PH.D. SCHOOL
UNIVERSITY OF MILANO-BICOCCA

Department of Informatics, Systems and Communication
Ph.D. Program in Computer Science - XXXIV Cycle

Image Enhancement and Restoration using Machine Learning Techniques

Ph.D. Dissertation of: Zini Simone

Supervisor: Prof. Raimondo Schettini
Co-Supervisor: Prof. Simone Bianco
Tutor: Prof. Giuseppe Vizzari
Ph.D. Coordinator: Prof. Leonardo Mariani

Academic Year 2020-2021

I would like to thank the people who have been close to me and the ones who inspired and supported me during my years of Ph.D. I would like to start by saying thank you to Raimondo and Simone: thank you for giving me the opportunity to do this experience and for guiding me during those years.
Then a very huge thank you goes to Marco Buzzelli, who has been my biggest guide and whose presence had changed the way I've faced these years. I would like also to thank my lab mates from the Imaging and Vision Laboratory with which I've shared time, knowledge, and laughs.
Another special thank you goes to the persons that were my supervisors during the period I've spent in Japan: Takahiro Toizumi and Kazutoshi Sagi. If it wasn't for you probably I would have never gotten the right strength and self-consciousness to start this life experience.
I would like now to say thank you to my family, my mum and dad who have helped me during this whole time, giving me the possibility to follow my ambitions and growing me into the man I am today. Thank you to my brother Daniele, which always brings into my life happiness and a different way to view things, sometimes remembering me to chill and to "enjoy the ride" a little more. Thanks to my closest friends, who have always been close to me, making every hard day much lighter and making me laugh also during difficult times. And also, thanks to Zip, my cat, who kept me company while writing this thesis and during the lockdown period working from home.
Last but not least, the biggest thanks goes to Ilaria. You have always been close to me during these three years, supporting me during every moment and giving me the strength to continue with my passion every time I felt down or not good enough for this kind of job. Thank you, you have always been there when I needed it most.

Abstract

Digital cameras record, manipulate, and store information electronically through sensors and built-in computers, which makes photography more available to final users which do not anymore need to rely on the use of chemicals and knowledge of mechanical procedures to develop their pictures. Different types of degradation and artifacts can affect images acquired using digital cameras, decreasing the perceptual fidelity of images and making harder many image processing and analysis tasks that can be performed on the collected images. Three elements can be identified as possible sources of artifacts in an image: the scene content, the hardware limitations and flaws, and finally the operations performed by the digital camera processing pipeline itself, from acquisition to compression and storing. Some artifacts are not directly treated in the typical camera processing pipeline, such as the presence of haze or rain that can reduce visibility of the scene in the depicted images. These artifacts require the design of ad hoc methods that are usually applied as post-processing on the acquired images. Other types of artifacts are related to the imaging process and to the image processing pipeline implemented on board of digital cameras. These include sensor noise, undesirable color cast, poor contrast and compression artifacts.

The objective of this thesis is the identification and design of new and more robust modules for image processing and restoration that can improve the quality of the acquired images, in particular in critical scenarios such as adverse weather conditions, poor light in the scene etc... . The artifacts identified are divided into two main groups: "in camera-generated artifacts" and "external artifacts and problems".

In the first group it has been identified and addressed four main issues: sensor camera noise removal, automatic white balancing, automatic contrast enhancement and compression artifacts removal. The design process of the proposed solutions has considered efficiency aspects, due to the possibility of directly integrating them in future camera pipelines.

The second group of artifacts are related to the presence of elements in the scene which may cause a degradation in terms of visual fidelity and/or usability of the images. In particular the focus is on artifacts induced by the presence of rain in the scene.

The thesis, after a brief review of the digital camera processing pipeline, analyzes the different types of artifacts that can affect image quality, and describes the design of the proposed solutions. All the proposed approaches are based on machine learning techniques, such as Convolutional Neural Networks and Bayesian optimization procedure, and are experimentally validated on standard images datasets.

The overall contributions of this thesis can be summarized in three points: integration of classical imaging approaches with machine learning optimization techniques, design of novel deep learning architectures and approaches and analysis and application of deep learning image processing algorithms in other computer vision tasks.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In the photography field, camera manufacturers have moved their interest from old analogical film cameras to the newer digital ones, which have brought a simplification of the photographs development procedure. Digital cameras record, manipulate, and store information electronically through sensors and built-in computers, which makes photography more available to final users which do not anymore need to rely on the use of chemicals and knowledge of mechanical procedures to develop their pictures. To create an image of a scene, digital cameras use lens systems that focus light from a scene onto a sensor which converts analog information to digital one, which is then processed by what is called the *Digital Camera Processing Pipeline* [120].

One important aspect of the pictures recorded and processed by cameras is the final image quality. Different types of degradation and artifacts can affect images acquired using digital cameras, decreasing the quality of an image and making harder many image processing and analysis tasks that can be performed on the collected images. Three elements can be identified as the possible source of artifacts in an image: the scene content, the hardware limitations and flaws, and finally the operations performed by the digital camera processing pipeline itself, from acquisition to compression and storing. While certain kinds of artifacts are not directly treated in the typical camera processing pipeline, such as the presence of haze or rain in the image which can reduce visibility and make other computer vision tasks much harder to perform, some others are explicitly treated in the camera processing pipeline. However, even if specific in-camera processing modules are designed to treat specific acquisition artifacts, due to the limitations of the standard approaches implemented, some kind of artifacts can still occur in output images.

1.1 Focus of this work

The objective of this thesis is the identification and design of new and more robust modules for image processing and restoration that can improve the quality of the acquired images, in particular in critical scenarios such as adverse weather conditions, poor light in the scene etc...

The artifacts identified are divided into two main groups: "in camera-generated artifacts" and "external artifacts". In the first group it has been identified and addressed four main issues: sensor camera noise removal, automatic white balancing, automatic contrast enhancement and compression artifacts removal. The design process of the proposed solutions has considered efficiency aspects, due to the possibility of directly integrating them in future camera pipelines. The second group of artifacts are related to the presence of elements in the scene which may cause a degradation in terms of visual fidelity and/or usability of the images. In particular the focus is on artifacts induced by the presence of rain in the scene.

The thesis, after a brief review of the digital camera processing pipeline, analyzes the different types of artifacts that can affect image quality, and describes the design of the proposed solutions. All the proposed approaches are based on machine learning techniques, such as Convolutional Neural Networks and Bayesian optimization procedure, and are experimentally validated on standard images datasets.

1.2 Thesis outline

The thesis is structured in three parts. The first one gives an overview of the image formation process, describing for first the Human Visual System and then the mathematical model which describes the entire digital image formation process. Then the typical digital camera color processing pipeline is introduced alongside a detailed description of each module that composes that pipeline. Finally, an overview of the types of artifacts that can affect images is presented. This is the content of part I.

In part II are treated all the in camera generated artifacts identified. This part is made of three different chapters. Chapter 5 presents a lightweight combinational approach to perform Automatic White Balancing operation, which has been also extended to the video domain. In Chapter 6 is discussed a framework for optimization of algorithms for contrast enhancement and correction

that can be adopted to optimize the processing blocks in relation to user preferences. Finally, in Chapter 7 an Auto-Encoder Neural Network for reduction of JPEG artifacts introduced by compression operation is described, alongside an extension on camera noise artifacts. All of the approaches presented in these chapters are presented alongside an analysis of the corresponding prior work and detailed tests and comparisons.

In part III are treated the artifacts identified as external artifacts related to pictures taken during rainy weather conditions. Here two chapters are present. Chapter 8 deals with the rain streaks and haze induced by the presence of rain. Here a Generative Adversarial Network for rain removal is first described and then used to analyze the impact of this kind of artifacts on two different computer vision tasks: Optical Character Recognition and Semantic Segmentation. Instead, Chapter 9 deals with the problem of raindrop removal. Here a new approach based on an autoencoder neural network that exploits frequency-based decomposition is presented.

Finally, Chapter 10 ends the thesis summarizing the results obtained, reporting the conclusions, and giving the directions for future works. Bibliography is given after this final chapter.

1.3 Scientific contributions

The main contributions of this thesis can be summarized into three different groups: *(i)* integration of classical imaging approaches with machine learning optimization techniques, *(ii)* design of novel deep learning architectures and approaches, and *(iii)* analysis of the impact of deep learning in imaging tasks.

The list of the manuscripts produced during the Ph.D. period, published or under review procedure, is reported in the next section. In this list are reported extra publications and works that have not been included in the thesis, which mainly covers the analysis and impact of color information in different applications such as auto white balancing field (P#10 P#11) and unsupervised learning (P#8) and image processing in unconventional shooting conditions such as remote sensing scenario (P#9).

- **Integration of classical imaging approaches with machine learning optimization techniques**

  When addressing the problem of designing processing blocks of a digital camera pipeline, methods performance and efficiency play an equally

important role. Computational complexity and memory usage are big restrictions in the design process, making the use of deep machine learning model in general impossible. Starting from these considerations, I have selected two important steps of the digital camera processing pipeline and designed new frameworks for efficient exploitation of machine learning inside digital camera processing blocks.

I've first designed and tested a combination framework for classical, physical-based, auto white balancing approaches. In paper P#6 the analysis of the proposed approach is done in terms of performances, with evaluations with standard metrics and datasets, and in terms of computational complexity and efficiency. The same framework, due to the good results in terms of computational costs, has also been adopted and tested in the field of video color constancy.

Another contribution regards the design and analysis of a framework for optimization of image contrast enhancement algorithms with the use of Bayesian Optimization strategy. Paper P#5 presents the results obtained by adopting the proposed framework with simple and more complex methods for image contrast enhancement. The framework has been proposed for both post-processing image enhancement optimization and onboard algorithm optimization.

- **Design of novel deep learning architectures and approaches**

  Moving to what is called post-processing operations, where hardware constraints are no more necessary since those kinds of operations can be performed outside of the camera pipeline, three different approaches have been designed exploiting deep learning techniques.

  In P#1 I've designed a new approach for compression artifacts image restoration, capable to treat jpeg artifacts at different magnitudes without any additional information. A procedure based on mixed data for training and in-depth analysis of the model alongside comparisons with the state of the art approaches is presented.

  In P#7 I have designed a new architecture for frequency decomposition-based image restoration. Based on the analysis of raindrop artifacts' appearance and impact on different frequencies of the corrupted images, I have designed a new architecture, a new training procedure, and the resulting approach has been widely analyzed and compared. My main contribution here is the presentation of a model which only relies on

image frequency information to restore the images, whereas the other state-of-the-art approaches generally rely on extra information which needs to be computed externally.

- **Analysis of the impact of deep learning in imaging tasks**

  The last typology of contribution regards the analysis of the impact of the application of image processing operations on other tasks that can be considered after applying a certain digital processing pipeline over images. In particular P#2 and P#3 present an analysis of the impact of image processing deep learning operations on two different computer visions tasks: Optical Character Recognition and Semantic Segmentation. Here an analysis of the impact of rain and rain-induced haze on those tasks, alongside an analysis of the effect of a deep learning GAN-based approach is presented. These works have the objective of showing a new way of analyzing deep learning approaches for image restoration, with the intent of putting the basis for new possibilities in driving the training of such kind of restoration models.

1.3.1 Articles

Published articles:

P#1 S. Zini, S. Bianco, and R. Schettini. Deep residual autoencoder for blind universal jpeg restoration. *IEEE Access*, 8:63283–63294, 2020

P#2 S. Zini, S. Bianco, and R. Schettini. Cnn-based rain reduction in street view images. In *Proceedings of the 2020 London Imaging Meeting*, pages 78–81, 2020. doi: doi.org/10.2352/issn.2694-118X.2020.LIM-12

P#3 S. Zini and M. Buzzelli. On the impact of rain over semantic segmentation of street scenes. In *Workshop on Metrification and Optimization of Input Image Quality in Deep Networks, ICPR 2020*, pages 597–610. Springer, 2021

P#4 A. Abdelhamed, R. Timofte, and M. S. Brown. Ntire 2019 challenge on real image denoising: Methods and results. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019

Submitted articles:

P#5 S. Zini, M. Buzzelli, S. Bianco, and R. Schettini. A framework for contrast enhancement algorithms optimization. In *Submitted at International Conference on Image Processing*, 2022

P#6 S. Zini, M. Buzzelli, S. Bianco, and R. Schettini. Cocoa: Combining color constancy algorithms for images and videos. *Submitted at IEEE Transactions on Computational Imaging*, 2022

P#7 S. Zini and M. Buzzelli. Laplacian encoder-decoder network for raindrop removal. *Submitted at Pattern Recognition Letters, Special issue VSI:VETERAN*, 2022

Extra publications:

P#8 S. Zini, M. Buzzelli, B. Twardowski, and J. van de Weijer. Planckian jitter: enhancing the color quality of self-supervised visual representations. In *Submitted at International Conference on Machine Learning*, 2022

P#9 T. Toizumi, S. Zini, K. Sagi, E. Kaneko, M. Tsukada, and R. Schettini. Artifact-free thin cloud removal using gans. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 3596–3600. IEEE, 2019

P#10 S. Bianco, M. Buzzelli, G. Ciocca, R. Schettini, M. Tchobanou, and S. Zini. Analysis of biases in automatic white balance datasets. In *Proceedings of the International Colour Association (AIC) Conference 2021. Milan, Italy. AIC*, pages 233–238, 2021

P#11 M. Buzzelli, S. Zini, S. Bianco, G. Ciocca, R. Schettini, and M. Tchobanou. Analysis of biases in automatic white balance datasets and methods. *Submitted at Color Research and Application*, 2022

Part I

Color Image Processing Pipeline

This first part introduces the theory behind the formation of the images in the Human Visual System (HVS) and the Digital Still Cameras (DSC). The digital camera processing pipeline will be presented and each component will be described to give a complete overview of the image generation process. The analysis of each of the building blocks of the pipeline is based on the works of [120] [141].

Chapter 2

Image Formation

In this section is described the process of image formation in the Human Visual System (HVS) and the way it is reproduced for the digital cameras. Here the fundamentals are introduced in order to give a baseline knowledge to then define, in section 3, the digital camera processing pipeline and its multiple components.

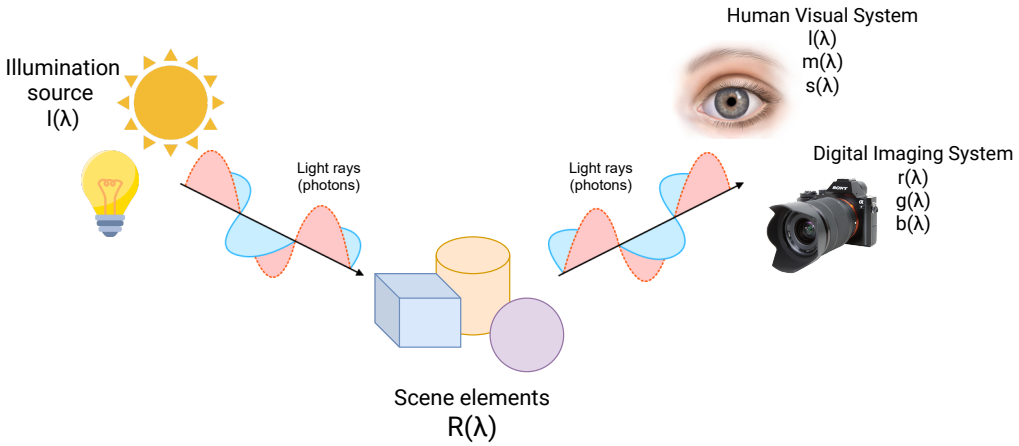


Figure 2.1: The trichromatic output of the Human Visual System or Digital Imaging System is given by the combination of multiple elements: the source of illumination in a scene, the elements in the scene and the receptor sensible to the light rays.

