time; **(c)** color interpolated image from Bayer input; **(d)** RGB skin detected in post processing; **(e)** exposure corrected image obtained from RGB image.



(a) (b) (c)

Figure 4.11 : Exposure correction results by post processing: **(a)** original color input image; **(b)** contrast and focus visually significant blocks detected; **(c)** exposure corrected image obtained from RGB image.

real time processing are reported in Fig.(**4.10**). In the RGB case the algorithm could be implemented as post-processing step. Examples of contrast/focus, indoor/outdoor and skin exposure correction are respectively shown in Fig.(**4.11**), Fig.(**4.12**) and Fig.(**4.13**). Results show how the features analysis capability of the proposed algorithm permits contrast enhancement taking into account some strong peculiarity of the input image. Major details and experiments can be found in [13].

4.4 Conclusion

The problem of the proper exposure settings for image acquisition is of course strictly related with the dynamic range of the real scene. In many cases some useful insights can be achieved by implementing ad-hoc metering strategies. Alternatively it is possible to apply some tone correction methods that enhance the overall contrast of the most salient regions of the picture. The limited dynamic

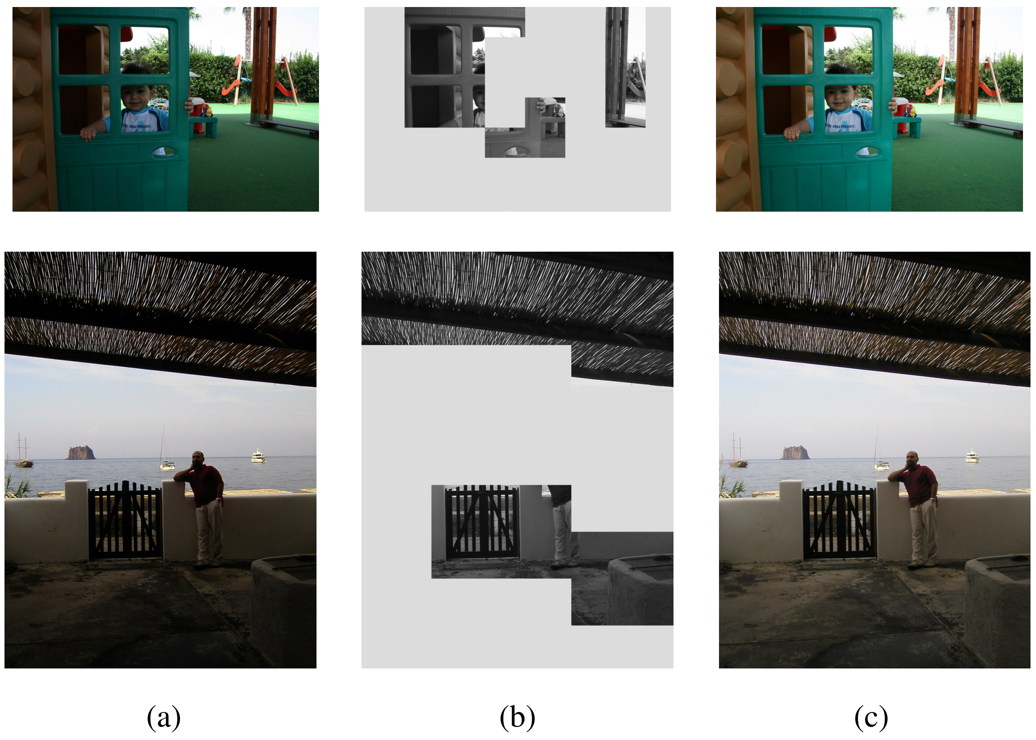


Figure 4.12 : Exposure correction results by post processing: **(a)** original color input image; **(b)** indoor blocks detected; **(c)** exposure corrected image obtained from RGB image.

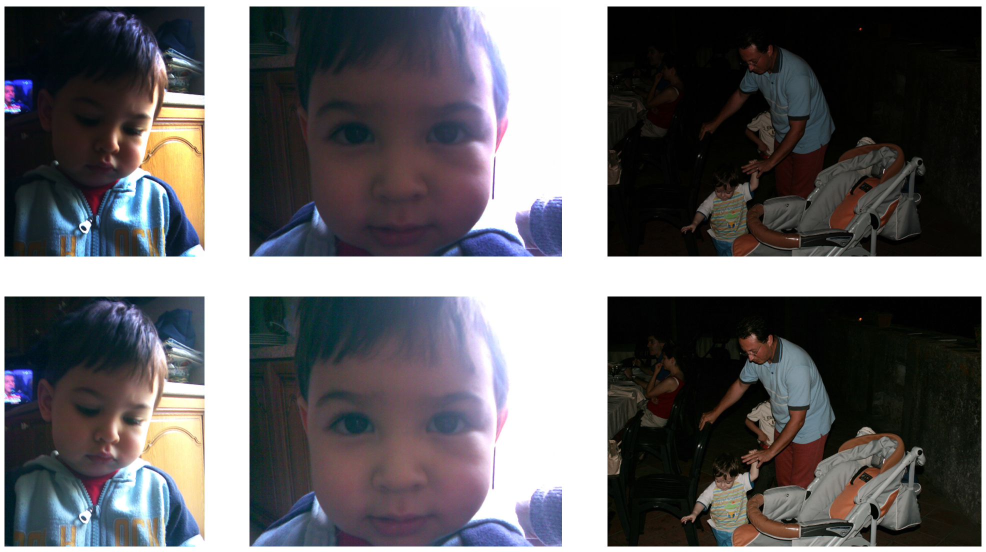


Figure 4.13 : Exposure correction results: in the first row the original images **(a)** and **(b)** acquired by Nokia 6125, 1.3Mpixels CMOS sensor; **(c)** picture acquired with CMOS sensor 10 Mpixels, Canon 400D camera; in the second row the corrected output.

range of the imaging sensors doesn't allow to recover the dynamic of the real world; in that case only by using ”bracketing” and acquiring several pictures of the same scene with different exposure timing a final good rendering can be realized.

In this work we have presented a brief review of automatic digital exposure correction methods trying to report the specific peculiarities of each solution. Just for completeness we report that recently, Raskar *et al.* [124] have proposed a novel strategy devoted to ”flutter” the camera's shutter open and closed during the chosen exposure time with a binary pseudo-random sequence. In this way high-frequency spatial details can be recovered especially when movements with constant speed are present. In particular a robust deconvolution process is achieved just considering the so-called coded-exposure that makes the problem well-posed. We think that the Raskar's technique could be used also in multi picture acquisition just to limit the overall number of images needed to reconstruct a reliable HDR map.

Part III

Forensics Image Processing

5. Forgery Detection

5.1 Introduction

Forgery is a subjective word. An image can become a forgery based upon the context in which it is used. For instance, an image altered for fun by someone who has taken a bad photo, but has been altered to improve its appearance, cannot be considered a forgery even though it has been altered from its original capture. The other side of forgery are those who perpetuate a forgery to gain payment and prestige. To achieve such purpose, they create an image in which they dupe the recipient into believing the image is real. There are different kinds of criminal forgeries:

- Images created through computer graphics tools (like 3D rendering) which looks like real but are completely virtual.
- Creating an image by altering its content, is another method. Duping the recipient into believing that the objects in an image are something else from what they really are (e.g., only by altering chrominances values). The image itself is not altered, and if examined will be proven as so (see example in Fig.5.1 from [48]).
- Objects are removed or added, for example, a person can be added or removed. The easiest way is to cut an object from one image and insert it into another image. By manipulating the content of an image the message can drastically change its meaning.

Altering images is not new - it has been around since the early days of photography. The first known forgery, see Figure5.2, was created only a few years after

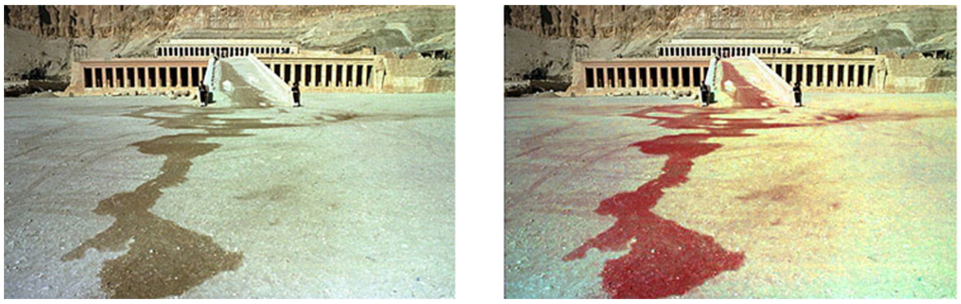


Figure 5.1 : After 58 tourists were killed in a terrorist attack (1997) at Hatshepsut's temple in Luxor Egypt, the Swiss tabloid Blick digitally altered a puddle of water (picture on the top) to appear as blood flowing from the temple (figure on the bottom).

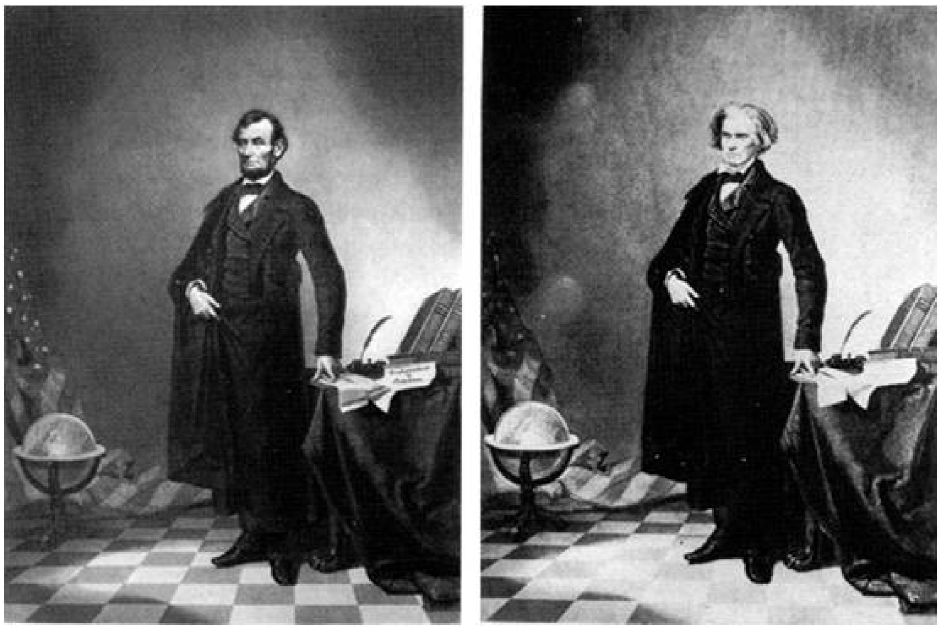


Figure 5.2 : circa 1860: This nearly iconic portrait of U.S. president Abraham Lincoln is a composite of Lincoln's head and the southern politicianJohn Calhoun's body.

the invention of the modern photography (circa 1826). The concepts have moved into the digital world by virtue of digital cameras and the availability of digital image editing software. The ease of use of photoshopping[1], which does not require any special skills, makes image manipulation easy to achieve. A good introduction to digital image forgery is given by Baron [12]. The book provides an excellent overview of the topic and describes some detailed examples. Furthermore, it illustrates methods and popular techniques in different contexts, from the historical forgeries until forensics aspects. The inverse problem of forgery detection is, on the other hand, a big issue. Several techniques take into account multiple aspect of image properties to achieve such purpose [47]. One of the promising approach, among others, considers the possibility to exploit the statistical distribution of DCT coefficients in order to reveal the irregularities due to the presence of a superimposed signal over the original one (e.g., copy and paste).

In this chapter we analyze in details the performances of three existing approaches evaluating their effectiveness by making use of different input datasets with respect to resolution size, compression ratio and just considering different kind of forgeries. Starting from a Bayesian approach to identify if a block is doctored or not, the authors of [69] use a support vector machine to classify the two classes of blocks into the forged JPEG image. In [149], the authors describe a passive approach to detect digital forgeries by checking inconsistencies of blocking artifact. Given a digital image, they find a blocking artifact measure based on an estimated quantization table using the power spectrum of the DCT

[1]Adobe Photoshop is a popular tool that can digitally enhance images. Images that have been modified using Photoshop or similar drawing tools (e.g., Gimp, Corel Paint, MS Paint, etc.) are described as being ”photoshopped” or ”shopped”. The quality of the shopped images depends on both the tool and the artist.

coefficient histogram. The approach in [46] explicitly detects if part of an image was compressed at a lower quality than the saved JPEG quality of the entire image. Such a region is detected by simply re-saving the image at a multitude of JPEG qualities and detecting spatially localized local minima in the difference between the image and its JPEG compressed counterpart.

All previous studies does not consider a sufficient number of effective cases including for examples different image resolution, compression size and forgery anomalies. Also the DCT basis to be evaluated for forgery detection should be selected in a proper way. In this chapter we start to build a systematic way to analyze the problem of forgery detection into DCT domain pointing out both strength and weak point of the existing solutions. Also some post processing strategies can be devised to properly mask the DCT anomalies.

The rest of the chapter is organized as follows. In section 5.2 the DCT quantization is described. In section 5.3 a deep description of the aforementioned techniques is presented. The experimental part is described in section 5.4 taking into consideration a large database which aim is to stress the described techniques. Finally the conclusion are given drawing possible improvements.

5.2 DCT Quantization

The DCT codec-engines typically apply a quantization step in the transform domain just considering non-overlapping blocks of the input data. Such quantization is usually achieved by a quantization table useful to differentiate the levels of quantization adapting its behavior to each DCT basis. The JPEG standard has fixed the quantization tables and just varying by a single multiplicative factor, these tables, different compression ratios (and of course different quality levels) are obtained [60, 118]. As the tables are included into the image file, they are also customizable [14, 22, 29]. In such a way the commercial codec solutions exploit proprietary tables. One among other is the largely used Photoshop which take into consideration thirteen levels of quality (from 0 to 12) and in the "Save as Web" configuration hundred quality levels (from 1 to 100). For each basis the same quantization step q_i (for $i = \{1, .., 63\}$) is applied over the entire set of non-overlapping blocks just producing a set of values that can be simply clustered by their values (integers) that constitute a clear periodic signal with period equal to q_i.

Once a first quantization has been performed a second quantization will introduces periodic anomalies into the signal (See Fig.5.3). The anomalies of such periodicity can be analyzed to discover possible forgeries. Mathematically the meaning of the double quantization is:

$$Q_{1,2}(u_{1,i}) = \left[\left[\frac{u_{1,i}}{q_{1,i}} \right] \frac{q_{1,i}}{q_{2,i}} \right] \tag{5.1}$$

where $q_{1,i}$ and $q_{2,i}$ are two different quantization factors and $u_{1,i}$ is the DCT coefficient at position i of the current block.

5.3 State of the Art Analysis

In the following we will describe the three aforementioned techniques. In particular we will show typical example of unrecognized forgeries cases, explaining the reason of such failures.

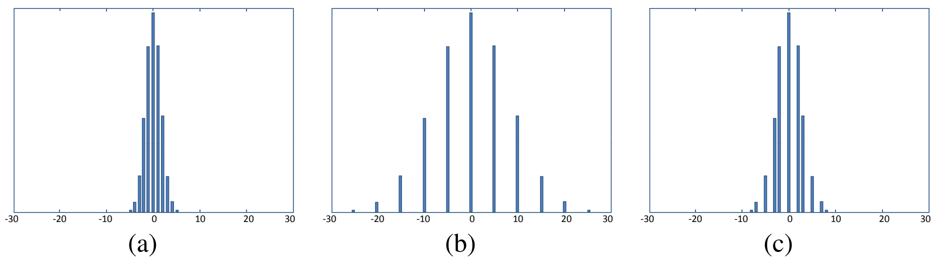
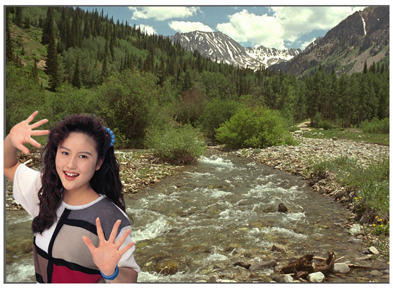


Figure 5.3 : Double quantization artifacts: (a) the distribution of singly quantized coefficients with q1 = 5; (b) the distribution of these coefficients de-quantized; (c) the distribution of doubly quantized coefficients with q1 = 5 followed by q2 = 3 (note the empty bins in this distribution);

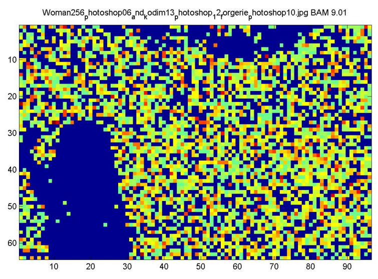
Quantization tables approximation

The approach of Ye et al. [149] consists of three main steps: collection of DCT statistics, analysis of statistics for quantization tables estimation, and assessment of DCT blocks errors with respect to the estimated quantization tables. The performances of such technique are strictly related with the amount of forged blocks in comparison with the total number of blocks. In other words the sensibility of the corresponding forgery detector is very high only at lower resolution size; at higher resolution the related performances degrades abruptly even in presence of an extended connected region (e.g., faces). Furthermore as the technique performs an estimation of the quantization tables to identify anomalies from the first quantization to the second one, the algorithm is very subject to the level of quality and number of compression.

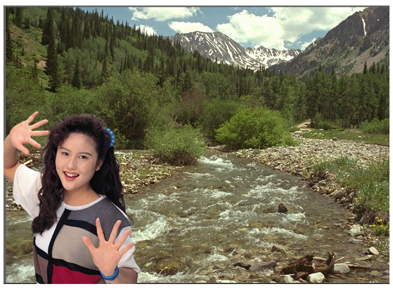
As example of failure we show the forged image obtained from uncompressed Kodak image [51] ("River" landscape) with overimpressed "Woman" (from JPEG Standard uncompressed test set) previously resampled from original size to 256×320. Before the "copy and paste" the two images "Woman" and "River" have been saved through photoshop respectively with level 6 and 12. The final photoshopped image has been saved through all the possible photoshop levels from 0 to 12. As result we show the two examples in Fig.5.4: in the first row (Fig.5.4(a)) the forged image saved with level 10 and the resulting error map (Fig.5.4(b)) which enhance several errors on blocks with high frequencies and less errors in low frequencies blocks (the woman and the sky). The second example of Fig.5.4(d) shows the resulting error map from the same image Fig.5.4(c) with a saved quality factor of 12 (maximum). In this final case the method was unable to find any kind of forgery and furthermore the level of estimated error is very low.



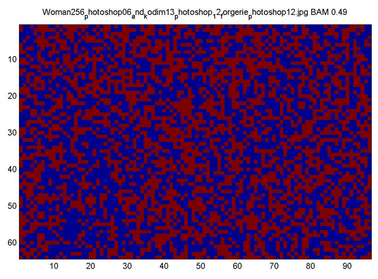
(a)



(b)



(c)



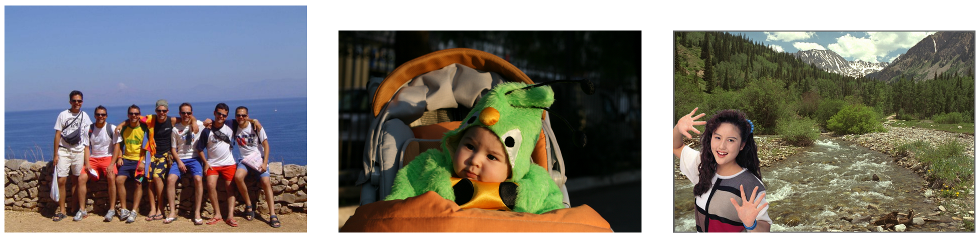
(d)

Figure 5.4 : Error maps obtained through [149] of a doctored image saved with different quality factor: (a) Saved doctored image with quality factor 10, (c) the same image saved with quality factor 12. (b) and (d) resulting blocks error maps.

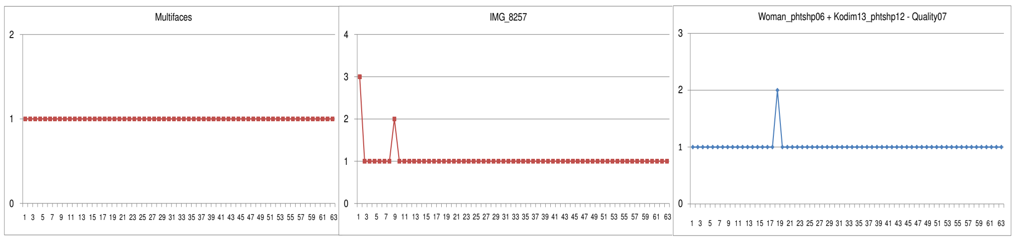
Bayesian Approach of Detecting Doctored Blocks

The four advantages possessed by the solution of He et al. [69], namely automatic doctored part determination, resistant to different kinds of forgery techniques in the doctored part, ability to work without full decompression, and fast detection speed, make the algorithm very attractive. But the initial quantization factor q_1 and the second one q_2 must be known in order to estimate the periodic function $n(k)$ needed to estimate the periodicity of histograms of double quantized signals. Actually q_2 can be dumped from the JPEG image. Unfortunately, q_1 is lost after the first decompression and hence has to be estimated. From our purpose, as the initial quantization factor is unknown, it can be only estimated from the image under analysis. Also for this reason we have designed a reference dataset where the original input image can be compressed by using different quantization parameters (i.e., different quantization tables at different quality levels).

The main issue of such approach is the correct estimation of $n(k)$ for such purpose we show some examples of periodicity estimation. The authors correctly assert that if the periodicity is $n(k) = 1$ it is impossible to estimate the forgery, otherwise if the periodicity is greater than 1 a forgery is likely to be present into the image. In Fig.5.5 we have considered three typical cases: Fig.5.5(a) is a doctored image where all the subjects into the scene have the same face, the image has been compressed through a medium quality level; Fig.5.5(b) shows an image without any kind of forgeries; Fig.5.5(c) is the same image as above with quality factor 7. As it is clearly visible the periodicity estimated Fig.5.5(d) for the image Fig.5.5(a) is uniformly equal to 1 for all the 63 coefficients statistics of the DCT (the DC has been excluded due to the nature of the histogram distribu-



(a) (b) (c)



(d) (e) (f)

Figure 5.5 : Periodicity Estimation through He et al. [69] method. (a) Original doctored image with low compression quality; (b) Original undoctored image; (c) Doctored image saved with photoshop quality 7. (d), (e) and (f) Periodicity estimation from the 63 DCT coefficients histograms.

tion). This means that the image is not affected by forgeries. On the other hand the estimated periodicity of the image Fig.5.5(b) depicts the presence of artifacts but the original images has been acquired directly from a Digital Camera and have not been altered. Finally we show the periodicity of the above mentioned test image; in this case the periodicity Fig.5.5(d) has been altered only for the coefficient 19 of the DCT basis, furthermore as there is only a small periodicity deviation the forgeries can not be estimated.

Multiple-Differences

Farid et al. [46] presents a technique based on multiple differences between original compressed JPEG and successive re-compressed version of the original image, with increasing compression factors. The disadvantage of this approach is that it is only effective when the tampered region is of lower quality than the image into which it was inserted. The advantage of this approach is that it is effective on low quality images and can detect relatively small regions that have been altered. In Fig.5.6 we show the error maps estimate from the original doctored image ”Woman” and ”river” with compression quality 11 and 12 respectively. The final doctored image has been saved with quality factor 12. The errors maps does not show significant ghost in any of the differences, in particular the sky and the woman present similar errors level and can not be estimated as clearly forged areas. The authors assert that the forgery is only effective when the tampered region is of lower quality than the image, which is the case of the example in Fig.5.6.