2.1 Human visual system

In most of the cases, the camera image processing hardware and pipeline have been designed on the natural design of human eyes. In that sense it is useful to first introduce the way images are formed in the human eyes and brain to then describe and analyse digital cameras design and imaging pipeline elements.

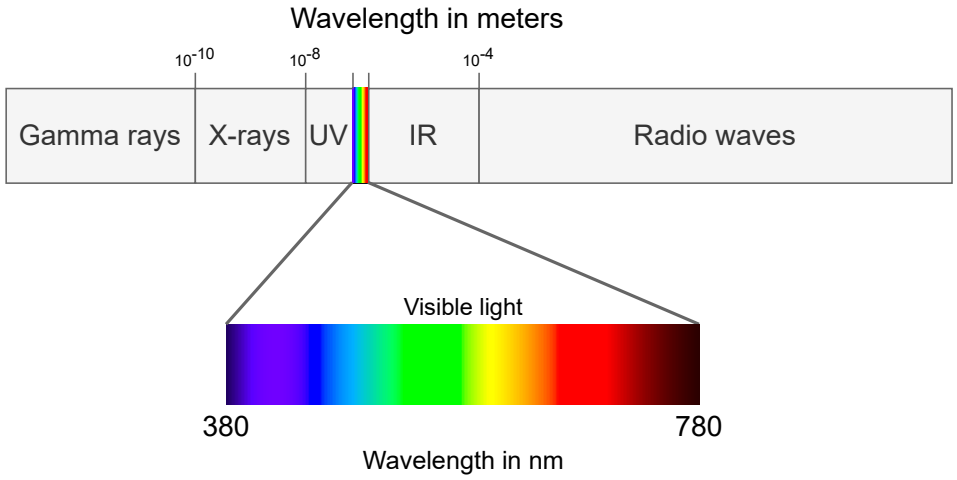


Figure 2.2: The regions of the electromagnetic spectrum, highlighting the optical spectrum which includes the visible and ultraviolet regions.

Visible light can be described as a function of power versus wavelength. This function is called *spectral distribution function*, or *spectrum*. The color of the light depends on the distribution of the energy over the visible spectrum; different wavelengths appear different colors. Visible spectrum roughly spans from $\lambda = 380\ nm$ to $\lambda = 780\ nm$. The interpretation of different colors is based on the wavelength of the electromagnetic waves that reaches the human eyes. Wavelengths of value λ lower than 380 nm are called Ultra-Violets (UV), while for values of λ higher than 780 nm the range is called Infra-Red (IR). These ranges are not visible by humans but find large use in image analysis, for example in medical imaging and radiology. The electromagnetic spectrum spans the total range of wavelengths of electromagnetic radiation from the shortest to the longest wavelength that can be generated physically. This range of wavelengths spans practically from zero to near infinity and can be broadly divided into regions as shown in Table 2.1, which includes radio waves, infrared, visible, ultraviolet, X-rays, and gamma rays. This division is

Table 2.1: Regions of the electromagnetic spectrum. Table from [206].

Wavelength range (nm)	Frequency range (s^{-1})	Description
0.1 nm	$10^{20} - 10^{23}$	Gamma rays
$0.1 - 10$ nm	$10^{17} - 10^{20}$	X-rays
$10 - 380$ nm	$10^{15} - 10^{17}$	Ultraviolet
$380 - 780$ nm	$10^{14} - 10^{15}$	Visible
700 nm to 1 mm	$10^{11} - 10^{14}$	Infrared
1 mm to 1 cm	$10^{10} - 10^{11}$	Microwaves
1 cm to 100 km	$10^{3} - 10^{10}$	Radio waves
$100 - 1,000$ km	$10^{2} - 10^{3}$	Audio frequency

not exact since there is a gradual transition from one region to the next, which is shown schematically in Figure 2.2.

The formation of images in the viewer eyes is related to three main elements: the scene illuminant, which is the source of light in a scene, the objects in the scene, which reflect the light coming from the scene illuminant, and the light receptor in the human eyes.

The illumination in a scene is described by its spectral power distribution curve, which shows the strength of the electromagnetic radiation at different wavelengths λ. In figure 2.3 are shown few examples of spectral power distributions of common illuminant sources. The spectral power distribution is denoted by $I(\lambda)$.

When the electromagnetic radiation is emitted by the illuminant, it can be absorbed, reflected or transmitted by object that collides with the radiation. For different items the portion of radiation that is reflected or transmitted varies with wavelengths and is an inherent characteristic of the objects material, totally independent from the type of illumination in the scene. This property can be characterized by a function of the wavelength, called object *spectral reflectance* or *spectral transmission*, and it is denoted as $R(\lambda)$. To illustrate, the spectral reflectance corresponding to several typical object colors have been plotted. These colors represent patches taken from the GretagMacbeth Color Checker which is often used to test digital camera performances. The spectral reflectance plots are shown in Figure 2.4. For each plot, the y-axis denotes the fraction of light that is being reflected from the object.

Illuminance and object reflectance together determine what is called *color stimulus*. The spectral power distribution determines the amount of energy

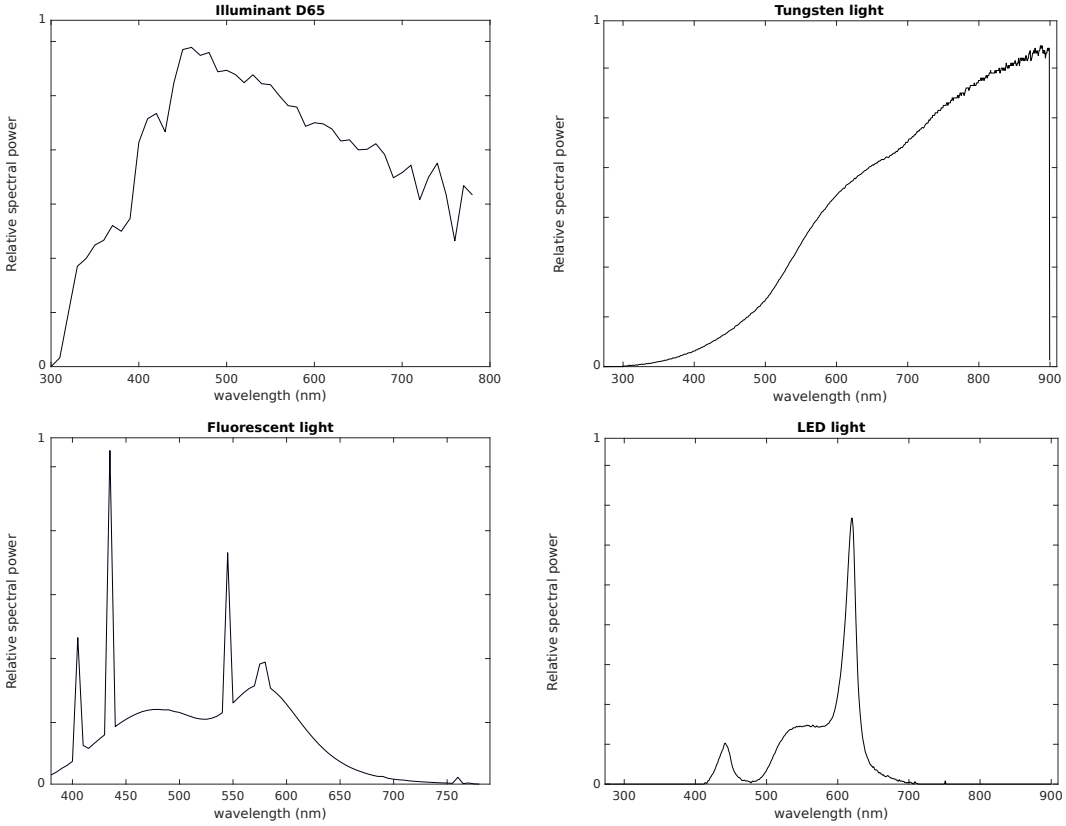


Figure 2.3: Spectral power distribution of various common types of illuminations: from left to right, top to bottom, sunlight (CIE standard D65), tungsten light, fluorescent light, and light-emitting diode (LED).

that is incident to the object at every wavelength, while the spectral reflectance (or transmission) dictates what fraction of that radiation is reflected (or transmitted) and will then reach the eye of the viewer. Under the assumption that objects present Lambertian surfaces, the *radiance* of an object can be described as the product of the spectral power of the illumination and the spectral reflectance of the object, mathematically denoted as $S(\lambda)$

$$S(\lambda) = I(\lambda)R(\lambda) \tag{2.1}$$

This model is an approximation that does not model all the kinds of possible rough surfaces, but it is often good and frequently used when the characteristics of the surface are unknown.

Finally the last element of the human visual system is the eye. The eyes

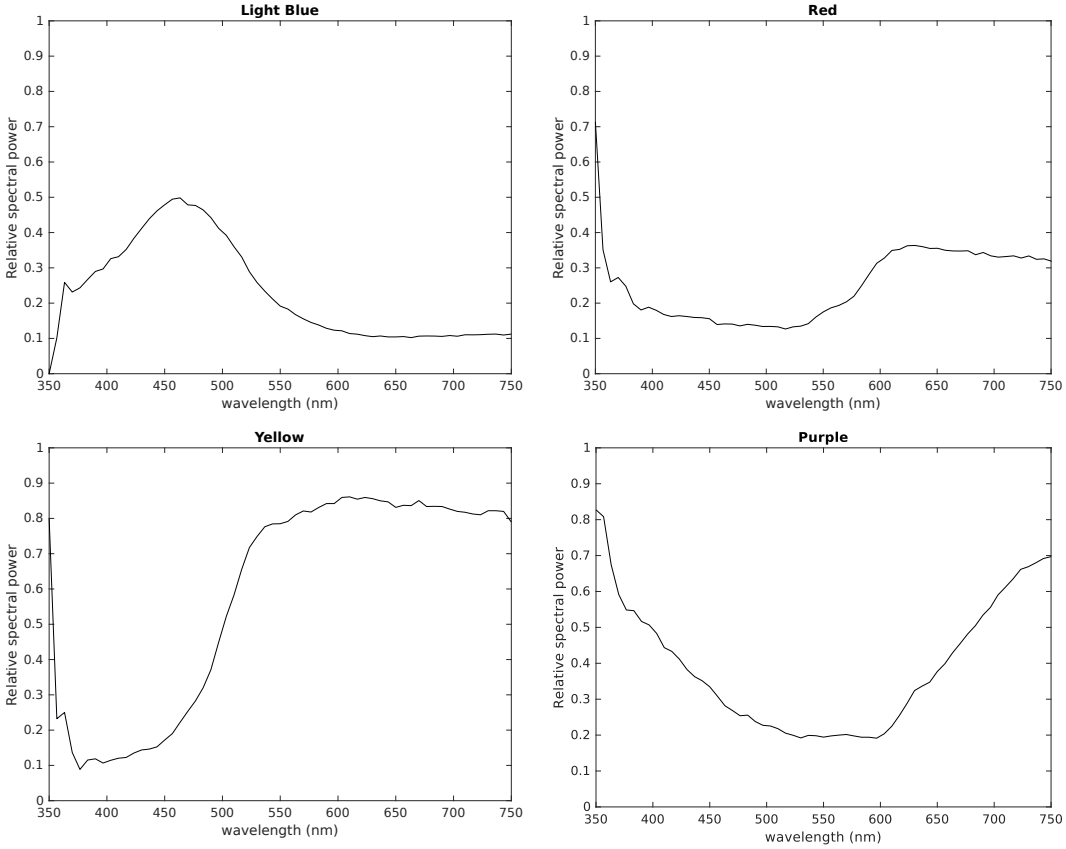


Figure 2.4: Spectral reflectance of different color patches, under D65 illuminant. Color spectral distributions taken from Munsell Color table.

have two different kind of receptors: the rods, which main aim is to give an overall picture of the scene, and the cones, which are responsible for the perception of colors. They help us resolve fine details in images, and are responsible for photopic, or bright-light, vision. The cones are divided in three different types:

- L-cones which have peak sensitivity towards the long wavelength section of the visible spectrum,
- M-cones which have peak sensitivity towards the middle wavelength section of the visible spectrum, and
- S-cones which have peak sensitivity towards the short wavelength section of the visible spectrum.

These three types of cone together gives humans the sensation of color vision. The spectral sensitivity responses of the L-, M-, and S-cones respectively are denoted as $l(\lambda)$, $m(\lambda)$ and $s(\lambda)$. Cones sensitivity to different wavelength is shown in Figure 2.5. It is interesting to observe that cones areas do not cover disjoint sections of the visible spectrum, nor do they cover it entirely. In fact, the responses of L-cones and M-cones overlap significantly, and all three curves show low response to stimulus below around 400 nm and above around 650 nm.

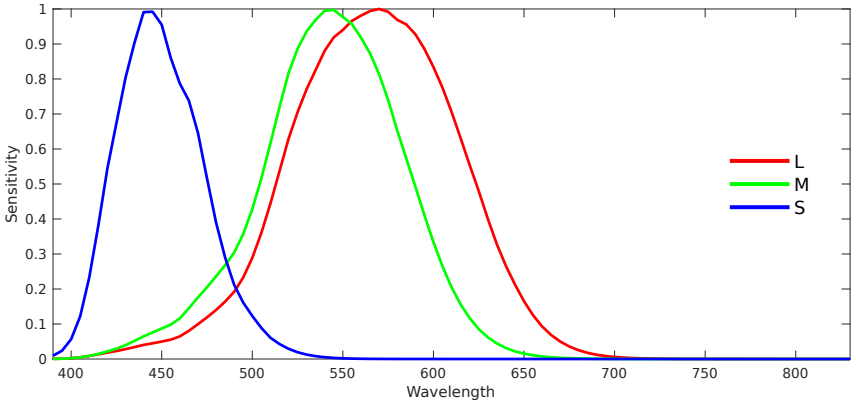


Figure 2.5: In the eyes, three type of cones are present, namely called Long, Medium and Short cones (L, M, S). Each type of cone is sensitive to a different range of the wavelength; in relation to the way those cones are stimulated by the light reaching the eye, a certain color sensation is formed in human brain.

Under a fixed set of viewing conditions, the response of these cones can be accurately modeled by a linear system defined by the spectral sensitivities of the cones. When an object with spectral power distribution $S(\lambda) = R(\lambda)I(\lambda)$ is observed, each of the three cones responds to the stimulus by summing up the reaction at all wavelengths. The response of the receptors can be mathematically expressed as the triplet (L,M,S), called *trichromatic response*.

$$\begin{aligned} L &= \int_{400}^{700} l(\lambda)R(\lambda)I(\lambda)\ d\lambda \\ M &= \int_{400}^{700} m(\lambda)R(\lambda)I(\lambda)\ d\lambda \\ S &= \int_{400}^{700} s(\lambda)R(\lambda)I(\lambda)\ d\lambda \end{aligned} \tag{2.2}$$

This representation of the human eye response to the stimulus have an important consequence in digital camera, which is that to represent color information only three values per pixel are needed.

If a standardized set of cone responses is defined, color may be specified using a three-value vector, known as a tristimulus vector [151]. Because the cone responses are difficult to measure directly, but nonsingular linear transformations of the cone responses are readily determined through color-matching experiments, such a transformed coordinate system is used for the measurement and specification of color.

2.2 Image in digital camera

The formation of the image in the camera sensor follow the same steps presented in section 2.1, with obvious difference in the final element of the chain, involving a digital sensor instead of the human eye receptors.

Analogously to the human visual system, the image captured by a camera sensor can be represented as ρ, a function depending on three physical factors: the illuminant spectral distribution $I(\lambda)$, the reflectance properties of the surface where the light collides $R(\lambda)$, (exactly as defined in the previous section) and the sensor spectral sensibility $C(\lambda)$. Given this notation is possible to define the sensor response at position (x, y) as:

$$\rho(x, y) = \int_{\omega} C(\lambda) Radiance(\lambda) d\lambda \tag{2.3}$$

which, under Lambertian surface assumption, can be rewritten by splitting *Radiance* as the multiplication of reflectance and illuminant spectral distribution as:

$$\rho(x, y) = \int_{\omega} C(\lambda) R(\lambda) I(\lambda) d\lambda \tag{2.4}$$

where ω is the wavelength range of the visible light spectrum (380 to 780 nm), ρ and $C(\lambda)$ are K-component vectors, where K is the number of spectral bands acquired by the sensor.

In consumer still-camera and video applications, color images are typically obtained via a spatial subsampling procedure implemented as a Color Filter Array (CFA), a physical construction whereby only a single component of the color space is measured at each pixel location. Digital Still Cameras (DSC)

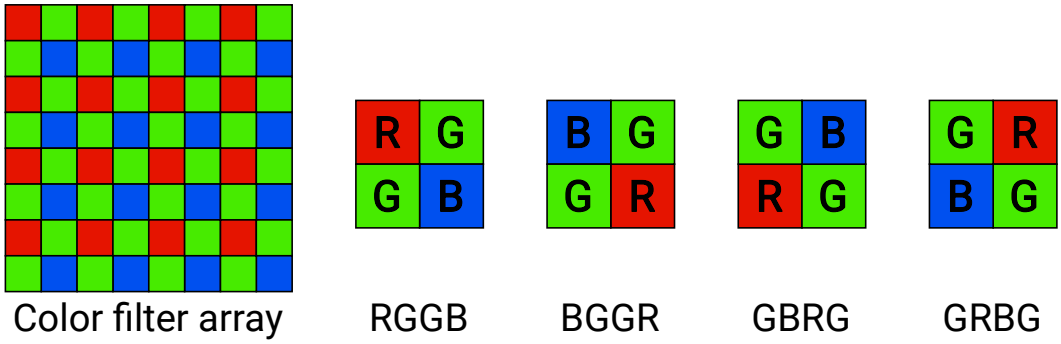


Figure 2.6: Different possible CFA configurations. The disposition of the color filters is an essential information for camera processing of the raw signal.

typically acquire images that are made of three different channels: the sensors are designed in order to collect the color components corresponding to red, green and blue components of the light signal. Photo-receptors of this kind of sensors (tristimulus value based) are disposed in the CFA in different possible configurations, such as the one called called Bayer pattern [20]. In Figure 2.6 is depicted the bayer pattern with possible color filter disposition. The three component vector $\rho = (\rho_1, \rho_2, \rho_3)$ is referred as sensor or camera raw $RGB = (R, G, B)$ triplet.

In the vector space of digital camera, the reflectance spectrum $R(\lambda)$ is sampled uniformly in the range $[\lambda_{min}, \lambda_{max}]$: this sampling of the reflectance spectrum leads to a discrete version of equation 2.4, that can be written as:

$$\rho(x, y) = R(x, y)IC \tag{2.5}$$

If noise n and non-linearity $\mathcal{N}$ coming from the system are taken in consideration the equation can be further extended to:

$$\rho(x, y) = \mathcal{N}(R(x, y)IC + n) \tag{2.6}$$

The noise considered in this formulation is additive but it can also be multiplicative or sensor dependent, making the formulation much more complex. The complex formulation is usually not modeled for implementations of commercial cameras, so will not be considered in this thesis [18].

An image can be considered as a 2-dimensional array, on M rows, N columns and K spectral bands (usually called channels). Each entry of this array is a K-channel vector pixel formed according to the model represented

in equation 2.6. The value of the pixel is given by the reflectance spectrum of the objects in the 3-dimensional scene, illuminated by the illuminant present in the scene. If we denote as f the representation of the full-color image in which each pixel is formed according to equation 2.6, we can further model the image formed on the sensor as:

$$g = \mathcal{B}\{Hf\} \tag{2.7}$$

where $\mathcal{B}$ is the sensor color filter array (CFA) and H is the point spread function (a blur effect) related to the optical system.

Given the description of how the image information is modeled and captured inside a digital camera, the next chapter will describe the different steps that come after the acquisition to process the captured information to obtain a final colored image.

Chapter 3

Digital camera pipeline

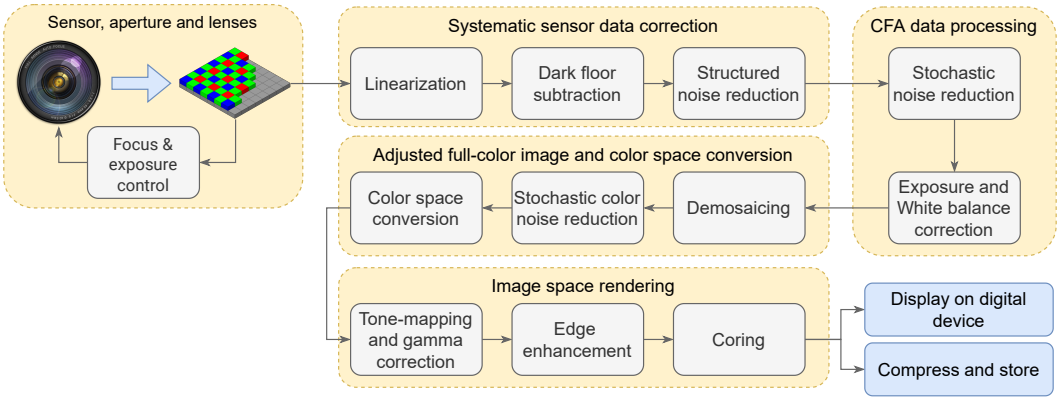


Figure 3.1: Typical DSC processing pipeline. This chain of operators can change by camera manufacturer to manufacturer.

The raw data acquired by a camera sensor, based on the model defined in equation 2.6, pass through different steps of processing before becoming a final full-color image. The computation chain inside of a digital still color camera (DSC), also called *Camera Processing Pipeline*, is made of different individual computation blocks that can be ordered in a tons of possible permutations. From one camera manufacturer to others, this pipeline can substantially change in the order and presence of operations and in the associated implementation. However, even if this problem seems to lead to infinite possible solutions, the design of this camera pipelines is constrained by two important factors: image quality must be maximized while compute resource use must be minimized. It is the minimization of required computational effort that, in

fact, severely restricts the number of degrees of freedom in the image processing chain design problem. Consequently, image processing operations that are highly effective may not be viable candidates for image processing chain for constrained compute environments.

A basic DSC processing pipeline is shown in Figure 3.1. This pipeline will be described piece by piece in the following of this section and represents the baseline of the work presented in this thesis. This structure has been used for the identification of possible limitations and bottlenecks presents in a hypotetical pipeline, in order to design and develop new and more efficient blocks to improve the image quality of existing DSC camera processing pipelines.

3.1 Sensor, aperture and lenses

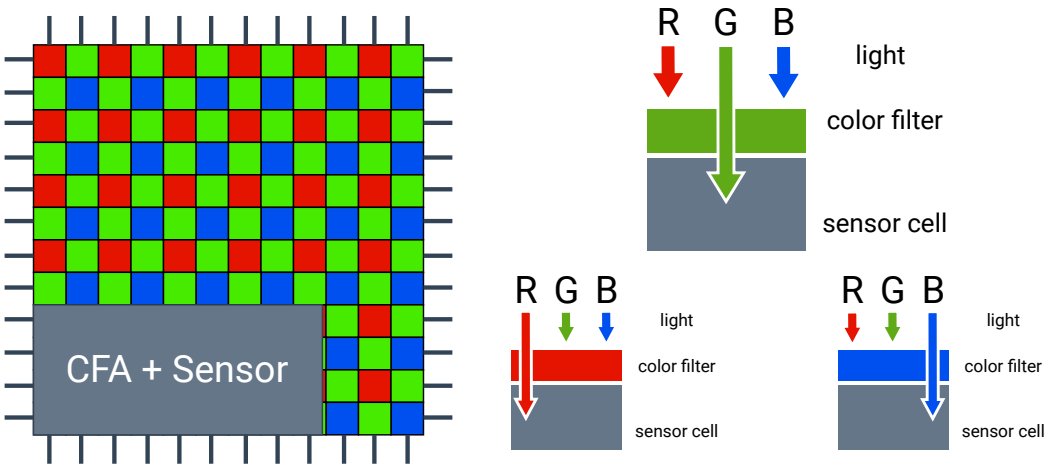


Figure 3.2: Image sensor covered by a Bayer color filter array and the concept of acquiring the visual information using color filters. Image from [120].

As introduced in section 3, cameras are designed to capture a tristimulus (R,G,B) signal, by acquiring the information related to red, green and blue wavelength ranges information of the light reflected by objects in a scene. To perform such operation a set of three different sensors should be needed. Instead of building cameras with three different sensor, one for each wavelength, camera manufacturers designed sensors with CFA places on top of sensor elements. The most commonly used CFA configuration is the Bayer pattern. In Figure 3.2 is depicted the concept of acquiring light information using sensor

cells with the CFA applied on top: the corresponding filter let the interested wavelength to be acquired by the sensor. The RAW data collected by the sensor will be structured with respect to the CFA applied.

In order to take a picture, control system interact with the sensor to determine the exposure and the focal position of the lens. These parameters needs to be determined dynamically based on the content of the scene.

Exposure control usually requires characterization of the brightness of the image: an over- or underexposed image will greatly affect output colors. In relation to the amount of light measured in the scene, the exposure control system changes the aperture size and/or the shutter speed to take a picture with the right amount of light needed to obtain a well-exposed image. Outdoor as well as indoor images taken with typical cameras can suffer from the problem of limited dynamic range, in the case the scene have an excessively backlit or frontlit. Dynamic range refer to the contrast ratio between the brightest and the darkest pixel in the image. The human visual system (HVS) can adapt to about four orders of magnitude in contrast ratio, while the sRGB system and typical computer monitors and television sets have a dynamic range of about two orders of magnitude. This limit leads to spatial details in darkest area indistinguishable from black and spatial details in brightest areas from white. This problem has been addressed in research in different ways: the most intuitive approach is the one of capturing multiple pictures of the same scene with different exposure values, and then combine the information in order to obtain an high dynamic range (HDR) image [16].

Focus control can be performed with two types of approaches: active and passive approaches. Active approaches typically use a pulse beam integrated in the camera, near the lens system, called auto-focus assist lamp. This pulse signal is used to estimate the distance of objects from the camera in order to adjust the camera focus plane. Passive approaches are based on the information present in the picture, such as spatial information, to determine the focus of the scene.

A drawback in cameras that use only one sensor to capture three-channel RGB images is the presence of aliasing. Aliasing introduces highly unpleasant artifacts in the final images captured by the cameras and is in general dependent on factors such has the CFA pattern used in combination with the demosaicing process and the sampling resolution of the sensors. In order to reduces those kind of artifact, usually camera manufacturers design cameras with anti-aliasing filters. Those filters have the objective of reducing the Moiré patterns that may occur due to the sampling process involved.

3.2 Systematic sensor data correction

After acquiring raw sensor data, the next step is the processing of this data in order to obtain a full-color image. The data collected by the sensor can be affected by different kind of noise and artifacts due to hardware design limits or issues that can occur in the sensor. For example, one first common step performed by cameras is the defective pixel (also called dead pixel) correction, a step that aim to fix missing information due to the presence of defective sensor cells that may not record information during the process described in section 3. This is just one of the processing blocks that are present in this first group of operators.

3.2.1 Linearization

In equation 2.6 we introduced the symbol $\mathcal{N}$ to model non-linearities coming from the camera image capturing system. These non-linearities are typically due the electronics involved and while most of the sensors in the market adopted by camera manufacturer have a linear response, in some cases is still necessary a step of data linearization. This process is performed in the camera by using an Opto-Electronic Conversion Function (OECF) that maps the input nonlinear data to a linear space. This correction transforms the raw measured data (typically with an 8-bit precision) into linear space (of higher bit precision, typically 12 or 16-bit). The linearization process corresponds to the inverse of $\mathcal{N}$ in equation 2.6.

3.2.2 Dark floor subtraction

An assumption that is often made is that a sensor cell receiving no light (e.g. with the camera cap on the lens) will register a value of zero, corresponding to black. Unfortunately, this assumption is not real since due to the thermal noise and other kinds of noise nonphoton related coming from the electric nature of the sensor, in no light conditions produces nonzero values. This noise is one contributing component to the term n in equation 2.6. To account this problem, a *dark floor* value is calculated for the specific camera, and is subtracted from the collected CFA image. This is the first step in the camera pipeline, after the linearization process. This process corresponds to the subtraction of a constant value from each pixel collected value; alternatively a spatial-dependent set of value can be subtracted from the collected image. In

general, to avoid data clipping and quantization errors in shadow regions, the subtraction operation is done in order to not remove completely the undesired bias in the pixel data, but instead some residual biases are maintained.

This approach of subtracting the *dark floor*, using a value computed before, can be achieved using a "lens cap shot" or the capture of a good-quality matte black test card. Alternately, if the sensor has shield pixels around its perimeter, these values can be interrogated at the time of capture. This last approach has the added benefit of characterizing the dark floor at the actual circumstances of the camera at the time of capture. It should be noted that dark floor is a function of exposure index (ISO), shutter time and the ambient temperature.

3.2.3 Structured noise reduction

The assumption that the dark floor is constant across the entire image capture is a simplification that is valid only if the user expectation is suitably lax enough. There are many potential causes to why the dark floor is not constant all across the extent of the sensor, apart from physical flows of electronic components. For example the proximity of the sensor to heating parts of the camera will warm one part of the sensor, producing non uniform dark floor noise. At this point the problem is now to subtract a *dark floor mask* from the collected CFA image. This mask is created either in the factory or during a dark field shot (i.e., closed shutter) that may be automatically captured during power up of the camera.

Dark floor subtraction operation tends to address low-frequency structured noise better than high-frequency structured noise. One of the most significant type of high-frequency noise is the defective pixel. Usually a sensor presents a certain number of defective pixels: whole columns and rows of pixels might be nonfunctional. Even if the sensor has been picked in factory as a defective free one, over time it is possible that pixel became defective. Those pixels can be divided in two categories: the ones which are completely nonfunctional, leading to zero values when light impact to the corresponding cells, and the ones which are partially functional, which may still respond to light, but with a significantly different gain factor from the majority pixel population of the sensor. This second type of defective pixels is more problematic with respect to the first one. Defective pixels of the first category can be easily mapped out in the factory and their locations stored in the camera firmware. Defective pixels of the second category or those of the first category that are formed after the camera has left the factory are more difficult to address. Since the

identification of defective pixel is a particularly difficult task, one may be forced to treat all unidentified defective pixels with stochastic noise cleaning methods [160, 164, 3]. However, the main method of defective pixel masking is to replace defective pixel values with the average values from known working neighboring pixels.