As shown in the three cases above, the reliability of the current solution is not so robust with respect to various aspects of image forgeries pipeline. A more sophisticated and extensive experimental assessment is needed for each forthcoming solution. In our opinion the relative ratio between q_1 and q_2 is fundamental to drive any existing forgery detection strategy into DCT domain. Also the relative size of the forged patch with respect to input resolution size should be taken into account. Finally all the methods that try to estimate the original quantization table block by block should be tuned just considering the content of each block in terms of local energy (e.g., textured vs. flat areas).

5.4 Dataset and Discussion

To assess the effectiveness of any forgery detection technique, a suitable dataset of examples should be used for evaluation. In this chapter we are interested in

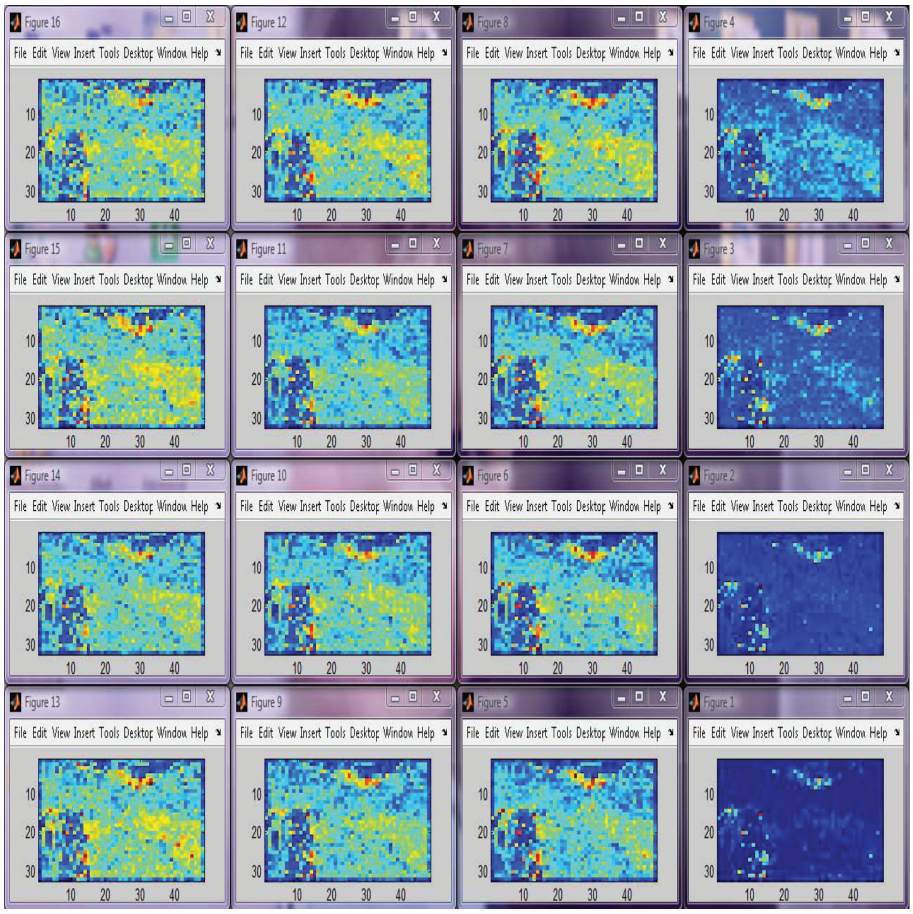


Figure 5.6 : Multiple differences maps from original ”woman+river” doctored image, in this case the woman was compressed with quality 11 and the river with quality 12. The doctored image was save with quality 12. The maps are represented from bottom right to top left in a decreasing level quality factor.

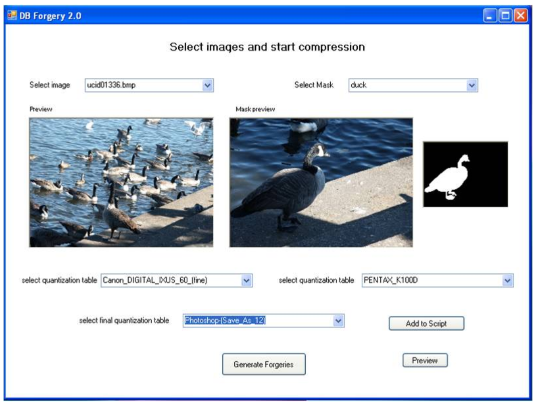
Figure 5.7 : Woman (JPEG Standard test image) and Lena masks, used to generate forgeries.

providing a reference dataset for the forgery detection to be used for performance assessment just including various critical (and in some cases typical) condition. It is also important to consider, into the dataset, images without any kind of forgeries to avoid false positives. The initial dataset contains the standard dataset from Kodak images [51] and from UCID v.2 [28].

A first version of the dataset used the quantization tables furnished by Photoshop [68] that were used to generate various compression ratio. The reason was mainly based upon the fact that if some one has to introduce forgeries into an image he must use some imaging software to achieve such intent. In particular we have designed a series of Photoshop scripting that were used in an unsupervised way to generate real examples. Using manually generated masks, as described in Fig.5.7, it was possible to generate several kinds of forgeries, modifying both forgery image and compression factor, also in this case making use of the scripting mechanism provided by Photoshop (which is largely used in image forensics analysis). Using this methodology it has been possible to generate hundreds of tampered images, through all possible combinations of compression ratios contained in Photoshop. This work has allowed us to create the first dataset, called DBForgery 1.0 [81]. This first version includes:

- 60 Images (UCID and Kodak dataset) [28,51];
- 30 Forgeries;
- 13 Quantization Tables.

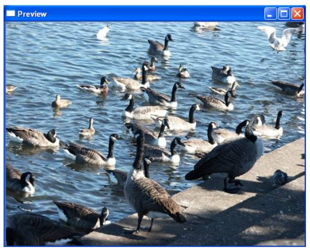
All possible combinations of Photoshop compression have been contemplated. Thus for each couple of background image and forgery image the number of possible combinations are 2197 (13x13x13 compression levels). This first version contains 30 forgeries combination (some are clearly evident and other are very



Select images and start compression

Select image: ucid01336.bmp
Select Mask: duck
select quantization table: Canon_DIGITAL_IXUS_60_(fine)
select quantization table: PENTAX_K100D
select final quantization table: Photoshop-(Save_As_12)

(a)



(b)

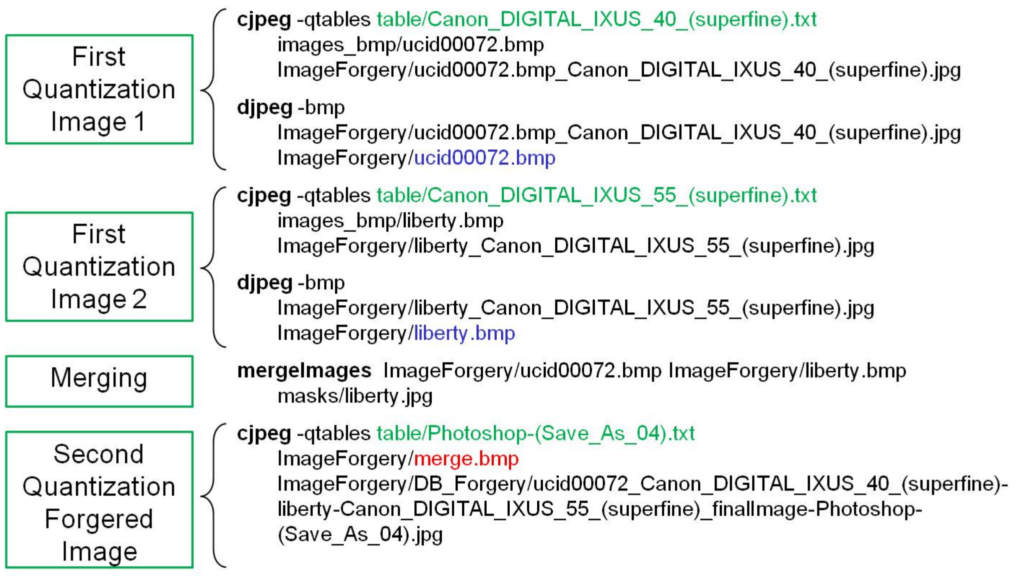
Figure 5.8 : User interface of DBForgery 2.0. (a) the main interface which permit to generate both forged images and scripts; (b) the preview of the forgery.

difficult to find). The initial dimension of the DB is around 39MBytes (before scripts) and expand to 9GBytes after forgeries generation. The final size of the dataset is of about 65 thousands forgeries.

Successively we have improved the database taking into account a bigger amount of quantization tables, the ability to store only the scripts for the subsequent regeneration of photomontages, and the independence from Photoshop [126]. The improved version of dataset contains:

- 156 Input quantization tables (iPhone, Canon, Fuji, Sony, Sigma, Pentax, Olympus, Nikon, Konica Minolta);
- 31 Output quantization tables (Photoshop, Photoshop for web, Infranview (Standard JPEG));
- 30 forgeries (same as DBForgery 1.0).

The possible combinations of all this quantization tables allow to generate 754416 triples. This means the final number of forged images is of about 2 millions. Thus to overcome the problem of an huge quantity of possible combination an user interface has been created. To be able to recreate the same dataset a scripting generator has been integrated into the application. Hence by sharing only the scripts, which contain all the needed information, the dataset can be recreated in the same way whenever it is necessary. The most important difference of DB-Forgery 2.0 respect to the previous version is the independence from Photoshop. Indeed, by taking into account the "cjpeg" and "djpeg" open source codes for jpeg compression/decompression coupled with a simple merging program (written in C ANSI standard), it has been possible to bypass the Photoshop interface.


First Quantization Image 1
First Quantization Image 2
Merging
Second Quantization Forgered Image

```
cjpeg -qtables table/Canon_DIGITAL_IXUS_40_(superfine).txt
    images_bmp/ucid00072.bmp
    ImageForgery/ucid00072.bmp_Canon_DIGITAL_IXUS_40_(superfine).jpg
djpeg -bmp
    ImageForgery/ucid00072.bmp_Canon_DIGITAL_IXUS_40_(superfine).jpg
    ImageForgery/ucid00072.bmp

cjpeg -qtables table/Canon_DIGITAL_IXUS_55_(superfine).txt
    images_bmp/liberty.bmp
    ImageForgery/liberty_Canon_DIGITAL_IXUS_55_(superfine).jpg
djpeg -bmp
    ImageForgery/liberty_Canon_DIGITAL_IXUS_55_(superfine).jpg
    ImageForgery/liberty.bmp

mergeImages  ImageForgery/ucid00072.bmp ImageForgery/liberty.bmp
    masks/liberty.jpg

cjpeg -qtables table/Photoshop-(Save_As_04).txt
    ImageForgery/merge.bmp
    ImageForgery/DB_Forgery/ucid00072_Canon_DIGITAL_IXUS_40_(superfine)-
    liberty-Canon_DIGITAL_IXUS_55_(superfine)_finalImage-Photoshop-
    (Save_As_04).jpg
```

Figure 5.9 : An example of script genreated by the U.I. containing all the information necessary to regenerate the forgery.

We plan to add to our dataset also a series of possible image alterations including among others: cropping, flip, rotation, rotation-scale, sharpening, Gaussian filtering, random bending, linear transformations, aspect ratio, scale changes, line removal, color reduction, etc. We hope to re-use in some sense, part of the available software already adopted in the field of benchmarking tool for still image watermarking algorithms (e.g., Stirmark, CheckMark,Optimark) [119, 120].

One possible strategy to improve robustness and reliability of forgery detection into DCT domain is related to the choice of the effective DCT basis to be considered in the process. Preliminary results have reported an improvement of about 10% in terms of forgery detection just implementing some heuristics devoted to filtering out low and high frequency coefficients just working only on mid-basis values.