A final class of structured noise to be discussed deals with variations in the thickness of the CFA color filters across the surface of the sensor. These are usually a consequence of flaws during sensor construction process. As a result, an image of a featureless neutral field may exhibit low-frequency variations in color. Because this is a stable phenomenon, it can be mapped in the factory and stored in the camera firmware. The correction is performed like the dark floor subtractions, with the difference that in this case each channel has its own separate mask.

3.3 CFA data processing

In this stage of the camera processing pipeline, the focus is on the reduction of stochastic noise and correction for exposure and white balance errors that may occur during the time of image capture. These two operations can be in arbitrary order in the pipeline since can be considered independent. This is principally related to the fact that stochastic noise reduction is mainly concerned with the high-frequency spatial component of the image data while the exposure and white balance correction will be focused on using the low-frequency spatial component to the image.

3.3.1 Stochastic noise reduction

In the camera processing pipeline, most of the operators act as signal amplifiers. For example, looking at Figure 3.1, color correction, tone scale and gamma correction, and edge enhancement are signal amplifier operations. Moving further back along the processing pipeline, CFA interpolation (Demosaicing) operation can act as a signal amplifier. In addition, this operation maybe linear or adaptive (non-linear), in which case the robustness of the algorithm could be affected by the presence of the noise in the CFA image data. The noise reduction is then performed before the demosaicing operation.

Due to the fact that for each pixel only one color channel value is available, it is particularly difficult to exploit the partial correlation between the color channels of the images. For this reason stochastic noise reduction is often

achieved by using single-channel grayscale image processing techniques (such as low-pass filters, sigma filtering [107], and median filtering), and a different stochastic color noise reduction operation is performed after the demosaicing operation as shown in Figure 3.1. Conceptually, the CFA image data is split into three or more color channel components by collecting pixels of like color into each component, as explained in section 3.1. At this point, each component can be treated as an individual grayscale image and treated for noise reduction in the preferred way. After noise reduction, the components can be recombined together in order to obtain the original CFA image data configuration.

3.3.2 Exposure and white balance correction

The human visual system is able to constantly and automatically adjust the apparent exposure and white point of what it sees, while digital cameras do not have such innate functionality: such adjustments must be performed algorithmically. The goal of such algorithms is to render neutral areas in the scene as regions of equal code values for all color channels in the final image. Sometimes the processes of exposure correction and white balance correction are referred to collectively as *scene balance correction*.

These adjustments can be divided in two groups: adjustments in response to user inputs and automatic adjustments based on collected data. In the first case, the user can specify a specific exposure compensation (measured in stops) and a specific scene illuminant (e.g. daylight, tungsten, neon etc...), and then, with this specific information the CFA image can be directly modified. For exposure, all the pixels values will be equally modified by the appropriate scale factor. For white balance correction, there would be a set of three scale factors, each for one of the channels.

In the second case the only data available is the one collected by the camera while capturing the image. In this second scenario the processing is performed by automatic exposure (AE) and automatic white balancing (AWB) algorithms [68, 163]. The way the CFA image is processed is the same as done by the algorithms in the first group, but in this scenario the scale factors (one for the exposure correction, three for the white balance correction) are estimated by those automatic algorithms. In the case of AWB, most of the simplest approaches are based on heuristic statistical models such as the gray world hypothesis [169].

3.4 Adjusted full-color image and color space conversion

In this stage of the digital processing pipeline, the CFA image has already been treated in order to reduce structural and stochastic noise, has been properly white balanced. From now on, the processing steps will assume these conditions.

In this block the CFA image is converted into a full-color image and it will then converted into a known, calibrated color space.

3.4.1 Demosaicing

Full-color image means that each pixel in the image has a color specification triplet. The process for the creation of this kind of image, starting from the CFA image data is called *demosaicing*, or *CFA interpolation*. Demosaicing is, by far, the most computationally intensive step in the processing pipeline.

There are two general approaches to the problem of CFA interpolation. The first is to use standard linear interpolation methods. The most common approach is to combine neighboring pixel values of the same color in some straightforward method to produce an estimate for the missing pixel value. This method can take the form of a convolution operation and implement such standard practices as pixel replication, bilinear interpolation, or bicubic interpolation. If, on the other hand, there is some understanding of the cross color channel correlation of the data, more than one color may be used in this process [48].

The second approach to CFA interpolation is to use nonlinear adaptive methods. With these systems, the segmentation of image data into luminance and chrominance channels becomes more important because the decisions made by the algorithm are generally keyed off the fine spatial detail in the image. As a result, the luminance channel is interpolated first by using some form of edge detection of the luminance data to determine the precise manner of interpolation from pixel to pixel [97, 98, 167, 84]. Once the luminance channel is fully populated, the chrominance channels are generally treated with the linear approaches previously described.

The result of the demosaicing operation is a full-color image, in a "camera" color space. The generated color space, in general, do not correspond to a standard calibrated color space, but instead to a sensor depended color space: it is defined by the spectral sensitivities of the camera image capture hardware.

This problem will be addressed in the color space conversion step, but first, another denoising step for the full-color image is necessary.

3.4.2 Stochastic color noise reduction

During the stochastic noise reduction applied to CFA image data, the color channels were treated as separate and independent grayscale channels. Now that all the color channels are fully populated, another facet of stochastic noise emerges. A texture that might be acceptable in the context of a single-channel image is deemed not acceptable when matched with similar, but different, textures in the other color channels. In an RGB image the stochastic noise corresponds to color variations in the images. This effect is most pronounced neutral (gray) areas of the images because of the color fluctuations, and is also more noticeable than the light-dark fluctuations in a single-channel image. The purpose of this block in the digital camera pipeline is handle this problem.

The simplest approach may be to, again, treat each color channel as an independent grayscale image and then clean these components separately. However, this may not be overly effective and tends to miss the whole point. It is far better to transform the image into a luminance-chrominance representation (assuming it is not already so), and then, the luminance and chrominance channels can be noise-cleaned in any appropriate manner. Generally, the luminance data will require a significantly different cleaning modality from that used for the chrominance data. If the same method is used, at least the tunings of the operation will be quite different.

Because this is a color noise reduction operation, its is possible that an approach prefers to work only on the chrominance channel to perform the noise-cancelling operations, while letting the luminance information untouched. If there is a reason, the luminance channel can also be noise-cleaned at this time using any method applicable to single channel grayscale images.

3.4.3 Color space conversion

The next step is the one of transforming the image into a standard calibrated color space. There are a number of possible destination color spaces to transform the image in, which are suitable for the most different purposes. The industry has standardized on the *sRGB* [157] color space, which has been designed for video, or soft-display, devices. However, since sRGB is itself a color transform from the CIE 1931 XYZ [47] color space, it's first necessary to con-

vert the full-color image from the camera color space to CIE 1931 XYZ and eventually in the sRGB color space.

CIE 1931 XYZ space (pages 101 to 110 in [87]) is a color space defined by standardized $\bar{x}(\lambda)$, $\bar{y}(\lambda)$, and $\bar{z}(\lambda)$ color matching functions. The first part of the color correction process is to transform the image data from camera color space into CIE 1931 XYZ space. Assuming an RGB camera color space, the operation becomes a 3×3 matrix multiply:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} R_{camera} \\ G_{camera} \\ B_{camera} \end{bmatrix} \tag{3.1}$$

Since the camera color space is strictly related to the specific sensor response to light stimulus, the coefficients of the transformation matrix are computed in the factory through a regression process using measured camera RGB tristimulus values of color patches with known XYZ tristimulus values. Once the XYZ tristimulus values have been computed, they can be transformed to sRGB tristimulus values with a standard matrix as follows:

$$\begin{bmatrix} R_{sRGB} \\ G_{sRGB} \\ B_{sRGB} \end{bmatrix} = \begin{bmatrix} 3.2410 & -1.5374 & -0.4986 \\ -0.9692 & 1.8760 & 0.0416 \\ 0.0556 & -0.2040 & 1.0570 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tag{3.2}$$

Concatenating the two matrices the entire color transformation can be summarized as a single matrix multiplication:

$$\begin{bmatrix} R_{sRGB} \\ G_{sRGB} \\ B_{sRGB} \end{bmatrix} = \begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{bmatrix} \begin{bmatrix} R_{camera} \\ G_{camera} \\ B_{camera} \end{bmatrix} \tag{3.3}$$

where

$$\begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{bmatrix} = \begin{bmatrix} 3.2410 & -1.5374 & -0.4986 \\ -0.9692 & 1.8760 & 0.0416 \\ 0.0556 & -0.2040 & 1.0570 \end{bmatrix} \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

It is important to note that, until this point in the digital camera processing pipeline, all of the processing operations are designed to be explicitly used on linear space data.

3.5 Image space rendering

The last steps of the processing pipeline are targeted at producing the final image to be displayed on a device. This means preparing the image to be displayed on a visualization device or compress and stored in a digital memory. The step of color space transformation, discussed in the previous section, is the first step in this direction. The process continues with the transformation of the image in a non-linear space, suitable for video devices. After that, there are different proprietary steps, aimed at image appearance enhancement: edge enhancement or sharpening and coring. These techniques are mostly heuristic based and require considerable fine-tuning.

3.5.1 Tone-mapping and gamma correction

As has been discussed in section 2.1, the human visual system's ability to adapt to a wide range of scene luminances. This characteristic of the HVS, as for the compensation of scene illuminant, is not a innate functionality of digital cameras, and so must be duplicated into cameras algorithmically. In this specific case, the overall contrast of the scene must be adjusted so that the image as viewed on the soft display device looks similar to the original scene viewed under illumination that was typically a hundred times as bright, if not more. Added to this, the image data must be transformed to account for the nonlinearity of the video display.

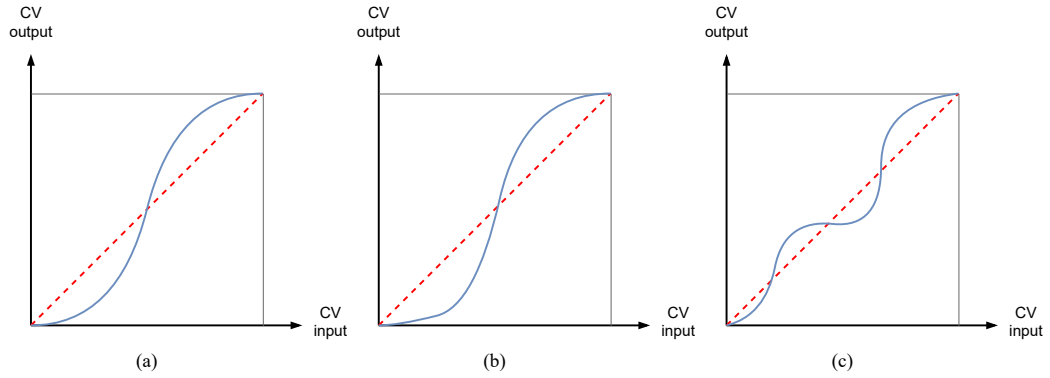


Figure 3.3: Examples of tone-mapping functions: (a) idealized S-curve function, (b) function with suppressed shadow response, (c) scene specific function. Image from [120].

The first operation, the *tone mapping* (or scaling), adjusts the contrast of the image. It is usually implemented as a fixed lookup table that is applied equally to the red, green, and blue channels. It assumes the input data is in a linear space that has been properly exposure corrected. There are generally two classes of tone mapping functions: the first class consists of fixed transforms that are installed in the factory and used on all images in the same way, the second consists of tone scale transforms that are generated dynamically on an image-by-image basis. There may be a single transform or a small family of fixed transforms, with each family member assigned to a different exposure compensation step. The shape of the fixed transform curve is typically "S" shaped as showed in Figure 3.3 [87]. The second class of functions are based on the consideration that the dynamic range of HVS is significantly greater than any current digital camera. A partial solution to this dilemma is to create custom tone scale functions that render both shadows and highlights at the expense of the midtones, which are visually less important in high dynamic range scenes. There are many ways of algorithmically producing such a tone scale based on histogram analysis of the image [78, 69]. The final tone mapping becomes a simple point transformation of the image data:

$$\begin{cases} R'_{sRGB} = T(R_{sRGB}) \\ G'_{sRGB} = T(G_{sRGB}) \\ B'_{sRGB} = T(B_{sRGB}) \end{cases} \tag{3.4}$$

The second operation is the *gamma correction*. This standard transform is also defined in the sRGB specification [157] and accounts for the fundamental photometric nonlinearity of the displays. This transform is essentially a simple power relationship:

$$X''_{sRGB} = \begin{cases} 12.92X'_{sRGB} & for X'_{sRGB} \leq 0.00304 \\ 1055X'^{(1/2.4)}_{sRGB} - 0.055 & for X'_{sRGB} > 0.00304 \end{cases} \tag{3.5}$$

where X'_{sRGB} is R_{sRGB} , G_{sRGB} , or B_{sRGB} normalized to $[0, 1]$.

Equation 3.5 is a point transform and can be concatenated with the tone scale correction to produce a final single point transform to perform both operations simultaneously:

$$X''_{sRGB} = V(X'_{sRGB}) = V(T(X_{sRGB})) = G(X_{sRGB}) \tag{3.6}$$

3.5.2 Edge enhancement

The essential purpose of edge enhancement (also called sharpening) is to amplify the high-frequency spatial components of an image to make it look sharper. Because noise has also high-frequency characteristics, attention must be given to the question of controlling noise amplification during edge enhancement.

The two main approaches to edge enhancement are *direct convolution* and *unsharp masking*. The direct convolution method consists of extracting a high-frequency record from the image via convolution with a high-pass kernel. Some scaled amount of this high-frequency record is then added back to the original image to produce the sharpened result:

$$A' = A + k(A * h) \tag{3.7}$$

where A is the original image, h is the high-pass convolution kernel, k is a scale factor, and A' is the resulting sharpened image.

In the case of unsharp masking, the high-frequency record is created by computing the difference between the image and a blurred (low-pass) version of itself:

$$A' = A + k(A - A * b) \tag{3.8}$$

where A is the original image, b is the low-pass convolution kernel, k is a scale factor, and A' is the resulting sharpened image. These two operations produce mathematically equivalent results when confined to the world of linear shift-invariant systems.

3.5.3 Coring

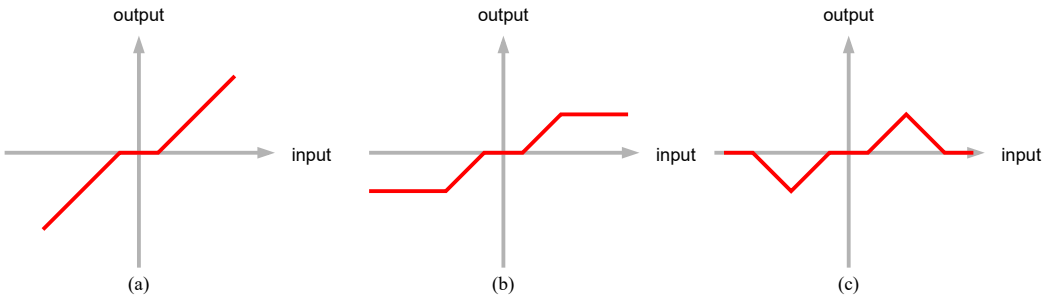


Figure 3.4: Examples of coring functions. Image from [120].

In order to control noise amplification during the edge enhancement process, the high-frequency record needs to be noise-cleaned in some manner prior to being added back to the original image [4, 83]. A *coring function* is an amplitude noise cleaning operation and usually is used to noise-clean the high-frequency record of the edge enhanced image. This function is a point operation that, like the previously described tone scale correction, is usually implemented as a lookup table. Examples of coring functions are depicted in Figure 3.4. The shape of the coring function is heuristically determined based on the fundamental noise characteristics of the digital camera system. Modifying Equations 3.7 and 3.8, coring operation, $C(\cdot)$, can be added:

$$A' = A + kC(A * h) \tag{3.9}$$

$$A' = A + kC(A - A * b) \tag{3.10}$$

Chapter 4

Artifacts in digital images

In section 3, an overview of the general digital camera pipeline has been presented, giving a general idea of how the processing operations inside a camera are combined in order to obtain a final sRGB image. However, the images that are obtained by applying a certain camera pipeline starting from RAW sensor data are not necessary free from problems. In fact, a certain camera pipeline, as already underlined during the description of some of the pipeline blocks in section 3, could introduce noise and artifacts in the resulting sRGB images. This can be related to multiple aspects: the relation between the shooting conditions and the parameters of the algorithms used in the camera pipeline, the kind of methods used in the pipeline and the way those methods are ordered. Considering the basic operations that have been discussed previously, such as demosaicing, those operations may introduce some sort of noise or artifacts in the processed image. In some cases the presence of these kind of noise elements can be accentuated by the shooting conditions. If we consider the example of a photo taken in low-light conditions, the final image will probably suffer from sensor noise, due to the high ISO values and long exposure time used for taking the image. If the camera pipeline blocks do not take in consideration the specific scenario in which the photo has been taken, the extra noise generated by the shooting conditions will not be removed, giving a non optimal result. This is the case in which the parameters of one of the blocks are not optimal for the specific case.

Another problem can be the presence of elements in the scene that can affect in some way the final sRGB image. Those elements are not related to the behaviour of the camera pipeline elements or hardware limits of the cameras, however they can reduce the overall image usability, more likely in

situations where the image will be used by other automatic systems. This is the case of images taken during hazy days or in underwater scenario. From this example came out also the necessity to have a specific definition of quality in relation to the final use of the images, which may vary in relation to a human user or to automatic systems, and from one image analysis system to another.

In this section a taxonomy of artifacts and problems that can occur in digital images is presented. This taxonomy is divided in two main classes, as already introduced, in relation to the origin of the artifacts: in camera and external image artifacts. A definition of image quality is also provided in order to be able to define in which terms the images are considered problematic or not, with the purpose of identify the possible improvements that can be done in the camera digital processing pipeline, or in the post-processing steps.

4.1 In camera image artifacts

In the first group of artifacts are considered all the ones that came from the operations performed in the digital camera pipeline. Here are reported artifacts coming from the application of the algorithms in an non optimal scenario, where the general application does not suit the specific case, and the one coming from flaws in the camera hardware or software application.

4.1.1 Image Noise

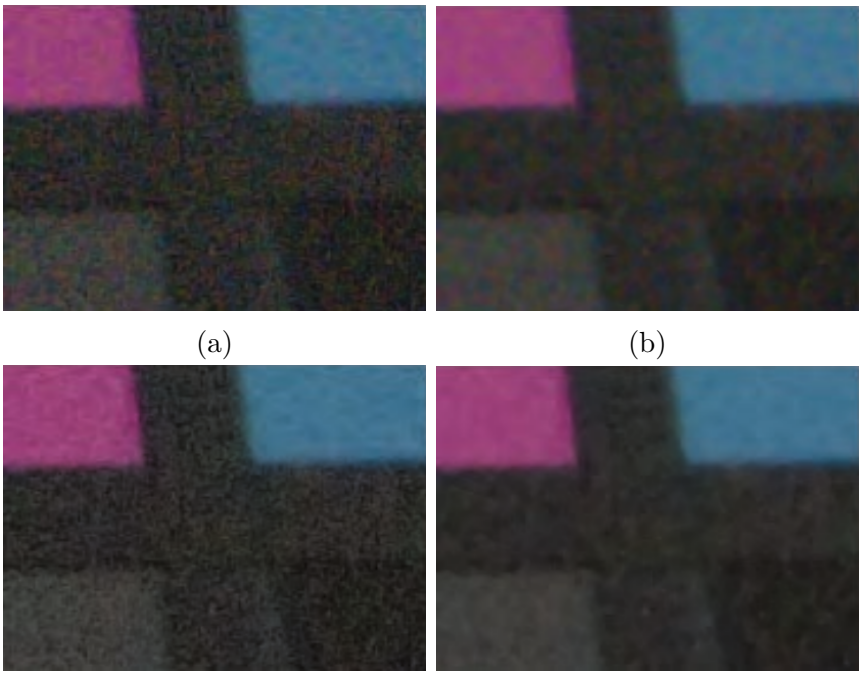
Noise in digital camera images usually appears as random speckles in otherwise smooth regions, altering both tone and color of the original pixels. Typically, noise is caused by random sources associated with quantum signal detection, signal independent fluctuations, and inhomogeneity of the responsiveness of the sensor elements. Noise increases with the *sensitivity* (ISO) setting in the camera, length of the *exposure time*, and *camera temperature*. One example is the case of low-light photography: to increase visibility in general sensitivity and exposure time parameters are set to high values, leading to pictures with a lot of noise. The level of noise also depends on characteristics of the camera electronics and the physical size of photosites in the sensor. Larger photosites usually have better light-gathering abilities, thus producing a stronger signal and higher signal-to-noise ratio. Noise can be seen as fluctuations in intensity (*luminance*) and color(*chromaticity*), and can be handled separately in the luminance and chrominance domain. The generally adopted way to model this kind of noise is the additive noise model [18]. In the additive noise model,

each pixel of the ideal image is contaminated by a random value drawn from a certain underlying noise distribution Z_d; this random quantity adds to the original ideal signal, generating the noisy observed image $N(x, y)$:

$$N(x, y) = I(x, y) + \eta(x, y) \tag{4.1}$$

The term $\eta(x, y)$ which is added to the ideal value $I(x, y)$ is generated by the contribution of many overlapping noise sources. Because of the central limit theorem, a common assumption is to model the contribution of all noise sources as zero-mean *Additive White Gaussian Noise (AWGN)*. Eventually, the noisy term N(x, y) is then observed and recorded.

In Figure 4.1 are shown a noisy image and three de-noised version. As can be seen removing color noise can be done without any big problem (Figure 4.1c), while suppressing luminance noise can result in unnatural looking images and excessive blur.



(a) (b)

(c) (d)

Figure 4.1: Cropped parts of a color checker image captured with ISO 1600 setting: (a) captured noisy image, (b) luminance noise suppression, (c) color noise suppression, and (d) both luminance and color noise suppression.

4.1.2 Demosaicing artifacts

One of the most important steps in the camera pipeline previously described is the demoisaicing operation, which consents to obtain a full-color image starting from the CFA image data. The interpolation process, as described in section 3.4.1, it is used to merge the information coming from the sensor with the CFA pattern applied, and to obtain for each pixel an RGB triplet which represents the RGB stimulus for each pixel of the final image. As already described, there are different possible way to approach the task of CFA image data interpolation, and each approach leads to a similar but different final result. The final full-color image can be affected by the presence of distortion, due to the algorithm adopted in the interpolation process. The color patterns that can occur are called *zipper effects*. Figure 4.2a shows an example of these kind of artifacts. These effects can usually be seen along abrupt edges when pixels from both side of an edge are used in demosaicing.

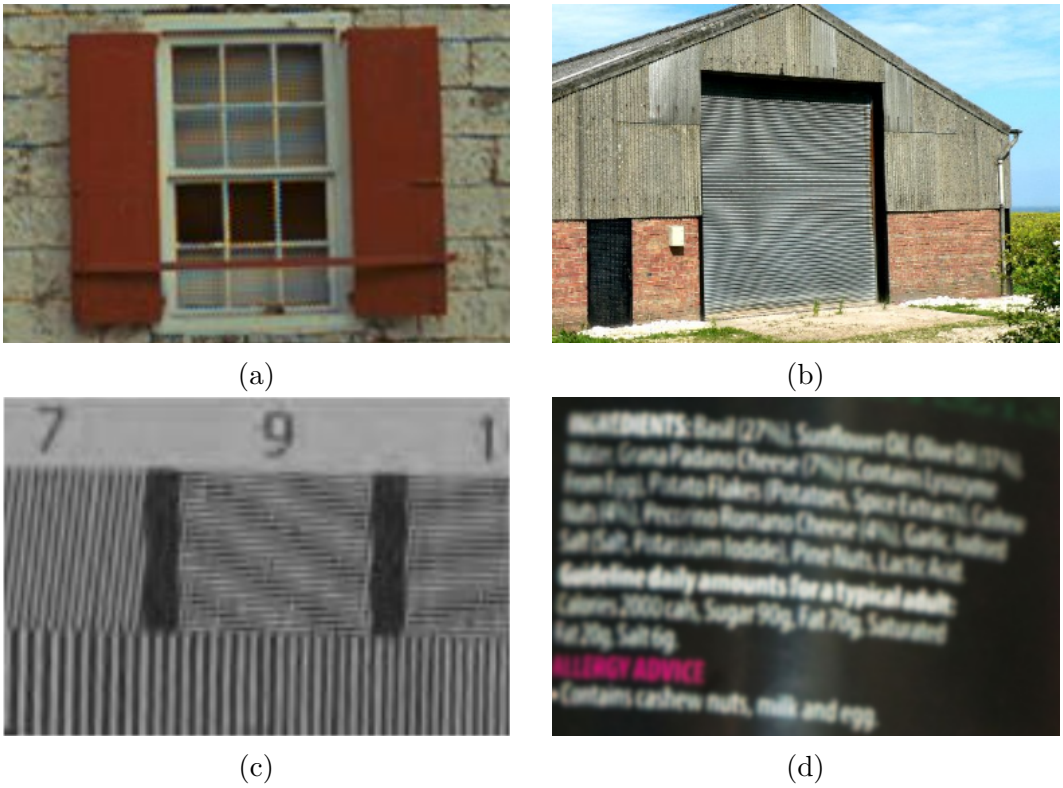
In addition to zipper effect, another kind of artifacts related to the demosaicing process are *aliasing* artifacts or *color moiré* patterns. These artifacts usually constitute large, visually annoying regions, as can be seen in Figure 4.2b and 4.2c. These artifacts cannot be therefore removed using traditional low-pass filters which rely on local image characteristics. Aliasing artifacts appear in areas where the resolution limit of the sensor has been reached and where color sampling prevents correctly detecting orientations of edges in an image. This is particularly true in fine texture regions, where aliasing artifacts often take the form of repeating patterns of false colors.

Finally the last kind of problem that can come from this step in the camera processing pipeline is the *blur*. This problem corresponds to an apparent resolution loss and is generally caused by demosaicing with insufficient edge-preserving characteristics. An example is shown in Figure 4.2d.

Demosaicing artifacts vary in their characteristics, appearance and size. Considering the complexity of the problem of reversing color sampling in areas with difficult structural content, demosaicing artifacts may never be fully avoided in real-life situations. Therefore, many digital camera designers focus on achieving trade-offs between noise, image sharpness, demosaicing artifacts and processing time rather than emphasizing any of these issues.

4.1.3 Coloration shifts

Image sensors are calibrated for certain light characteristics. Whenever an image is shot under light of a different color temperature from those for which



(a) (b)

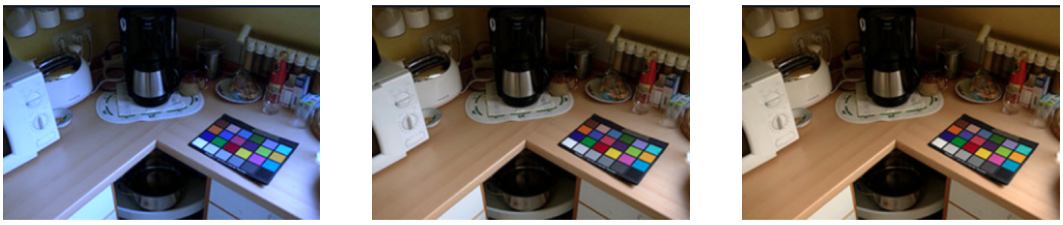
(c) (d)

Figure 4.2: Typical demosaicking defects: (a) zipper effects, (b) and (c) aliasing artifacts, and (d) blur effects.

sensors were calibrated, the image coloration is shifted from the perceived coloration of a scene. This is well observable in the case of neutral (i.e., achromatic) colors, particularly white, which is one of the most recognizable colors due to high and approximately equal contributions of all three color primaries. The main difference with color shifts from noise and demosaicing artifact, is the global impact on the entire image, while the previously presented artifacts are local variation in the images. The white balancing block, discussed in section 3.3.2, operates over the input image in order to produce images with natural *color tint*, compensating for unnatural (global) variations in the collected image.

Digital SLR cameras offer to the users the possibility to manually set the white balancing preferences for each shot. Alternatively the color balance can be automatically adjusted using Auto White Balancing (AWB) algorithms. Those algorithms generally exploit scene information to make an estimate of

the illuminant of the scene, and uses this estimate to correct image color tint. Those algorithms can fail in the estimation process leading to images that may look bluish or reddish, respectively called *cold* or *warm*. Those kinds of wrong images are not visually pleasing as the one obtained by manually setting the illuminant information.



(a) (b) (c)

Figure 4.3: Coloration shifts due to different white balance settings: (a) cool appearance, (b) neutral, (c) warm appearance.

4.1.4 Exposure shifts

Another problem that can affect appearance of the captured images is the *exposure shift*. As seen in section 3.1, to control the amount of light that reaches the camera sensor, the camera adjusts the aperture of the diaphragm, the sensor exposure time and sensor sensitivity (ISO) value. By deciding how long to leave the shutter open and how much to open it, the camera (or the photographer in manual mode) controls the period for which the sensor is exposed to the light to collect photons. The other way to control the amount of light perceived by the sensor is by adjusting the sensor sensitivity (ISO) value. High ISO values lead to high light sensor sensitivity and then brighter images. The combination of these three parameters determine the exposure of the image captured. Depending on exposure settings, the appearance of images can range from dark, which is the effect known as *underexposure*, to bright, which is referred to as *overexposure*. The imaging community uses a measure called Exposure Value (EV) to specify the relationship between the aperture (f-number), F, and exposure time, T:

$$EV = log_2\left(\frac{F^2}{T}\right) = 2log_2(F) - log_2(T) \tag{4.2}$$

The exposure value 4.2 becomes smaller as the exposure duration increases, and it becomes larger as the f-number grows. Figure 4.4 shows images of the same scene captured with different exposure settings.

Digital consumer devices make use of ad-hoc strategies and heuristics to derive exposure setting parameters. 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 4.2, 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:

$$EV_opt = EV_pre + log_2(L_pre) - log_2(L_opt) \tag{4.3}$$

The ideal value L_{opt} for each algorithm is typically selected empirically.

These methods, however, often fail in complex scenarios with different subjects having different reflectivity, due to their blindness with respect to the content of the captured scene. After acquisition phase, typical postprocessing techniques try to realize an effective enhancement via global approaches, such as histogram specification, histogram equalization and gamma correction to improve global contrast appearance. However such kind of approaches may fail in hard case scenarios of images, such as images with double-exposure (areas over exposed and areas underexposed in the same image) or some special cases.