5.5 Acknowledgments

The authors want to thanks the Interpol Crime Against Children Group of Lyon who have collaborate to redirect this work in a much more detailed analysis of realistic problems. Their contribution have enhanced the issues caused by theoretical approaches and have forced the direction of our project to a much more robust and helpful application.

5.6 Conclusions

The forgery detection is not a trivial task. The DCT domain, just considers specific peculiarities of the quantized basis. Unfortunately, the variability of the context especially in real cases includes many situations (e.g. resolution size, camera model, compression ratio, quantization parameters, number of forgeries, etc.) that are not fully exploited. In this chapter we have presented experimental evidence of weakness and strength points of the current solutions. We have also started to build a comprehensive dataset that can be used for a robust assessment and evaluation of any forthcoming technique in the field. At conference time a more detailed experimental framework together with a freely accessible dataset to be used for DCT-based forgery detection will be presented.

6. Chain of Evidence

6.1 Introduction

Nowadays digital cameras are certainly the most used devices to capture images. They are everywhere, including mobile phones, personal digital assistants (PDAs), and surveillance and home security systems. There is no doubt that the quality of the images obtained by digital cameras, regardless of the context in which they are used, has improved a lot since digital cameras early days. However, there are still a variety of problems which need to be tacked regarding the quality of the acquired images [13, 15, 24, 93].

The use of evidence from surveillance video cameras is expected to increase greatly. For instance, the NYC Surveillance Camera Project [2] is a project of volunteers which aim is to sign out every camera, public or private, which records people in public space in Manhattan. At the current stage the number of mapped surveillance camera is 2,397. Furthermore, the recording tools still using obsolete technologies (e.g., video tapes), and the quality of the output is usually very poor. Common problems of such videos are poor resolution, poor contrast due to under or over-exposure, corruption with noise, motion blur or poor focus and misalignment of rows from line jitter. To be able to find some invisible particulars, into such images, a processing enhancement is obviously required.

In this work we are mainly interested to contrast enhancement of input data. A common definition of contrast can be summarized as:"the standard variation of the image gray-levels or luminance". The histogram of the gray-levels is thus stretched and/or re-arranged just using the entire range of the discrete levels available, in the way that an enhancement of image contrast is achieved [60].

The forensics rules to produce evidences require a complete documentation of the processing steps, enabling the replication of the entire procedure. Literally the Chain of Evidence is :"The movement and location of physical evidence from the time it is obtained until the time it is presented in court". The House of Lords [1] has fixed some rules about "Digital Images as Evidence". More specifically, There are two elements to establish authenticity: to have an "audit trail" which records everything that happens to the image from capture to its presentation in court; and/or to have a technological solution which brands or "watermarks" the image at the time of capture and can subsequently show it is authentic. The FBI [3] , to facilitate the integration of imaging technologies and systems in the criminal justice system, has also provided definitions and recommendations for the capture, storage, processing, analysis, transmission, and output of images. The automation of enhancement techniques is thus quite difficult and needs to be carefully documented.

This work presents an automatic procedure, based on sub-regions analysis. Using a suitable features extraction technique the algorithm fixes a value to each region. This operation permits to seek visually relevant regions (e.g., contrast and focus). Once the "visually important" regions are identified a global tone correction technique is applied using as main parameter the mean gray levels of the relevant regions. The correction is then achieved through an automatically generated JavaScript, Adobe Photoshop compliant, which contains the estimated correction curve (lookup table). Once the script has been generated the contrast enhancement can be applied, in the same manner, at anytime.

6.2 Image Analysis

The approach is divided into three principal steps: features extraction, data analysis and correction. In this work we present the ”contrast” and ”focus” analyzer, which is used as general purpose to estimate exposure. This module can be expanded or substituted with other kind of feature extractors: skin and/or face, car plate, text, etc. .

6.2.1 Features Extraction

To be able to identify regions of the image that contain more information, the luminance plane of the input image is subdivided into N blocks of equal dimensions (in our experiments we employed N=64). For each block, statistical measures of “contrast” and “focus” are computed. Therefore it is assumed that well focused or high-contrast blocks are more relevant compared to the others. Contrast refers to the range of tones present in the image. A high contrast leads to a higher number of perceptual significance regions inside a block. Focus characterizes the sharpness or edginess of the block and is useful in identifying regions where high frequency components (i.e., details) are present.

If the aforementioned measures were simply computed on highly under-exposed images, then regions having better exposure would always have higher contrast and edginess compared to those that are obscured. In order to perform a visual analysis revealing the most important features regardless to lighting conditions, a new “visibility image” is constructed by pushing the mean gray level of the luminance channel to 128. The push operation is performed using the same function that is used to adjust the exposure level and it will be described later. In order to remove irrelevant peaks, the histogram is slightly smoothed by replacing each entry with its mean in a ray 2 neighborhood. Thus, the original histogram entry is replaced with the gray-level:

$$\tilde{I}[i] = \frac{(I[i-2]+I[i-1]+I[i]+I[i+1]+I[i+2])}{5} \tag{6.1}$$

The contrast measure C is then computed by simply building a histogram for each block and then calculating its deviation (6.2) from the mean value M (6.3). A high deviation value denotes good contrast and vice versa. Therefore the histogram deviation C is computed as:

$$C = \frac{\sum_{i=0}^{255} |i-M| \cdot \tilde{I}[i]}{\sum_{i=0}^{255} \tilde{I}[i]} \tag{6.2}$$

where M is the mean value :

$$M = \frac{\sum_{i=0}^{255} i \cdot \tilde{I}[i]}{\sum_{i=0}^{255} \tilde{I}[i]} \tag{6.3}$$

The focus measure F is computed by convolving each block with a simple 3×3 Laplacian filter. In order to discard irrelevant high frequency pixels (mostly noise), the outputs of the convolution at each pixel are thresholded. The mean focus value of each block is computed as:

$$F = \frac{\sum_{i=1}^{P} thresh[lapl(i), Noise]}{P} \tag{6.4}$$

where P is the number of pixels per blocks and the $thresh()$ operator discards values lower than a fixed threshold $Noise$. Once the values F and C are computed for all blocks, relevant regions will be classified using a linear combination of both values.

We have also improved the quality of the region detection by applying a coarse to fine approach. In this case the block are iteratively subdivided and analyzed to obtain finer statistical details. In this way it is possible to generate a more precise map of the regions of interest.

6.2.2 Features Analysis

Once the visually relevant regions are identified, the exposure correction is estimated using the mean gray value of those regions as reference point. A simulated camera response curve is used for this purpose, which gives an appraisal of how light values falling on the sensor become final pixel values. Thus it is a function

$$f(q) = I \tag{6.5}$$

where q represents the "light" quantity and I the final pixel value. This function can be expressed by using a simple parametric closed form representation:

$$f(q) = \frac{255}{\left(1 + e^{-(\alpha q)}\right)^{\gamma}} \tag{6.6}$$

where parameters α, and γ can be used to control the shape of the curve and q is supposed to be expressed in 2-based logarithmic unit (usually referred as "stops"). These parameters could be estimated, depending on the specific image acquisition device, using the techniques described in ([100]) or chosen experimentally. The offset from the ideal exposure is computed using the f curve and the average gray level of visually relevant regions avg, as:

$$\Delta = f^{-1}(Trg) - f^{-1}(avg) \tag{6.7}$$

where Trg is the desired target gray level. Trg should be a value around the middle of the histogram spectrum. The final luminance value Y' of a pixel (x, y)

is thus modified as follows:

$$Y'(x,y) = f(f^{-1}(Y(x,y)) + \Delta) \tag{6.8}$$

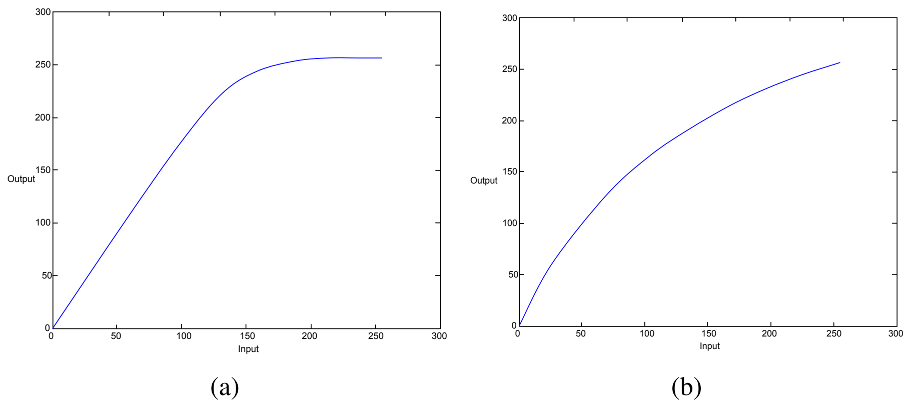


Figure 6.1 : LUTs derived from curves with (a) $\alpha = 7$ and $\gamma = 0.13$; (b) $\alpha = 0.85$ and $\gamma = 1$.

Note that all pixels are corrected. Basically the previous step is implemented as a LUT (Lookup Table) transform (Figure 6.1 shows two correction curves with different α, γ parameters).

6.2.3 Script Generation

Usually standard contrast enhancement approaches tend to represent pictures in a more pleasant aspect. In this particular case, which is much more oriented to forensics, the content is more important than the aspect. Two possible solutions can be taken into account to retrieve information from under-exposed pictures:

1. Correct the contrast block by block: in this case each block is processed in function of the medium luminance value (128) and due to its medium value a softer or an hard enhancement is achieved.

2. Correct the overall contrast: this mean the global contrast enhancement.

Using the scripting approach we are able to generate automatically a couple of images for both types of corrections.

To generate the JavaScript code we use a fixed template which already opens the image and splits the image into three channels (RGB). The photoshop document object uses the "activeLayer" to select the working plane. As the "adjustCurves" method, used to apply the contrast correction, allows using a maximum of fourteen points, the LUT is quantized using fourteen values. The resulting points are passed to the methods and then the script is associated to the image. As example we show the following code with 3 points:

```
app.activeDocument.activeLayer.adjustCurves( Array(
Array(35,80), Array(128,128), Array(230,190) ) );
```

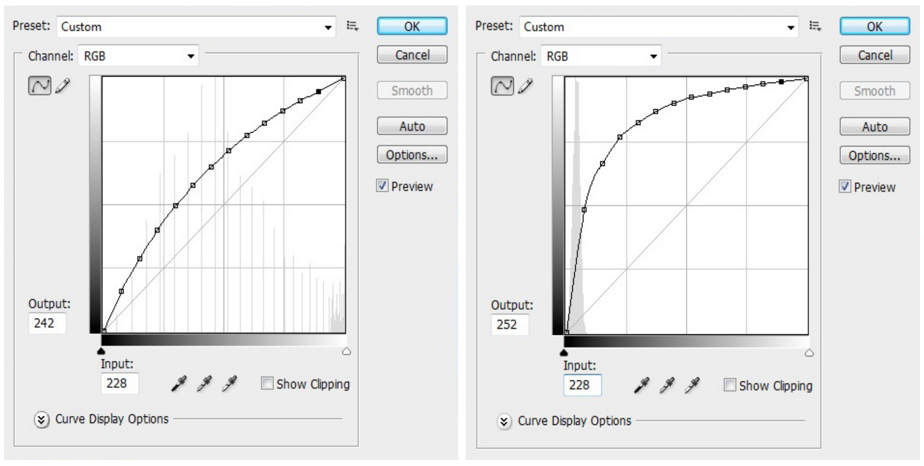
To achieve the correction only on the selected regions the Photoshop method "selection" is also used.

6.3 Experiments

To show the effectiveness of our methods a dataset of images underexposed and affected with backlight, has been taken into account. The images have been processed through the two solution: local correction and global correction.

As example of curves used to correct the contrast, we have selected two examples in Fig.6.2. In particular the first curve (Fig.6.2.a) has been created using the following Photoshop script code:

```
app.activeDocument.activeLayer.adjustCurves(
Array(Array(0,0),Array(19,38),Array(38,70),Array(57,98),
```


Preset: Custom
Channel: RGB
Output: 242
Input: 228
Preview ✓

Preset: Custom
Channel: RGB
Output: 252
Input: 228
Preview ✓

(a) (b)

Figure 6.2 : Examples of Photoshop curves used to modify the contrast both locally and globally.

```
Array(76,122),Array(95,144),Array(114,163),Array(133,179),
Array(152,194),Array(171,208),Array(190,220),Array(209,231),
Array(228,241),Array(255,255)));
```

This kind of curves have been used both for local correction and for global contrast enhancement. To achieve the global correction a statistical measure on blocks has been achieved, taking into account the most recurrent curve, that should be applied on blocks, as the global correction curve. In Fig.6.3[1] a group of young boys have been pictured in a place with several signs. At a first look it seems that only three persons are present in the scene and furthermore due to the low light quality the signs on the walls are not clear (Fig.6.3.a). If this was the scene of an investigation it would be necessary to enhance the contrast. Using a local contrast correction the details came out. In effect the Fig.6.3.b permit to better identify the scene and the number of persons present. In this case the local enhancement is clearly more powerful than the global one (Fig.6.3.c).

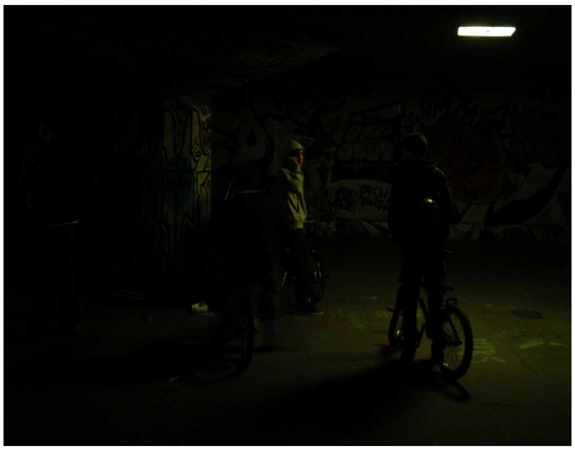
[1]All the pictures used for testing have been downloaded from internet and are full property of their owners.

In the following we will show different aspect of correction that must be taken into account when such algorithm is performed. In Fig.6.4 a similar result as previous example is shown. In particular the local enhancement, Fig.6.4.b, has permit to identify a person sitting in front of a window with backlight. The Fig.6.5 show a case of a better global enhancement than local one. In this case the local enhancement has introduce a heavy overshooting, causing loss of information.

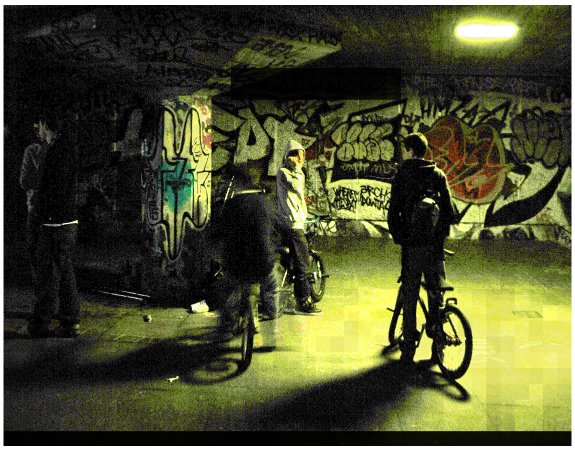
Finally two examples of highly compressed images (Fig 6.5 and Fig.6.6). In these cases the correction, both local and global, have been correctly achieved, but the quality of the images has allowed introduction of new information from the scene.

6.4 Conclusion

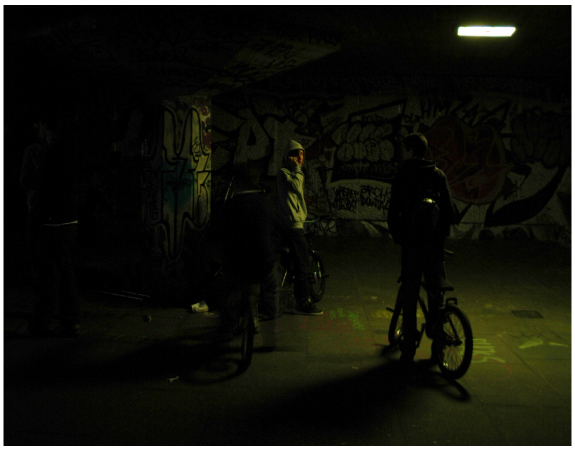
In this chapter an automatic procedure, based on sub-regions analysis, has been presented. Using a suitable features extraction technique the algorithm is able to fix a value to each region. Once the “visually important” regions have been identified a global tone correction technique is applied using as main parameter the mean gray levels of the relevant regions. The correction is then achieved through an automatically generated JavaScript, Adobe Photoshop compliant, which contains the estimated correction curve (lookup table). In this particular case, which is much more oriented to forensics, the content is more important than the aspect, thus the quality of the final results are more oriented to the content than to the image pleasance. The main aspect of this work is that once the script has been generated the contrast enhancement can be applied, in the same identical manner, at anytime.



(a)

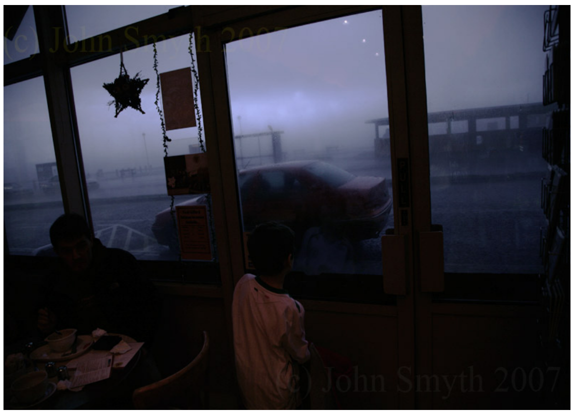


(b)

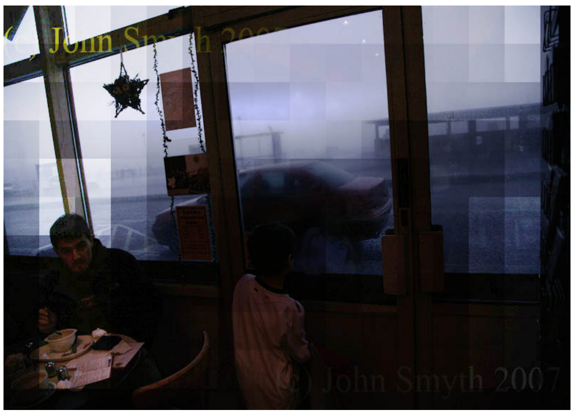


(c)

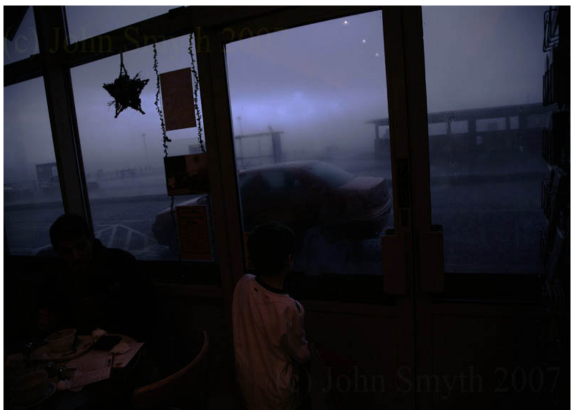
Figure 6.3 : Example of contrast enhancement from an underexposed image. (a) Input, (b) Local correction; (c) Global correction.



(a)

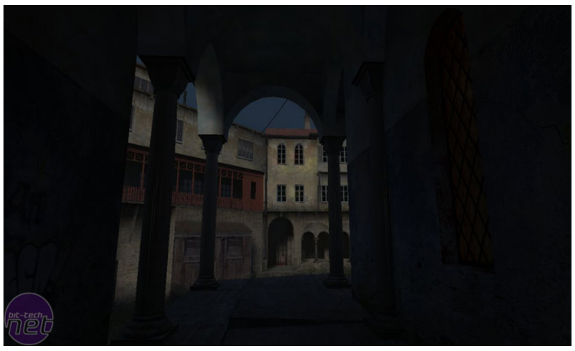


(b)

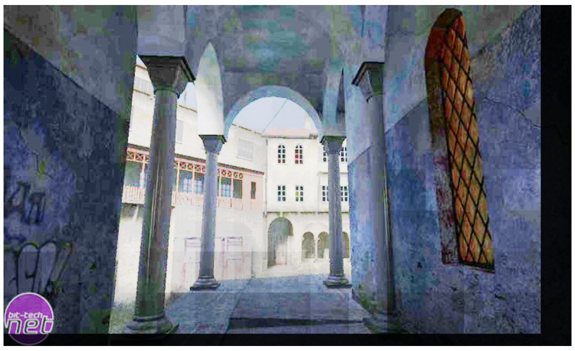


(c)

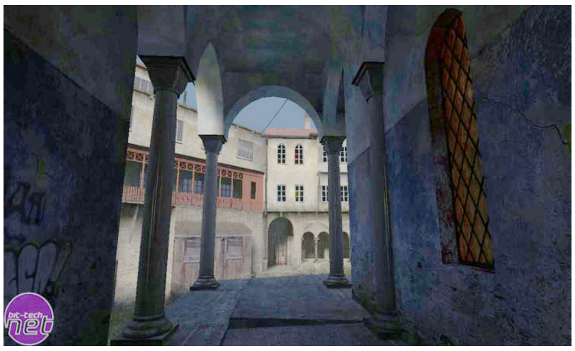
Figure 6.4 : Example of contrast enhancement from an underexposed image. (a) Input, (b) Local correction; (c) Global correction.



(a)

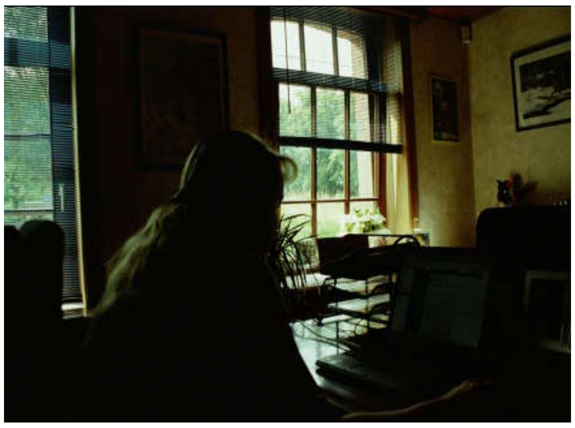


(b)

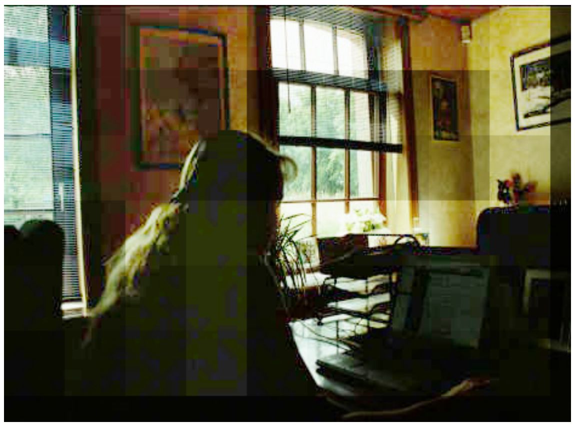


(c)

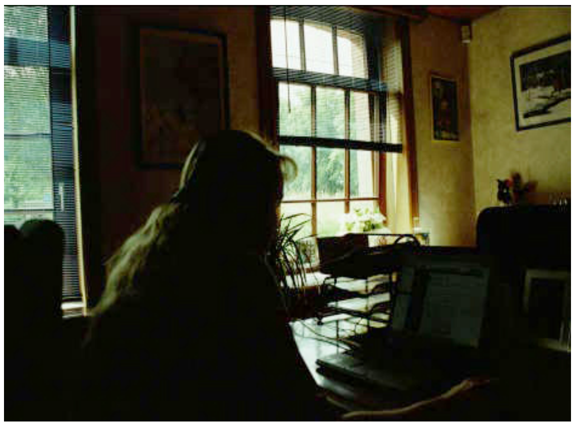
Figure 6.5 : Example of contrast enhancement from an underexposed image. (a) Input, (b) Local correction; (c) Global correction.