4.1.5 Image compression artifacts

The final step of the digital processing pipeline corresponds to the final visualization of the sRGB images. The image processed by the pipeline can be both displayed on a digital visualization device or stored in digital memory. In the latter case, the image is generally compressed in using JPEG compression format, which can reduce the size of original files ten-fold, and for images with solid color backgrounds even more. The amount of compression is defined

(a) (b) (c)

Figure 4.4: Influence of exposure settings on image quality: (a) underexposure, (b) normal exposure, and (c) overexposure.

(a) (b) (c)

Figure 4.5: Impact of JPEG compression at different compression ratio: (a) quality factor 10, (b) quality factor 50, and (c) uncompressed file.

by the settings of the compression algorithm. JPEG compression is a lossy compression algorithm. This means that part of the information is lost with respect to the original RAW data collected by the image. In general there is a trade-off between the amount of compression and the final quality of the image, in order to reduce the space occupied in the storing system by the image without loosing too much information and obtaining images that still look good enough.

Due to lossy coding, JPEG-compressed images typically have a blocky appearance which is often referred to as image compression artifacts. These artifacts (and basically also compression abilities of JPEG) result from casting away neighboring pixels with similar luminance and chrominance components in a manner which prevents recovering their original values. Since JPEG and other lossy compression formats ruin fine details and edges, compression artifacts are considered by many a bigger problem than sensor noise.

Figure 4.5a and 4.5b demonstrate the effect of compression on the image quality. As shown in figure, compressing full-color data using JPEG produces block artifacts and reduces original structural content. It also suppresses potential demosaicing artifacts due to the low-pass nature of lossy compression. Figure 4.5c shows that compression artifacts can be avoided by using lossless coding.

4.2 External image artifacts and deterioration

This second group contains all the artifacts and distortions from the interaction of elements of the scene with the lens system and eventually with the sensor. These type of degradation can be dived in two groups: one containing the artifacts coming from distortions related to the lenses and focus plane, and another one related to elements in the scene that do not directly interact with the camera system.

4.2.1 Lens related artifacts

The very first step of the digital processing pipeline is the collection of sensor data. The only element in between the scene and the camera sensor in a digital SLR camera are the lenses. The lens system has the objective of refract light in order to concentrate light rays on the camera sensor, in order to collect light data for the formation of the image. Light rays, after being reflected by objects in the scene reaches the camera sensor by passing trough the camera lenses which, due to their design, refract light rays in order to *focus* in a specific point, called *focal point*. This process by which the light rays are guided to the sensor is not artifact free, since different distortions can occur due to physical aspects related to refraction and the way light is transmitted. Those artifacts are chromatic aberrations, vignetting and flare effects.

Chromatic aberrations are coloured green, red, purple or blue halos which are highly visible around high-contrast edges in the captured images. This artifacts are caused by the fact that, in a compound lens systems, the individual lens elements have different refractive indices for different wavelengths. This causes the fact that not all wavelengths converge to the same point after travelling through the lens, producing the color artifacts in the images. In Figure 4.6b are shown examples of chromatic aberration artifacts. Two types of chromatic aberrations can be identified: *Longitudinal Chromatic Aberration* and *Transverse Chromatic Aberration*. Figure 4.6a depict, in a single

lens example, the two types of aberrations and the effect of both aberrations combined together.

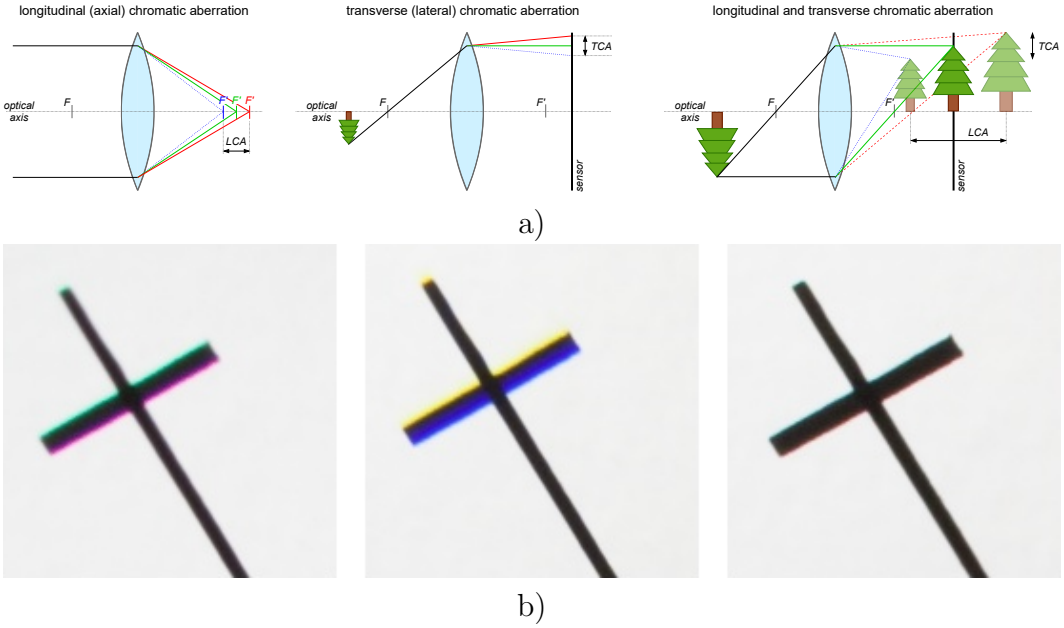


Figure 4.6: Chromatic aberration: in the first row is depicted an exaggerated version of the occourence of the two tipes of abeeration LCA and TCA, in the second row examples of chromatic aberration from a lens test chart. Top images from [171]

Other two kinds of defects coming from the lenses are vignetting and flare. *Flare* is a type of artifact that occurs when the stray radiation from a bright source enters the lens: many internal lens (and even sensor) reflections can occur and then, when this radiation is finally picked up by the sensor, it causes the image to be 'washed out'. When the bright radiation source itself is in the field of view of the camera, bright spots might appear in the image. These spots in general appear in a row, due to the multiple lenses inside the compound lens system. The solution to this problem is simple: to prevent the radiation source from appearing in the camera's field of view by changing the camera position or focal length, or reducing the aperture in order to reduce reflection inside of the lens system.

Vignetting is the phenomena that corresponds to a decrease of brightness towards the edge and corners of the images. Natural vignetting is associated

with the natural illumination falloff when radiation hits the sensor and is unavoidable anytime there is a standard lens in combination with a rectangular sensor. When the radial decrease of image illumination is purely natural or optical, it is easy to correct since it is inherent to lens design. Together with natural vignetting, optical vignetting is an unintentional vignetting that creates a gradually darkened image towards the corners. Luckily, it is easy to correct using image processing algorithms by using camera- and lens-specific profiles that can counteract these types of vignetting. Figure 4.7 shows examples of both Flare and vignetting effect on the final images.

(a) (b)

Figure 4.7: Examples of flare (a) and vignetting (b) effects.

4.2.2 Atmospheric and environmental elements

The last group of problems that can affect an image is the one regarding scene related elements. The implicit assumption that has been done in the previous section is the one that the pictures are always taken in good visibility condition, with a clear vision of the scene content, where the only type of artifacts and problems can come from the processing steps. However, in most of the cases the pictures taken by cameras, from Digital SLR cameras to security camera or cameras mounted on cars etc... , are taken in the most diverse atmospheric conditions, from clear sunny days, to foggy or rainy ones. This aspect introduces a new set of conditions that can affect the overall quality of the images taken. Images of foggy scenes tend to be with low contrast and reduced visibility of objects far from the camera, while in the case of pictures taken in rainy days have a combination of elements which can occlude information and refract light rays before reaching the camera sensor, generating

unpleasing artifacts. These type of elements are not only unpleasing from the aesthetic point of view but can constitute a problem in some specific scenarios like the case of automotive or video security: occluded information or unrecognizable objects can become a serious problem for self-driving cars.

(a) (b) (c)

Figure 4.8: Images taken in adverse atmospheric conditions: the visibility is affected by the presence of external elements such as mist and rain droplets.

These types of artifacts are in general treated after obtaining the sRGB full-color images, in what is called post-processing step. The way the reduction of defects and artifacts or enhancement of visibility is performed depends in general on the final purpose of the images. Programs for image post-processing offers to photographers the possibility to apply algorithms for haze-removal or contrast stretch, but the way those modules operates may not suit more task-related cases like the ones mention before.

4.3 Image quality

To determine how good an image is, in terms of presence of artifacts, overall beauty or usability, it is necessary to have a definition of *image quality*. Defining what image quality is, is an hard task, due to the fact that the meaning of quality differ in relation to the final use of the image. For example, an image that is modified in order to enhance the edges to make them easier to get recognize for an autonomous system, will probably not be considered as a good or beautiful image by a photographer. This specific case is the one in which the definition of quality is related to the concept of *usefulness*. We can mainly identify three definitions of image quality, that can be considered when evaluating an image: *usefulness*, *naturalness* and *fidelity*. As shown in figure 4.9, the definition of quality can be a mixture of these tree main concepts.

In general, the image quality can be defined as the weighted combination of all of the visually significant attributes of an image, such as sharpness,

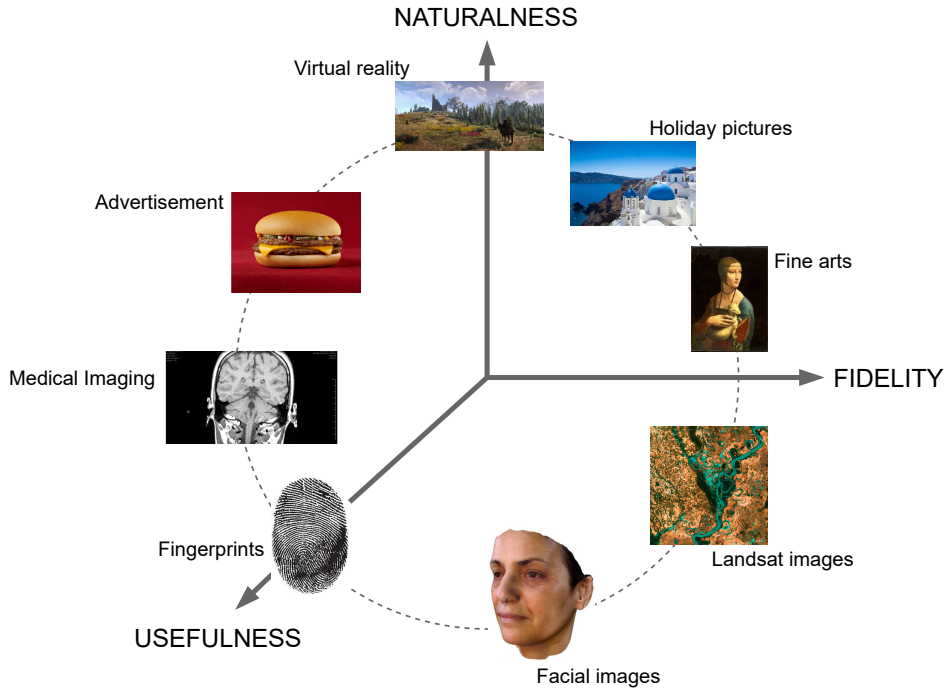


Figure 4.9: Different definitions of image quality. On the three axis are reported the three main definitions of image quality: *usefulness*, *naturalness* and *fidelity*. Even if we give these three definitions, as can be seen, different tasks can consider image quality as a mixture of those concepts.

colorfulness, presence of geometrical distortions, and so on [33]. Technical image quality is often described in terms of a limited set of attributes, each representing different local and global aspects of the imaging experience as well as geometrical attributes like optical distortion and so on. The set of attributes can vary from task to task, and also the relevance of each attribute can change, as in the previous example, where the edge sharpness can be seen in different ways in relation to the automotive point of view or the one of a photographer. In general a perfect image (in relation to the specific task) is the one free from all visible defects.

As seen in the previous sections, image quality of a picture taken with a digital camera is influenced by both camera performances, including shooting conditions and construction flaws or limitations, and scene content. The analysis of the image quality in terms of artifacts or defects is performed us-

ing different metrics, each of which, in general, considers one of the multiple aspects regarding the image quality.

Quality metrics can be divided in two main groups: *full-reference* and *no-reference*. The first group contains all the metrics which uses a reference image to evaluate the considered one. For example metrics like Peak to Signal Noise Ration (PSNR) or Structural Similarity Index Metric (SSIM) [181, 12, 13] are metrics which compare each pixel of an image with a corresponding reference.

Given a reference image f and a test image g, both of size $M \times N$, the PSNR between f and g is defined by:

$$PSNR(f,g) = 10log_{10}\left(\frac{255^2}{MSE(f,g)}\right) \tag{4.4}$$

where

$$MSE(f,g) = \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}(f_{ij} - g_{ij})^2 \tag{4.5}$$

The PSNR approaches infinity as the MSE approaches zero; this show that higher PSNR means higer imag quality.

The metrics in the second group instead make an evaluation based only on the image which we want to analyze. In this group there are also new kinds of metrics based on neural networks, which for each image gives a score of image quality modeled on user perceived quality and other attributes of the images. An example is the Naturalness Image Quality Evaluator (NIQE) [126].

In the next chapter for each approach presented will be introduced the metrics used to evaluate image quality. Since each work is related to a different type of artifact and to a different point of the digital processing pipeline, different metrics will be used, in relation to the specific problem treated.

Part II

Addressing in-camera generated artifacts

As seen in Section 4.1, a certain number of image artifacts come from the application of digital processing steps present in the digital camera processing pipeline. In this chapter, the objective is the correction or replacement of the work done by the processing steps presented in Figure 3.1, by exploiting the power given by machine learning.

In particular in this chapter are treated the problem of the color cast coming from the white balancing step, the image exposure and contrast correction, and finally the compression artifacts and the introduction of noise, which as seen in the previous chapter can come from multiple steps of the digital processing pipeline.

Few aspects play an important role in the design of these processing approaches: the computational complexity and the memory load of the proposed solutions. Since the objective is the one of applying machine learning to the internal processing step of the digital processing pipeline, in the case in which the solution should potentially replace an already existing block, the computational complexity and hardware demands should be taken into consideration. These aspects have been considered in the design process of the solutions proposed in Chapters 5 and 6, leading to lightweight approaches which however exploit the possibilities given by machine learning.

For each of the different processing blocks treated in this chapter an analysis of the specific associated artifacts and existing approaches is presented, alongside an overview of datasets, metrics, and the description of the proposed solutions and related experiments.