(a)

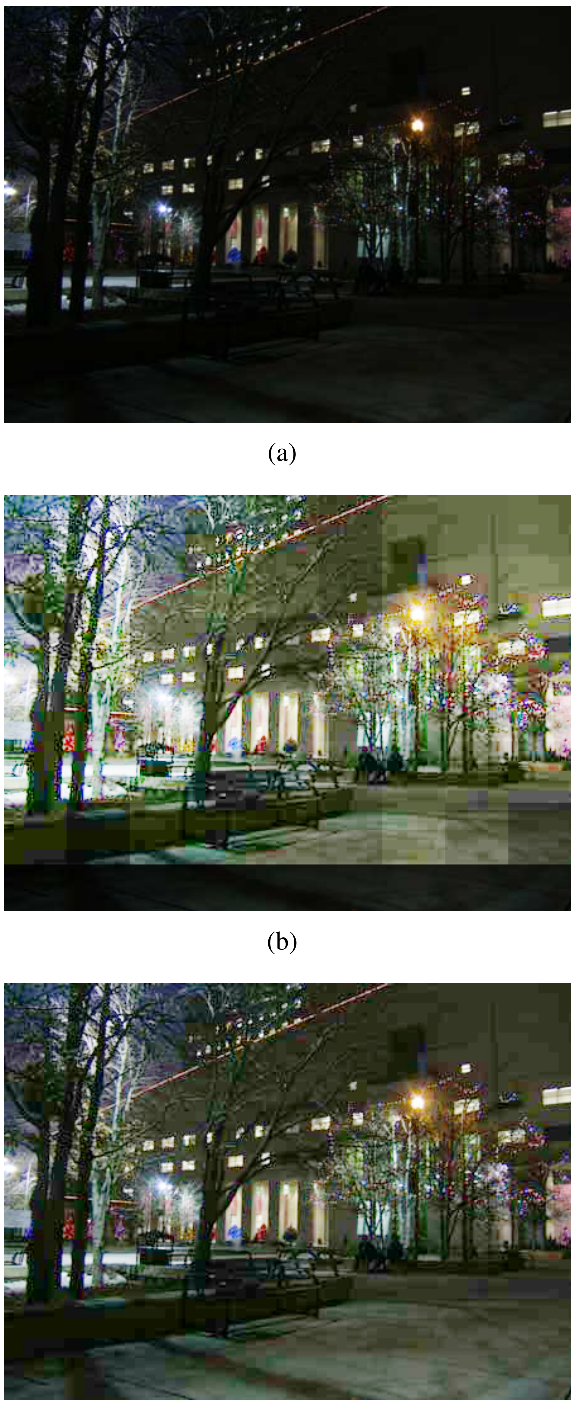


(b)



(c)

Figure 6.6 : Example of contrast enhancement from an underexposed image. (a) Input, (b) Local correction; (c) Global correction.


(a)

(b)

(c)

Figure 6.7 : Example of contrast enhancement from an underexposed image. (a) Input, (b) Local correction; (c) Global correction.

Bibliography

[1] *Fifth Report - Digital Images as Evidence*. House of Lords, science and technology edition, 1998. ISBN-0104064986.

[2] **New York City Surveillance Camera Project**, 1998. http://www.mediaeater.com/cameras/.

[3] *Definitions and Guidelines for the Use of Imaging Technologies in the Criminal Justice System*. Scientific Working Group on Imaging Technologies (SWGIT), 1999. http://www.fbi.gov/hq/lab/fsc/backissu/oct1999/swgit1.htm.

[4] ACDSee. www.acdsee.com.

[5] J. E. Adams. **Interactions between color plane interpolation and other image processing functions in electronic photography**. In *SPIE-IS&T Electronic Imaging*, pages 144–151, 1995.

[6] J. E. Adams JR., J. F. Hamilton JR., and J. A. Hamilton. **Removing color aliasing artifacts from color digital images**. *US Patent*, (US6804392), 2004.

[7] Adobe Photoshop. www.adobe.com/products/photoshop.

[8] D. Alleysson and B. Chaix de Lavarène. **Frequency selection demosaicking: a review and a look ahead**. *Visual Communications and Image Processing (VCIP-2008)*, **6822**(1):13, 2008.

[9] D. Alleysson, S. Süsstrunk, and J. Hérault. **Color Demosaicing by Estimating Luminance and Opponent Chromatic Signals in the Fourier Domain**. In *Color Imaging Conference*, pages 331–336, 2002.

[10] D. Alleysson, S. Susstrunk, and J. Herault. **Linear demosaicing inspired by the human visual system**. *IEEE Transactions on Image Processing*, **14**(4):439–449, 2005.

[11] T. Arici, S. Dikbas, and Y. Altunbasak. **A Histogram Modification Framework and its Application for Image Contrast Enhancement**. *IEEE Transaction on Image Processing*, **18**(9):1921–1935, 2009.

[12] C. Baron. *Adobe Photoshop Forensics*. Course Technology PTR, 1st edition, 2007. ISBN 1598634054.

[13] S. Battiato, A. Bosco, A. Castorina, and G. Messina. **Automatic Image Enhancement by Content Dependent Exposure Correction**. *EURASIP Journal on Applied Signal Processing*, **12**:1849–1860, 2004.

[14] S. Battiato, A. Capra, I. Guarneri, and M. Mancuso. **DCT Optimization for CFA Data Images**. In M. M. Blouke, N. Sampat, and R. J. Motta, editors, *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, **5301** of *Presented at the Society of Photo-Optical Instrumentation Engineers (SPIE) Conference*, pages 429–437, jun 2004.

[15] S. Battiato, A. Castorina, M. Guarnera, and P. Vivirito. **An Adaptive Global Enhancement Pipeline for Low Cost Imaging Sensors**. *IEEE Transactions on Consumer Electronics*, **49**:670–675, 2003.

[16] S. Battiato, A. Castorina, and M. Mancuso. **High dynamic range imaging for digital still camera: an overview**. *Journal of Electronic Imaging*, **12**(3):459–469, July 2003.

[17] S. Battiato, G.M. Farinella, M. Guarnera, G. Messina, and D. Ravì. **Boosting Gray Codes for Red Eyes Removal**. In *International Conference on Pattern Recognition (ICPR 2010)*, Instanbul (TK), 2010.

[18] S. Battiato, G.M. Farinella, M. Guarnera, G. Messina, and D. Ravì. **Red-Eyes Removal Through Cluster Based Boosting on Gray Codes**. *EURASIP Journal on Image and Video Processing, Special Issue on Emerging Methods for Color Image and Video Quality Enhancement*, 2010. To appear.

[19] S. Battiato, G.M. Farinella, M. Guarnera, G. Messina, and D. Ravì. **Red-Eyes Removal Through Cluster Based Linear Discriminant Analysis**. In *International Conference on Image Processing (ICIP 2010)*, Hong Kong, 2010.

[20] S. Battiato, G.M. Farinella, G. Messina, and G. Puglisi. *IISFA Memberbook 2010, Digital Forensics, Condivisione della conoscenza tra i membri dell'IISFA ITALIAN CHAPTER*, chapter 11. Digital Video Forensics: Status e Prospettive, pages 271–296. International Information Systems Forensics Association, December 2010. in press.

[21] S. Battiato, M.I. Guarnera, G. Messina, and V. Tomaselli. **Recent Patents on Color Demosaicing**. *Recent Patents on Computer Science,* Bentham Science Publishers Ltd, **1**(2), 2008.

[22] S. Battiato, M. Mancuso, A. Bosco, and M. Guarnera. **Psychovisual and statistical optimization of quantization tables for DCT compression engines**. In *Image Analysis and Processing, 2001. Proceedings. 11th International Conference on*, pages 602–606, Sep 2001.

[23] S. Battiato, G. Messina, and R. Rizzo. *IISFA Memberbook 2009, Digital Forensics, Condivisione della conoscenza tra i membri dell'IISFA ITALIAN CHAPTER*, chapter 1. Image Forenscis: Contraffazione Digitale e Identificazione della Camera di Acquisizione: Status e Prospettive, pages 1–48. International Information Systems Forensics Association, December 2009.

[24] S. Battiato, G. Puglisi, and A.R. Bruna. **A robust video stabilization system by adaptive motion vectors filtering**. pages 373–376, 23 2008-April 26 2008.

[25] B.E. Bayer. **Color Imaging Array**. *U.S. Patent* 3,971,065, 1976.

[26] P. Benati, R. Gray, and P. Cosgrove. **Automated detection and correction of eye color defects due to flash illumination**. *U.S. Patent*, (US5748764), 1998.

[27] S.A. Bhukhanwala and T.V. Ramabadran. **Automated Global Enhancement Of Digitized Photographs**. *IEEE Transactions on Consumer Electronics*, **40**(1), 1994.

[28] D. Borghesani, C. Grana, and R. Cucchiara. **Color features comparison for retrieval in personal photo collections**. In *ACS'08: Proceedings of the 8th conference on Applied computer scince*, pages 265–268, Stevens Point, Wisconsin, USA, 2008. World Scientific and Engineering Academy and Society (WSEAS).

[29] A. Bruna, A. Capra, S. Battiato, and S. La Rosa. **Advanced DCT rate control by single step analysis**. In *Consumer Electronics, 2005. ICCE. 2005 Digest of Technical Papers. International Conference on*, pages 453–454, Jan. 2005.

[30] C. J. Burges. **A tutorial on support vector machines for pattern recognition**. *Data Mining and Knowledge Discovery*, **2**:121–167, 1998.

[31] W. C. Chan, O. C. Au, and M. F. Fu. **A novel color interpolation framework in modified YCbCr domain for digital cameras**. In *IEEE International Conference on Image Processing (ICIP 2003)*, pages 925–928, 2003.

[32] E. CHANG, S. CHEUNG, AND D. Y. PAN. **Color filter array recovery using a threshold-based variable number of gradients**. In N. SAMPAT AND T. YEH, editors, *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, **3650** of *Presented at the Society of Photo-Optical Instrumentation Engineers (SPIE) Conference*, pages 36–43, mar 1999.

[33] L. CHANG AND Y.-P. TAN. **Effective use of spatial and spectral correlations for color filter array demosaicking**. *IEEE Transactions on Consumer Electronics*, **50**(1):355–365, 2004.

[34] C. CHEN, C. LIN, C. WU, Z. YIN, AND C. WANG. **Demosaicking method and apparatus for color filter array interpolation in digital image acquisition systems**. *U.S. Patent*, (US7292725B2), 2007.

[35] T. CHEN. **A Study of Spatial Color Interpolation Algorithms for Single-Detector Digital Cameras**. Internet Link. http://scien.stanford.edu/ class/ psych221/ projects/ 99/ tingchen/ main.htm.

[36] K.-H. CHUNG AND Y.-H. CHAN. **Color Demosaicing Using Variance of Color Differences**. *Image Processing, IEEE Transactions on*, **15**(10):2944–2955, Oct. 2006.

[37] K.L. CHUNG, W.J. YANG, W.M. YAN, AND C.C. WANG. **Demosaicing of Color Filter Array Captured Images Using Gradient Edge Detection Masks and Adaptive Heterogeneity-Projection**. *IEEE Transactions on Image Processing*, **17**(12):2356–2367, Dec 2008.

[38] P. CORCORAN, P. BIGIOI, E. STEINBERG, AND A. POSOSIN. **Automated in-camera detection of flash eye-defects**. In *Int. Conf. on Consumer Electronics*, 2005.

[39] COREL PAINT SHOP PRO. www.jasc.com.

[40] I. DAUBECHIES. *Ten lectures on wavelets*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1992.

[41] E. DUBOIS. **Filter Design for Adaptive Frequency-Domain Bayer Demosaicking**. In *IEEE International Conference on Image Processing (ICIP 2006)*, pages 2705–2708, 2006.

[42] R. O. DUDA, P. E. HART, AND D. G. STORK. *Pattern Classification (2nd Edition)*. Wiley-Interscience, 2000.

[43] S. EFIMOV, A. NEFYODOV, AND M. RYCHAGOV. **Block-Based Image Exposure Assessment and Indoor/Outdoor Classification**. In *International conference on the Computer Graphics and Vision (GraphiCon'2007)*, Moscow, Russia, June 2007.

[44] FACE.COM. **Photofinder - Face Recognition Application for Facebook.**, 2009. website: http://www.face.com, blog: http://blog.face.com.

[45] H. FARID. **Digital Image Ballistics from JPEG Quantization: A Followup Study**. Technical Report TR2008-638, Department of Computer Science, Dartmouth College, 2008. Available from: `www.cs.dartmouth.edu/farid/publications/tr08.html`.

[46] H. FARID. **Exposing Digital Forgeries from JPEG Ghosts**. *IEEE Transactions on Information Forensics and Security*, **1**(4):154–160, 2009. Available from: `www.cs.dartmouth.edu/farid/publications/tifs09.html`.

[47] H. FARID. **Image Forgery Detection: A Survey**. *IEEE Signal Processing Magazine*, **2**(26):16–25, 2009. Available from: `www.cs.dartmouth.edu/farid/publications/spm09.html`.

[48] H. FARID. **Photo Tampering Throughout History**, 2009. Available from: `http://www.cs.dartmouth.edu/farid/research/digitaltampering/`.

[49] G. M. FARINELLA, S. BATTIATO, G. GALLO, AND R. CIPOLLA. **Natural Versus Artificial Scene Classification by Ordering Discrete Fourier Power Spectra**. In *SSPR & SPR '08: Proceedings of the 2008 Joint IAPR International Workshop on Structural, Syntactic, and Statistical Pattern Recognition*, pages 137–146, Berlin, Heidelberg, 2008. Springer-Verlag.

[50] A. M. FERMAN. **Automatic Detection of Red-Eye Artifacts in Digital Color Photos**. In *Int. Conf. on Image Processing*, 2008.

[51] R. FRANZEN. **Kodak Lossless True Color Image Suite**. Internet Link. http://r0k.us/graphics/ kodak/.

[52] T. W. FREEMAN. **Median Filter for Reconstructing Missing Color Samples**. *US Patent*, (US4724395), 1988.

[53] Y. FREUND AND R. E. SCHAPIRE. **A decision-theoretic generalization of on-line learning and an application to boosting**. In *European Conference on Computational Learning Theory (EuroCOLT 1995)*, pages 23–37. Springer-Verlag, 1995.

[54] J. FRIEDMAN, T. HASTIE, AND R. TIBSHIRANI. **Additive logistic regression: a statistical view of boosting**. *Annals of Statistics*, 2000.

[55] F. GASPARINI AND R. SCHETTINI. **Automatic Redeye Removal for Smart Enhancement of Photos of Unknown Origin**. In *Visual Information and Information Systems (VISUAL-2005)*, **3736** of *Lecture Notes in Computer Science*, pages 226–233, 2005.

[56] F. GASPARINI AND R. SCHETTINI. **Automatic Red-Eye Removal for digital photography**. In R. LUKAC, editor, *Single-Sensor Imaging: Methods and Applications For Digital Cameras*, pages 429–457. CRC Press, 2008.

[57] F. GASPARINI AND R. SCHETTINI. **A Review of Redeye Detection and Removal in Digital Images Through Patents**. *Recent Patents on Electrical Engineering*, 2009.

[58] M. GAUBATZ AND R. ULICHNEY. **Automatic red-eye detection and correction**. In *Int. Conf. on Image Processing*, 2002.

[59] JOHN W. GLOTZBACH, RONALD W. SCHAFER, AND KLAUS ILLGNER. **A method of color filter array interpolation with alias cancellation properties**. In *IEEE International Conference on Image Processing (ICIP-2001)*, pages 141–144, 2001.

[60] R.C. GONZALEZ AND R.E. WOODS. *Digital Image Processing 3rd Edition*. Prentice Hall, 2008.

[61] GOOGLE. **Picasa - Name Tags**, 2009. website: http://picasa.google.com, video demo: http://www.youtube.com/watch?v=teeGF-w5Cpw.

[62] M. GUARNERA, V. TOMASELLI, AND G. MESSINA. **Method and relative device of color interpolation of an image acquired by a digital color sensor**. US Patent Application 2009/0010539, January 2009.

[63] M. I. GUARNERA, G. MESSINA, AND V. TOMASELLI. **Method and system for demosaicing artifact removal, and computer program product therefor**. *US Patent Application*, (US20060087567A1), 2006.

[64] M.I. GUARNERA, G. MESSINA, AND V. TOMASELLI. **Adaptive color demosaicing and false color removal**. *Journal of Electronic Imaging, Special Issue on Digital Photography, SPIE and IS&T*, **19**(2):1–16, 2010.

[65] B. K. Gunturk, Y. Altunbasak, and R. M. Mersereau. **Color plane interpolation using alternating projections**. *IEEE Transactions on Image Processing*, **11**(9):997–1013, 2002. http://www.ece.gatech.edu/ research/ labs/ MCCL/ research/ p_demosaick.html.

[66] J. F. Hamilton and J. E. Adams. **Adaptive color plane interpolation in single sensor color electronic camera**. *U.S. Patent*, (US05629734A), 1997.

[67] J. Y. Hardeberg. **Red Eye Removal using Digital Color Image Processing**. pages 283–287, Montral, Canada, 2001. Image Processing, Image Quality, Image Capture, System Conference.

[68] C. Hass. **JPEG Compression Quality tables for Digital Cameras and Digital Photography Software**. Internet Link. http://www.impulseadventure.com/photo/jpeg-quantization.html.

[69] J. He, Z. Lin, L. Wang, and X. Tang. **Detecting Doctored JPEG Images Via DCT Coefficient Analysis**. In *Computer Vision - ECCV 2006, 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006, Proceedings, Part III*, pages 423–435, 2006.

[70] Y. Hel-or and D. Keren. **Image demosaicing method utilizing directional smoothing**. *U.S. Patent*, (US6404918B1), 2002.

[71] Y. Hel-or and D. Keren. **Image demosaicing method utilizing directional smoothing**. *U.S. Patent*, (US6618503B2), 2003.

[72] A. Held. **Model-Based Correction of Red-Eye Defects**. In *IS&T Color Imaging Conference (CIC-02)*, pages 223–228, 2002.

[73] K. Hirakawa. **Cross-talk explained**. In *IEEE International Conference on Image Processing (ICIP 2008)*, pages 677–680, 2008.

[74] K. Hirakawa and T. W. Parks. **Adaptive homogeneity-directed demosaicing algorithm**. *IEEE Transactions on Image Processing*, **14**(3):360–369, 2005. http://www.accidentalmark.com/ research/ packages/ MNdemosaic.zip.

[75] K. Hirakawa and T. W. Parks. **Joint demosaicing and denoising**. *IEEE Transactions on Image Processing*, **15**(8):2146–2157, 2006.

[76] K. Hirakawa and P. J. Wolfe. **Spatio-Spectral Color Filter Array Design for Enhanced Image Fidelity**. In *IEEE International Conference on Image Processing (ICIP-2007)*, pages 81–84, 2007.

[77] K. Hirakawa and P.J. Wolfe. **Spatio-Spectral Color Filter Array Design for Optimal Image Recovery**. *IEEE Transactions on Image Processing*, **17**(10):1876–1890, 2008.

[78] L. Hongliang, K.N. Ngan, and L. Qiang. **FaceSeg: Automatic Face Segmentation for Real-Time Video**. *IEEE Transactions on Multimedia*, **11**(1):77–88, Jan. 2009.

[79] R.L. Hsu, M. Abdel-Mottaleb, and A.K. Jain. **Face Detection in Color Images**. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **24**(5):696–706, 2002.

[80] A.A. Hunter and S.B. Pollard. **Electronic image colour plane reconstruction**. *U.S. Patent Application*, (US20030086606A1), 2003.

[81] IPLab. **Image Processing Laboratory - Forensic Database**. link: http://iplab.dmi.unict.it/index.php? option=com_docman&Itemid=111.

[82] O. Kalevo and H. Rantanen. **Color filter array interpolation**. *U.S. Patent Application*, (US20040218073A1), 2004.

[83] D. KEREN. **Image demosaicing method**. *U.S. Patent*, (US6625305B1), 2003.

[84] C. W. KIM AND M. G. KANG. **Noise insensitive high resolution demosaicing algorithm considering cross-channel correlation**. In *IEEE International Conference on Image Processing (ICIP 2005)*, pages 1100–1103, 2005.

[85] R. KIMMEL. **Demosaicing: Image reconstruction from CCD samples**. *IEEE Transactions on Image Processing*, **8**(6):1221–1228, Jun 1999.

[86] K. KIMURA AND T. YOSHIDA. **Data processing apparatus, image processing apparatus, camera, and data processing method**. *U.S. Patent Application*, (US20050007470A1), 2005.

[87] X. LI. **Demosaicing by successive approximation**. *IEEE Transactions on Image Processing*, **14**(3):370–379, 2005.

[88] X. LI, B. K. GUNTURK, AND L. ZHANG. **Image demosaicing: a systematic survey**. In *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, **6822** of *Presented at the Society of Photo-Optical Instrumentation Engineers (SPIE) Conference*, January 2008.

[89] N. LIAN, L. CHANG, Y.-P. TAN, AND V. ZAGORODNOV. **Adaptive Filtering for Color Filter Array Demosaicking**. *IEEE Transactions on Image Processing*, **16**(10):2515–2525, 2007.

[90] R. LIENHART, E. KURANOV, AND V. PISAREVSKY. **Empirical Analysis of Detection Cascades of Boosted Classifiers for Rapid Object Detection**. In *In DAGM 25th Pattern Recognition Symposium*, pages 297–304, 2003.

[91] R. LIENHART AND J. MAYDT. **An extended set of Haar-like features for rapid object detection**. In *International Conference on Image Processing (ICIP 2002)*, **1**, pages I–900–I–903 vol.1, 2002.

[92] W. LU AND Y.-P. TAN. **Color filter array demosaicking: new method and performance measures**. *IEEE Transactions on Image Processing*, **12**(10):1194–1210, 2003.

[93] R. LUKAC. *Single-Sensor Imaging: Methods and Applications for Digital Cameras*. CRC Press, Inc., Boca Raton, FL, USA, 2008.

[94] R. LUKAC, K. MARTIN, AND K. N. PLATANIOTIS. **Demosaicked image postprocessing using local color ratios**. *IEEE Transactions on Circuits and Systems for Video Technology*, **14**(6):914–920, 2004. http://www.dsp.utoronto.ca/˜lukacr/download.

[95] R. LUKAC AND K. N. PLATANIOTIS. **A Robust, Cost-Effective Postprocessor for Enhancing Demosaicked Camera Images**. *Real-Time Imaging, Special Issue on Spectral Imaging II*, **11**(2):139–150, 2005. http://www.dsp.utoronto.ca/˜lukacr/download.

[96] R. LUKAC AND K.N. PLATANIOTIS. **A normalized model for color-ratio based demosaicking schemes**. In *Image Processing, 2004. ICIP '04. 2004 International Conference on*, **3**, pages 1657–1660, Oct. 2004.

[97] R. LUKAC, K.N. PLATANIOTIS, D. HATZINAKOS, AND M. ALEKSIX. **A novel cost effective demosaicing approach**. *Consumer Electronics, IEEE Transactions on*, **50**(1):256–261, 2004.