Chapter 5

Combination of AWB algorithms for single image and video illuminant estimation

The term chromatic adaptation refers to the human visual system's capability to adjust to widely varying colors of illumination to approximately preserve the appearance of object colors. Digital cameras, or more in general image capturing systems, cannot adjust the relative responsivities of their red, green, and blue imaging layers in the way the human visual system adjusts the responsivities of its color mechanisms. Humans perceive relatively little change in the colors of objects when the illumination is changed from daylight to incandescent. Computational color constancy aims at reducing the chromatic dominant in a digital image, originated from the light source that illuminates the scene. This goal is typically pursued through the development of an algorithm for illuminant estimation. A deeper discussion about the relationship between human chromatic adaptation and camera AWB can be found in Colour appearance models by Mark Fairchild [58]. The research community has been tackling the problem of computational color constancy for several years, designing a disparate set of approaches, which range from handcrafted methods based on low-level image statistics, to data-driven methods based on middle-to-high level analysis. Each individual approach necessarily exploits a specific set of biases and rationales. Due to the ill-posed nature of the problem, in fact, color constancy is not mathematically solvable without relying on additional assumptions on the imaged content. For example, the edge-based color constancy framework by van de Weijer et al. [169] describes

with a unified formulation several low-statistics algorithms that are based on different assumptions: the gray-world hypothesis, which assumes that the average reflectance in a scene is achromatic, the white patch hypothesis, based on the assumption that the reflectance achieved for each of the color channels is equal, or the gray-edge hypothesis, according to which the average of the reflectance differences in a scene is achromatic. Gamut mapping methods are based on the rationale of taking image data captured under an unknown light to a gamut of reference colours taken under a known light. More recent data-driven methods, such as deep learning solutions, implicitly operate higher-level abstractions, by both exploiting statistical biases in the training data, as well as associations with the semantics of the image content. However the most recent and performing methods are increasingly computationally and memory demanding, and require large dataset to be properly trained. The latter point is often taken in low consideration but it is known that most recent methods are sensor-dependent, and therefore need to be retrained for different sensors.

Since different color constancy methods often rely on different assumptions, they can be expected to provide different and uncorrelated outputs, and to consequently perform better on different types of input. The illuminants estimated by these methods, being influenced by the underlying assumptions, can then be considered as image-describing features, and thus properly combined through a fusion strategy for improved color constancy. This approach has been successfully adopted in a wide range of domains, from change detection algorithms for background segmentation [28], to saliency estimation methods [31], to color constancy itself [40, 24, 108]. One of the main drawbacks of the fusion approach is often represented by the inference time, as it requires running multiple independent algorithms on the same input, and subsequently combining the results with a sufficiently-advanced fusion strategy. This aspect becomes particularly problematic in a video-oriented domain, where time is considered critical. In such a scenario, therefore, it is fundamental to select efficient input methods, and to develop an efficient combination strategy. Barron et al. [15] report a threshold of 30 frames per seconds (FPS) to consider an algorithm viable for application in the camera viewfinder stream. The same threshold is also commonly accepted in other fields, such as responsive systems for assisted driving [123]. Even in an off-line color constancy setup, where live-feedback is not required, a fast computation is still critical for the processing of long video sequences.

In the work presented in this section , assuming to have a set of input color constancy methods, not necessarily the most effective ones, the design of

a very efficient late-fusion combination strategy is presented. The presented approach is able to reach an accuracy close to the best algorithms in the state of the art, keeping at the same time the computational burden also suitable for the real time video domain. The proposed single-frame lightweight combination strategy has been applied to a selection of methods based on simple image statistics [169], proving to be effective even when an extremely limited amount of training data is available. The proposed solution outperform other combination strategies on a standard dataset for single-frame color constancy, and reach an illuminant estimation accuracy comparable to more sophisticated solutions.

Morover, an extension of the single-image fusion strategy that exploits a Long Short-Term Memory (LSTM) module to handle varying-length video sequences is presented in this section. Experiments on the recent Burst Color Constancy dataset (BCC) [139] show that: i) exploiting the temporal component after the combination gives better results than exploiting it before the combination; ii) the proposed method outperforms other strategies that can be implemented to exploit the temporal component; iii) the proposed method is able to reach an illuminant estimation accuracy on video sequences comparable to more sophisticated and computationally-demanding solutions specifically designed for video applications.

The proposed solution has also been evaluated in terms of inference time, showing how the combination represents a negligible overhead on the computational time required by the combined algorithms. By optimizing the redundancies of the underlying set of input methods, the model is able to reach real time performance at 31 frames per seconds. Finally, are presented a series of experiments aimed at analyzing the behavior of the proposed combining method, and at assessing the individual contribution of each underlying method towards the final illuminant estimation.

5.1 Related Works

5.1.1 Single-frame combinational illuminant estimation methods

Combinational illuminant estimation methods give an estimate of the scene illuminant by combining the estimates given by a set of input methods. Combinational illuminant estimation methods have been reviewed in [109], where

they have been categorized into two main classes on the basis of the information they use as input: direct combination methods provide their final estimate as a combination of the estimates given by the input methods to be combined; guided combination methods exploit additional information extracted from the input image, in terms of semantic class or features, together with the the estimates given by the input methods to be combined. DC methods have been further grouped into supervised combination (SC) and unsupervised combination (UC) methods: the former ones have a training phase to learn how to combine the estimates given by the input methods, while the latter ones directly combine them without any training phase.

Concerning the direct combination methods (DC), Cardei and Funt proposed two combining methods [40]: Simple Committee, belonging to the UC methods since the combination is performed by simply averaging the estimates of the combined algoriths, and LMS Committee, belonging to the SC methods where the combination weights are learned in a Least Mean Squares optimization.

Bianco et al. [24] proposed a set of different DC-UC methods by exploiting the spatial positions of the estimations to be combined. Considering the estimates as points in the space, Nearest-X averages the estimates of the X algorithms that are closest between each other. The Nearest-X% combination averages all the estimates for which the distance between any pair of them is below $(100 + X)\%$ of that between the two closest ones. The No-N-Max method instead averages the estimates excluding the N estimates having the highest distance from the other estimates. The last method they propose is the Median combinational strategy that selects the estimate having the smallest total distance from all the others.

Li et al. [108] proposed two DC-SC methods: the first uses an Extreme Learning Machine to perform the combination, while the second exploits a Support Vector Regression.

Guided combination (GC) methods exploit additional information extracted from the image to drive the combination: in [23] each image is described by a set of low-level features related to color, texture, and edge distribution and exploits tree-based image classifier trained on indoor, outdoor, close-up classes; [25] uses general-purpose features and problem dependent low-level features without the need of a proxy constituted by semantic classes; a similar approach is used in [73], that exploits texture and contrast summarized in terms of the Weibull parameterization; [170] uses high-level visual information to im-

prove illuminant estimation by modelling the image as a mixture of semantic classes, such as sky, grass, road, and building; [119] use rough 3D scene geometry to model an image in terms of different geometrical regions and depth layers.

Given the success of the above combining methods Li et al. [110] proposed a multi-cue method that combines the information provided by different cues, e.g. properties of the low-level RGB color distribution, mid-level initial illuminant estimates provided by subordinate method, and high-level knowledge of scene content, within the framework of a tree-structured group joint sparse representation.

Subhashdas et al. [158, 159] propose a hybrid multi-class dynamic weight model with an ensemble of classifiers: their method classifies images into several groups and uses a distinct dynamic weight generation model (DWM) for each group. The DWM generates dynamic weight using an image feature that has a correlation with the capability of the input algorithms used for combination.

5.1.2 Video illuminant estimation methods

Although frame-based illuminant estimation methods can be applied also to videos and/or image sequences on a per-frame basis, there are only a few methods actually able to exploit the temporal component to produce a more robust illuminant estimate.

Yang et al. [187] extract illuminant color from two distinct frames of the same scene exploiting highlights on specular surfaces. Prinet et al. [135] propose a probabilistic and more robust version of [187].

Wang et al. [177] propose a multi-frame illuminant estimation method by clustering illuminant estimate coming from a standard method on each frame into a number of video shots and then exploit a summary statistics to provide a global estimate for the whole shot.

More recently, Barron et al. [15] extended their single frame method to work on image sequences by building a smoothing model inspired by Kalman filter in order to smooth wrong predictions that may happen on individual frames.

The work of Quian et al. [137] is the first to actually exploit the information available in the input sequence. They propose an end-to-end trainable recurrent color constancy network that exploits AlexNet features and a Long

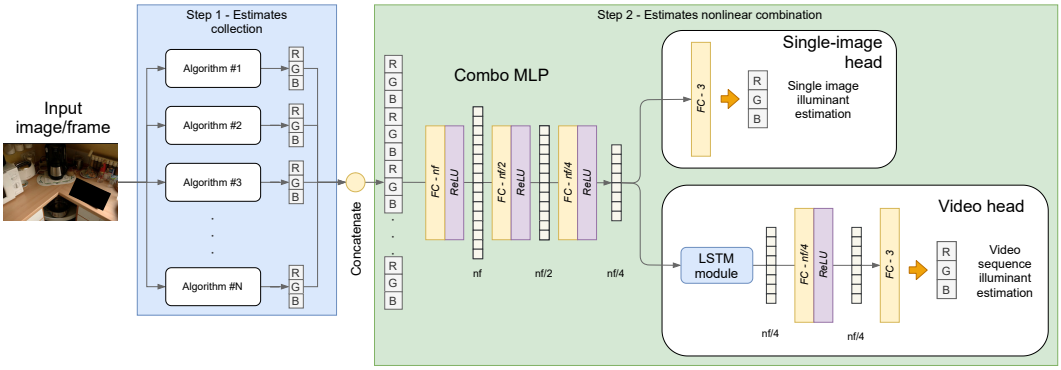


Figure 5.1: Combination framework for illuminant estimation combination. The framework is composed of two different steps: the first one corresponds to the collection of the statistics-based approaches estimations, the second one corresponds to the actual combination of the estimations previously collected. As can be seen, the Single-Image model and the Video model shares the same architecture for the first combination part: the two models differs for the different heads. In the Single-Image case, the head is made of only one linear layer, used to map the $nf/4$ features to the output dimensionality. For the Video model, the $nf/4$ features are further processed by a LSTM to exploit the temporal nature of the video sequence. The details of the video sequence processing are shown in figure 5.2.

Short Term Memory (LSTM) recurrent neural network to process sequential input frames. Their method has been then improved [139] by exploiting a more powerful backbone network for the semantic feature extraction, and using a 2D LSTM that provides more effective spatial recurrent information.

5.2 Proposed Method

The proposed solution consists of a framework for the non-linear combination of illuminant estimations, using a small neural network composed of few hidden layers in a multilayer perceptron (MLP) architecture. The general idea is to exploit the different assumptions related to different illuminant estimation algorithms. Two variants of this model have been designed: a single-image illuminant estimation version, and a video estimation version operating on multiple input frames. In this section the framework both configurations are presented, with the respective architectures and the objective function adopted

for training.

5.2.1 Single-Image model

The proposed framework for illuminant estimations combination is illustrated in Figure 5.1. The procedure is divided into two steps: the first step consists in performing the initial illuminant estimation using a given set of algorithms, in order to collect the different estimations to be combined. The second step corresponds to using our multilayer perceptron, called COCOA, to obtain the corresponding non-linear combination of the input estimations.

The COCOA network is a multilayer perceptron model made of four linear layers which uses Rectifying Linear Unit (ReLU) activation functions. The structure of COCOA is represented in Figure 5.1. As can be seen from Figure 5.1 the number of perceptrons per layer is defined as a function of the number of perceptrons in the first layer. In the proposed configuration the nf has been setted at 256, obtaining a four-layer model with respectively 256, 128, 64, and 3 perceptrons.

Given a set of algorithms for combination, the COCOA model is trained by giving as input the concatenation of the estimations, in normalized RGB space, and compare the output combination with the ground truth. The number of algorithms used to obtain the starting estimations determine the dimensionality of the first layer of COCOA.

5.2.2 Video model

Here is presented a variant of the proposed model, specifically designed for the processing of video sequences.

In this scenario, for each frame in a given sequence, our model takes as input a set of estimations, performed with a set of input illuminant estimation methods, and extracts a vector of nf/4 features. This part of the model corresponds to the first three layers of the single-image illuminant estimation model. The resulting features are are then processed by a Long Short-Term Memory module (LSTM).

For each frame, the LSTM module takes in input the representation given by the MLP and generates a new set of features, representing the frame sequence until the last processed frame. For each frame of the sequence, the MLP feature extraction step with the LSTM module temporal processing is repeated, using as input the estimations of the input methods corresponding

to the new frame, and the hidden state coming from the previous step of the LSTM module (with exception for the first frame). The final results coming from the processing of each frame is eventually passed to a final group of two fully connected layers, which outputs the estimation for the entire sequence. The video estimations combination process is depicted in Figure 5.2. As can be seen from Figure 5.1, the model for the video estimation combination is an extension of the original single-image model presented in Section 5.2.1. Instead of having a final layer which maps the nf/4 representation to the output dimensionality 3, there are new components which handle the multi-frame nature of the video sequence.

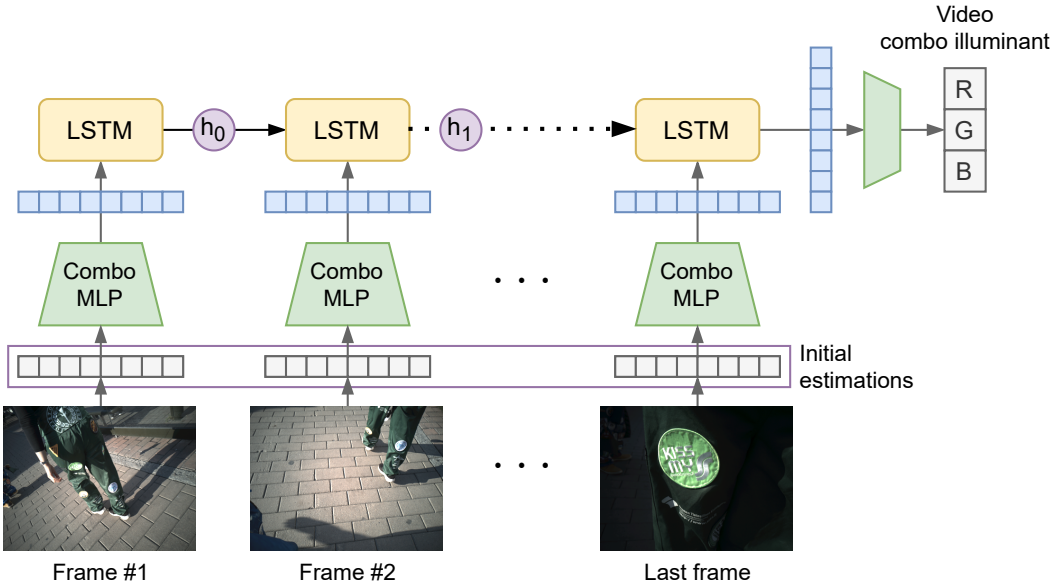


Figure 5.2: Combination of the illuminant estimations between frames of a video sequence. For each frame the 6 estimations are first processed by the Combo MLP component, then are given in input to the LSTM module. Finally the processed features are sent to the last two layers, giving in output the final estimation.

The LSTM is initialized with starting hidden state and starting cell state at zero values, with hidden state dimension equal to nf/4. For the last two layers, as can be seen in Figure 5.1, the number of output features for the two fully connect layers is respectively nf/4 (64 in our configuration using nf = 256) and 3.

5.2.3 Loss function

The two models are trained by minimizing the recovery angular error (expressed in degrees) between the output of COCOA and the ground truth illuminant associated to the image or to the video sequence. In the case of the single-image model, there is a ground truth illuminant for each image in the dataset, while for the video case scenario, for each video sequence in the dataset there is a single ground truth illuminant triplet for the entire video sequence. This is determined by the chosen datasets for experimentation, as illustrated in Section 5.3.2.