[98] H. LUO, J. YEN, AND D. TRETTER. **An Efficient Automatic Redeye Detection and Correction Algorithm**. In *Int. Conf. on Pattern Recognition*, 2004.

[99] M. MANCUSO AND S. BATTIATO. **An Introduction to the Digital Still Camera Technology**. *ST Journal of System Research*, **2**(2):1–9, 2001.

[100] S. MANN. **Comparametric equations with practical applications in quantigraphic image processing**. *Image Processing, IEEE Transactions on*, **9**(8):1389–1406, Aug 2000.

[101] L. MARCHESOTTI, M. BRESSAN, AND G. CSURKA. **Safe Red-Eye Correction Plug-in Using Adaptive Methods**. In *International Conference on Image Analysis and Processing - Workshops (ICIAPW-07)*, pages 192–165, 2007.

[102] R. P. MAURER. **Reduction of chromatic bleeding artifacts in image**, 2003.

[103] D. MENON, S. ANDRIANI, AND G. CALVAGNO. **Demosaicing With Directional Filtering and a posteriori Decision**. *IEEE Transactions on Image Processing*, **16**(1):132–141, 2007. http://www.danielemenon.it/pub/dfapd/dfapd.html.

[104] D. MENON AND G. CALVAGNO. **Demosaicing Based on Wavelet Analysis of the Luminance Component**. In *IEEE International Conference on Image Processing (ICIP 2007)*, **2**, pages 181–184, 2007.

[105] G. MESSINA, M. GUARNERA, V. TOMASELLI, A.R. BRUNA, G. SPAMPINATO, AND A. CASTORINA. **Color interpolation method of an image acquired by a digital sensor by directional filtering**. *U.S. Patent*, (US7305123B2), 2007.

[106] G. MESSINA AND T. MECCIO. **Red Eye Removal**. In S. BATTIATO, A. R. BRUNA, G. MESSINA, AND G. PUGLISI, editors, *Image Processing for Embedded Devices*, Applied Digital Imaging Ebook Series. Bentham Science, 2010.

[107] X.-P. MIAO AND T. SIM. **Automatic red-eye detection and removal**. In *International Conference on Multimedia and Expo (ICME-2004)*, 2004.

[108] J. M. MIR. **Apparatus & Method for Minimizing Red-Eye in Flash Photography**. *U.S. Patent*, (US4285588), 1981.

[109] T. NGUYEN. **System and method for asymmetrically demosaicing raw data images using color discontinuity equalization**. *U.S. Patent Application*, (US20020167602A1), 2002.

[110] A. NILSSON AND P. NORDBLOM. **Weighted gradient based and color corrected interpolation**. *U.S. Patent Application*, (US20070292022A1), 2007.

[111] S. NOHDA. **Image pickup apparatus having an interpolation function**. *U.S. Patent*, (US6295087B1), 2001.

[112] N. O'HARE AND A.F. SMEATON. **Context-Aware Person Identification in Personal Photo Collections**. *IEEE Transactions on Multimedia*, **11**(2):220–228, Feb. 2009.

[113] Y. OTHA, T. KANADE, AND T. SAKAI. **Color information for region segmentation**. *Computer Graphic Image Processing*, **13**:222 – 241, 1980.

[114] M. K. ÖZKAN, A. M. TEKALP, AND M. I. SEZAN. **POCS-based restoration of space-varying blurred images**. *IEEE Transactions on Image Processing*, **3**(4):450–454, 1994.

[115] D. PALIY, V. KATKOVNIK, R. BILCU, S. ALENIUS, AND K. EGIAZARIAN. **Spatially adaptive color filter array interpolation for noiseless and noisy data: Articles**. *Int. J. Imaging Syst. Technol.*, **17**(3):105–122, 2007.

[116] R. PALUM. **Image Sampling with the Bayer Color Filter Array**. In *Image Processing, Image Quality, Image Capture Systems Conference (PICS-01)*, pages 239–245, Montral, Quebec, Canada, 2001. IS&T. ISBN 0-89208-232-1.

[117] A. J. PATTI, K. KONSTANTINIDES, D. TRETTER, AND Q LIN. **Automatic Digital Redeye Reduction**. In *Int. Conf. on Image Processing*, 1998.

[118] W. B. PENNEBAKER AND J. L. MITCHELL. *JPEG Still Image Data Compression Standard.* Kluwer Academic Publishers, Norwell, MA, USA, 1992.

[119] F. A. P. PETITCOLAS. **Watermarking schemes evaluation**. *IEEE Signal Processing*, **17**(5):58–64, September 2000.

[120] F. A. P. PETITCOLAS, R. J. ANDERSON, AND M. G. KUHN. **Attacks on copyright marking systems**. In DAVID AUCSMITH, editor, *Information Hiding, Second International Workshop, IH98*, pages 219–239, Portland, Oregon, U.S.A., April 1998. Springer-Verlag. ISBN 3-540-65386-4.

[121] G. PETSCHNIGG, R. SZELISKI, M. AGRAWALA, M. F. COHEN, H. HOPPE, AND K. TOYAMA. **Digital photography with flash and no-flash image pairs**. *ACM Transaction on Graphics*, 2004.

[122] S.L. PHUNG, A. BOUZERDOUM, AND D. CHAI. **A Novel Skin Color Model In YCbCr Color Space And Its Application To Human Face Detection**. In *International Conference on Image Processing*, **1**, pages 289–292, 2002.

[123] S.L. PHUNG, A. BOUZERDOUM, AND D. CHAI. **Skin segmentation using color pixel classification: analysis and comparison**. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **27**(1):148–154, Jan. 2005.

[124] R. RASKAR, A. AGRAWAL, AND J. TUMBLIN. **Coded Exposure Photography:Motion Deblurring using Flattered Shutter**. In *SIGGRAPH - International Conference on Computer Graphics and Interactive Techniques*, Boston, July-August 2006.

[125] K. REVATHY, G. RAJU, AND R. PRABHAKARAN NAYAR. **Image Zooming by Wavelets**. *Fractals - an Interdisciplinary Journal on the Complex Geometry*, **8**(3):247–254, 2000.

[126] L. TRUPPIA S. BATTIATO, G. MESSINA. **An Improved Benchmarking Dataset for Forgery Detection Strategies**. In *Minisymposium on Image and Video Forensics*, Cagliari (Italy), June 2010. SIMAI.

[127] T. L. SAATY. *Decision Making for Leaders: The Analytic Hierarchy Process for Decisions in a Complex World*, **2**. Analytic Hierarchy Process Series, 2001. New Edition.

[128] I. V. SAFONOV. **Automatic Red-Eye Detection**. In *International conference on the Computer Graphics and Vision*, 2007.

[129] R. E. SCHAPIRE. **The strength of weak learnability**. In *Machine Learning*, pages 197–227, 1990.

[130] R. E. SCHAPIRE. **The boosting approach to machine learning: An overview**. In *MSRI Workshop on Nonlinear Estimation and Classification*, 2001.

[131] J. S. SCHILDKRAUT AND R. T. GRAY. **A fully automatic redeye detection and correction algorithm**. In *Int. Conf. on Image Processing*, 2002.

[132] N. SERRANO, A. SAVAKIS, AND J. LUO. **A computationally efficient approach to indoor/outdoor scene classification**. In *International Conference on Pattern Recognition (ICPR 2002)*, pages 146–149, 2002.

[133] B. SMOLKA, K. CZUBIN, J. Y. HARDEBERG, K. N. PLATANIOTIS, M. SZCZEPANSKI, AND K. W. WOJCIECHOWSKI. **Towards automatic redeye effect removal**. *Pattern Recognition Letters*, **24**(11):1767–1785, 2003.

[134] M. SORIANO, B. MARTINKAUPPI, AND M. LAAKSONEN S. HUOVINEN. **Skin Color Modeling Under Varying Illumination Conditions Using the Skin Locus for Selecting Training Pixels**. *Real-time Image Sequence Analysis (RISA 2000)*, 2000.

[135] G. SPAMPINATO, A. BRUNA, G. SANGUEDOLCE, E. ARDIZZONE, AND M. LA CASCIA. **Improved color interpolation using discrete wavelet transform**. *Image and Video Communications and Processing 2005*, **5685**(1):753–760, 2005. Available from: `http://link.aip.org/link/?PSI/5685/753/1`.

[136] H. STARK AND P. OSKOUI. **High-resolution image recovery from image-plane arrays, using convex projections**. *Journal of the Optical Society of America A*, **6**:1715–1726, 1989.

[137] STMICROELECTRONICS. **Colour Sensor Evaluation Kit VV6501**. Edinburgh - *www.edb.st.com/products/image/sensors/*501/6501*evk.htm.*

[138] C.-Y. SU. **Highly effective iterative demosaicing using weighted-edge and color-difference interpolations**. *Consumer Electronics, IEEE Transactions on*, **52**(2):639–645, May 2006.

[139] K. TAJIMA. **Image processing apparatus**. *U.S. Patent Application*, (US20050058361A1), 2005.

[140] V. TOMASELLI, M. I. GUARNERA, AND G. MESSINA. **False-color removal on the YCC color space**. In *SPIE-IS&T Electronic Imaging*, **7250**, 2009.

[141] A. TORRALBA AND K. P. MURPHY. **Sharing Visual Features for Multiclass and Multiview Object Detection**. *IEEE Trans. Pattern Anal. Mach. Intell.*, **29**(5):854–869, 2007.

[142] P. VIOLA AND M. JONES. **Robust real-time face detection**. *International Journal of Computer Vision*, **57**(2):137–154, 2004.

[143] F. VOLKEN, J. TERRIER, AND P. VANDEWALLE. **Automatic Red-Eye Removal based on Sclera and Skin Tone Detection**. In *European Conference on Color in Graphics, Imaging and Vision*, 2006.

[144] J. WILLAMOWSKI AND G. CSURKA. **Probabilistic Automatic Red Eye Detection and Correction**. In *IEEE International Conference on Pattern Recognition (ICPR-06)*, pages 762–765, 2006.

[145] X. WU AND N. ZHANG. **Primary-consistent soft-decision color demosaicking for digital cameras (patent pending)**. *IEEE Transactions on Image Processing*, **13**(9):1263–1274, 2004.

[146] J. YANG, W. LU, AND A. WAIBEL. **Skin-colour Modeling and Adaptation**. *Technical Report CMU-CS-97-146 School of Computer Science, Carnegie Mellon University*, 1997.

[147] M.-H. YANG, D.J. KRIEGMAN, AND N. AHUJA. **Detecting faces in images: a survey**. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **24**(1):34–58, Jan 2002.

[148] R. YANG, L. YIN, M. GABBOUJ, J. ASTOLA, AND Y. NEUVO. **Optimal weighted median filtering under structural constraints**. *Signal Processing, IEEE Transactions on*, **43**(3):591–604, Mar 1995.

[149] S. YE, Q. SUN, AND E.-C. CHANG. **Detecting Digital Image Forgeries by Measuring Inconsistencies of Blocking Artifact**. In *Proceedings of the 2007 IEEE International Conference on Multimedia and Expo, ICME 2007, July 2-5, 2007, Beijing, China*, pages 12–15, 2007.

[150] S. YOO AND R.-H. PARK. **Red-eye detection and correction using inpainting in digital photographs**. *IEEE Transactions on Consumer Electronics*, **55**(3):1006–1014, 2009.

[151] B.D. ZARIT, B.J. SUPER, AND F.K.H. QUEK. **Comparison of Five Colour Models in Skin Pixel Classification**. In *International Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems, IEEE Computer Society*, pages 58–63, 1999.

[152] L. ZHANG, Y. SUN, M. LI, AND H. ZHANG. **Automated red-eye detection and correction in digital photographs**. In *Int. Conf. on Image Processing*, 2004.

[153] L. ZHANG AND X. WU. **Color demosaicking via directional linear minimum mean square-error estimation**. *Image Processing, IEEE Transactions on*, **14**(12):2167–2178, Dec. 2005.