The recovery angular error, which quantifies the illuminant estimation error, is represented by the angle between the vector given by the target illuminant triplet $\rho^{gt} = (R^{gt}, G^{gt}, B^{gt})$ and the one corresponding to the result of the combination $\rho^{E} = (R^{E}, G^{E}, B^{E})$. A generalization of the recovery angular error corresponds to what is called SAM in the context of satellite image and spectral comparison metrics [6]. Given two illuminants ρ^{E} and ρ^{gt}, the recovery angular error can be calculated as:

$$\theta = \arccos\left(\frac{(\rho^{E} \cdot \rho^{gt})}{||\rho^{E}|| \cdot ||\rho^{gt}||}\right) \tag{5.1}$$

5.3 Experimental Setup

5.3.1 Training setup

The COCOA architecture is written in Pytorch 1.7.0 and trained on an NVIDIA Titan V with 12 GB of memory. Training was performed using the Adam [104] optimizer; for the single image scenario starting learning rate is set at 0.003 and weight decay of 1e-5, while for the multi-frame training the starting learning rate used is 1e-4 and weight decay of 1e-5. Both models were trained for a total amount of 3000 epochs. These hyper parameters have been empirically determined among different training runs.

5.3.2 Datasets

To train and evaluate the performance of the COCOA model, different setups and datasets for the image and video tasks are used. For single-image illuminant estimation the 569 images of the Shi-Gehler reprocessed dataset [72, 153]

have been adopted, while for the video illuminant estimation task the 600 sequences of the Burst Color Constancy dataset (BCC) from Qian et al. [139] have been used.

For the evaluation of the single-image illuminant estimation on the Shi-Gehler dataset has been adopted the original three-fold cross validation division, as done previously by [86, 15]; the validation has been performed by randomly selecting the 20% of the training images for each fold. For the video dataset has been used the original dataset division provided by the authors, for training and test. To validate the model, 20% of the video sequences from the training set have been randomly selected and used.

5.3.3 Combined input methods

For each image or frame, have been collected six different illuminant estimations from different statistics-based algorithms:

- Shades of Gray (SoG)
- General Gray World (gGW)
- Gray Edge 1st order (GE1)
- Gray Edge 2nd order (GE2)
- Gray World (GW)
- White point (WP)

This particular selection aims at creating an overall illuminant estimation pipeline that is also practical, i.e. by relying on simple input methods, its computational complexity remains low and suitable for a real-time application, as shown in Section 5.4.4. The illuminant estimation using those models have been performed using the framework from van de Weijer et al. [169], which offers a single equation to perform the illuminant estimation corresponding to different assumptions over the images. The general hypothesis is described as:

$$\left(\int \left| \frac{\partial^n f^\sigma(x)}{\partial x^n} \right|^p dx \right)^{1/p} = k e^{n,p,\sigma} \tag{5.2}$$

where n identifies the derivative order, σ is the standard deviation for a Gaussian filter, and p is the order of the Minkowski norm. To specialize the behaviour of the six algorithms listed above, the parameters of Equation 5.2

have been selected as shown in first columns of Table 5.1 (COCOA) , following [26]. An alternative set of parameters is also investigated, by fixing all free parameters to 1 (columns COCOA-fast). This experiment is driven by several motivations: 1) to avoid relying on arbitrary parameters that were potentially optimized to a specific dataset, 2) to speed up the computation by only exploiting small convolutional kernels, and 3) to further speed up the computation by sharing common processing steps among multiple methods. As it can be observed from the table, this second set of parameters has the Shades of Gray algorithm collapse into a Gray World, thus reducing the effective total number of input methods from six to five.

Table 5.1: Parameters for each illuminant estimation algorithm. The free parameters that can be changed without switching to a different method are highlighted in boldface.

	COCOA			COCOA-fast		
	n	p	σ	n	p	σ
Shades Of Gray (SoG)	0	**4**	0	0	**1**	0
Gray World (GW)	0	1	0	0	1	0
Gray Edge 1st order (GE1)	1	**1**	**6**	1	**1**	**1**
Gray Edge 2nd order (GE2)	2	**1**	**1**	2	**1**	**1**
general Gray World (gGW)	0	**9**	**9**	0	**1**	**1**
White Patch (WP)	0	∞	0	0	∞	0

The camera black level is subtracted from all images, and these are subsequently rescaled to have their maximum side be 256 pixels long. After this pre-processing, each image has been eventually fed into each one of the algorithms, obtaining a total amount of six estimations per image. These six estimations are first normalized and then concatenated and used as input for the COCOA model. A series of preliminary experiments have been conducted to define the most appropriate normalization strategy, including L2 normalization, green channel normalization, and conversion to various chromaticity representations. The final configuration, adopted throughout all our experiments, relies on green-channel normalization. The final output of the network consists of an RGB triplet corresponding to the non-linear combination of the input estimates.

5.4 Experimental Results

5.4.1 Combinational single-image illuminant estimation

In this section is for first presented the improvement induced by the proposed COCOA-IH with respect to the input methods described in Section 5.3.3, and then is shown the comparison of the results with the application of other combinational methods in the state of the art. The combinational methods belong to the three categories identified in Section 5.1: direct combination using unsupervised combination (DC-UC), direct combination using supervised combination (DC-SC), and guided combination (GC). In order to perform a fair comparison, all the compared methods consider the same set of input methods (and parameters) as our COCOA-IH solution. The only exception is the Multi-Cue (MC) method by Li et al. [110], whose code is not available for reproduction, and whose reported results are based on the same methods although with slight variation in the choice of parameters.

The results in terms of average, median and maximum angular error statistics on the Shi-Gehler dataset are reported in Table 5.2. Our COCOA-IH model is able to reduce by 32% the mean angular error with respect to the best input method (GE1) and by 58% with respect to the worst one (WP), thus suggesting a good ability at feature selection and combination. An in-depth analysis of the impact of each underlying input method is provided in Section 5.4.5. From the reported results it is possible to see that COCOA-IH is also able to outperform by a large margin the other compared combinational methods belonging to all analyzed groups. The version of our model with fast parameters, COCOA-IH-fast, produces generally equivalent results with respect to COCOA-IH in this setup.

In Figure 5.5 are shown the three images of the Shi-Gehler dataset on which COCOA-IH obtains the worst results, while the three images on which it obtains the best results are reported in Figure 5.6. It is possible to notice how the worst results correspond to images with colored background/objects and to a scene with multiple illuminants. The best results instead correspond to images in which the underlying assumptions of the individual methods used by COCOA-IH are more likely to be satisfied.

To further analyze the performances of the proposed combination framework COCOA-IH have been trained with reduced versions of the training set of the Shi-Gehler dataset. The sizes considered are determined by successively halving its original size from 1, corresponding to the original size, to 1/32. For

Table 5.2: Results of combinational single-image illuminant estimation algorithms, in terms of angular error on the Shi-Gehler dataset, and comparison with the combinational algorithms in the state of the art. Algorithms are divided into direct combination with unsupervised combination (DC-UC), direct combination with supervised combination (DC-SC), and guided combination (GC).

	Method	Mean	Med.	Max
Input	Shades of Gray (SoG) (0,4,0)	4.58	2.58	22.79
	Gray World (GW) (0,1,0)	4.78	3.65	24.91
	Gray Edge 1st order (GE1) (1,1,6)	3.94	2.85	23.37
	Gray Edge 1st order (GE1-fast) (1,1,1)	4.09	3.15	18.91
	Gray Edge 2nd order (GE2) (2,1,1)	4.12	3.31	17.77
	general Gray World (gGW) (0,9,9)	4.40	2.89	22.40
	general Gray World (gGW-fast) (0,1,1)	4.79	3.67	25.03
	White Patch (WP) (0,∞,0)	6.36	3.93	45.78
DC-UC	Simple Committee [40]	4.18	3.00	20.55
	Nearest-2 (global) (N2) [24]	3.93	2.88	19.99
	Nearest-2 (per image) (N2) [24]	4.04	2.54	22.07
	Nearest-10% (global) (N-10%) [24]	3.98	2.63	20.80
	Nearest-10% (per image) (N-10%) [24]	4.01	2.55	21.97
	Nearest-30% (global) (N-30%) [24]	3.98	2.68	21.49
	Nearest-30% (per image) (N-30%) [24]	4.02	2.65	22.65
	No-1-max (global) (N1M) [24]	4.03	2.84	20.57
	No-1-max (per image) (N1M) [24]	3.96	2.71	21.32
	No-2-max (global) (N2M) [24]	3.98	2.63	20.80
	No-2-max (per image) (N2M) [24]	3.90	2.49	20.83
	Median (global) (MD) [24]	3.94	2.85	23.37
	Median (per image) (MD) [24]	3.89	2.66	20.83
DC-SC	LMS Committee [40]	4.27	2.62	68.72
	Extreme Learning Machine (ELM) [108]	4.40	3.25	21.15
	Support Vector Regr. (lin) (SVRL) [108]	3.51	2.87	16.51
	Support Vector Regr. (rbf) (SVRR) [108]	3.26	2.45	18.16
	COCOA-IH (this work)	2.66	1.78	21.45
	COCOA-IH-fast (this work)	2.64	1.86	16.23
GC	Natural Image Statistics comb. (NIS) [73]	4.07	2.98	20.37
	Bianco et al. 2010 [25]	4.09	2.93	20.44
	Bianco et al. 2008 [23]	3.89	2.63	20.68
	Multi-Cue (MC) [110]	3.25	2.20	

each training set size, five different random selections (runs) are performed: in Figure 5.3 is reported a plot with the average angular error and its standard deviation for each trained model, averaged over the different runs performed. In the same plot are reported the performances of the best input algorithm combined by COCOA-IH (i.e. Gray Edge 1st order) as a dashed line. As can be seen in the plot the best performance are obtained when all the data available for training in the original splits of the Shi-Gehler dataset are used (i.e. about 378 images, averaged over the three cross validation folds). As expected as the training set size is reduced the average angular error increases. Nevertheless, even reducing the training set to 1/8 of its original size, which corresponds to a total of about 48 images (to be further split into the actual training set and validation set according to a 80%-20% ratio), COCOA-IH still performs better than the best input method combined. For smaller training sets the average angular error rapidly degrades, showing no advantage of using COCOA-IH over the best input method combined for a training set size equal to 1/16 of its original size, corresponding to a total of about 24 images to be further divided into train and validation. The performed experiment shows how the proposed method COCOA-IH can improve over the best input method, even when the number of images available for training is scarce.

5.4.2 State-of-the-art single-image illuminant estimation

With this experiment are compared the performance of the proposed COCOA-IH with respect to individual state of the art algorithms for single-image illuminant estimation on the Shi-Gehler dataset. The 21 compared methods belong to three different groups on the basis of the type and level of training they need. The first group encompasses the parametric methods: Bright Pixels (BP) [96], Cheng et al. [45], and Grey Pixel (edge) [186]. In the second group there are learning-based methods that require no supervision in terms of illuminant ground truth: Buzzelli et al. (global normalization and channel normalization) [35], and Quasi-Unsupervised [22]. The third group comprises the fully-supervised methods, that need a complete training on illuminant data to properly operate: Bayesian [72], Spatio-Spectral (ML and GP) [43], Natural Image Statistics [73], Exemplar-based [95], Chakrabarti (Empirical and End-to-end) [42], Cheng et al. [46], Bianco et al. [29], FFCC [15], Oh and Kim [130], CCC (dist+ext) [14], FC4 (AlexNet) [86], DS-Net (HypNet+SelNet) [154], and Quasi-Unsupervised with Fine Tuning [22].

The results in terms of average, median and maximum angular error statis-

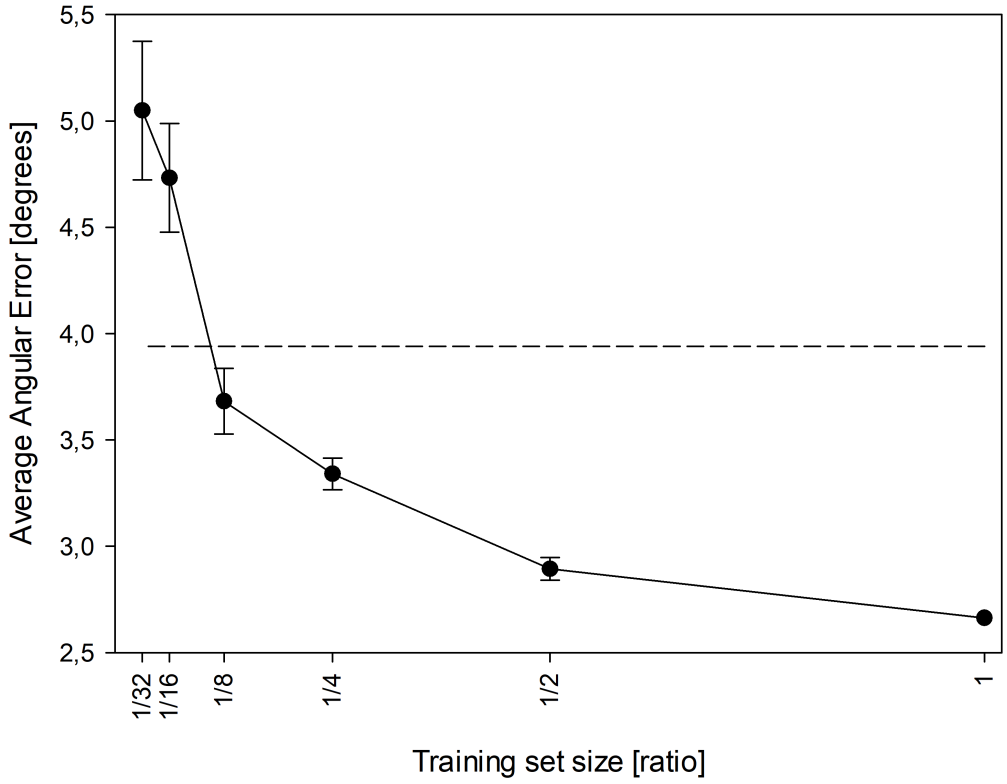


Figure 5.3: Performance of COCOA-IH in terms of average angular error reducing the training set size as a ratio of the classical data partition of the Shi-Gehler dataset. The dashed line represents the performance of the best input algorithm used by COCOA-IH, i.e. Gray Edge 1st order (GE1).

tics are reported in Table 5.3. It is possible to notice how the best results are obtained within the group of supervised algorithms. The proposed method with Image Head, i.e. COCOA-IH, compares favorably with the state of the art, placing itself in the upper part of an hypothetical ranking, close to some early CNN-based methods, despite it only combines unsupervised and parametric methods.

In addition to comparing methods across multiple statistics (mean, median, maximum errors), an ideal assessment would involve the Wilcoxon signed-rank test [183] to compare the entire error distributions, and thus provide a level of statistical significance. This was, however, not possible due to the unavailability of illuminant estimations for the compared methods (aggregate statistics are reported from the corresponding publications). On the other hand, it is possible to observe that, according to a literature survey by Gijsenij et al. [74], a deviation of 1° in angular error with the ground truth is considered below the level of what can be perceived by a human being [67], while the range between 2° and 3° is considered detectable but still acceptable [60, 63].

5.4.3 Exploiting the temporal component

several solutions to exploit the temporal component have been investigated, in order to process video sequences. In general, they can be classified as embedding the temporal component before or after the combination of input methods. Combining before (B) allows exploiting the temporal component in each single input method, while combining after (A) means exploiting the temporal information only once, at the combination level.

The investigated solutions are the following.

- Frame average: it is the simplest approach where the output illuminant for a video corresponds to the average of the estimates on each frame. If it is applied before (B) the combination, the estimates of each single input algorithm are individually averaged to give the corresponding video illuminant estimate; these estimates are then combined by COCOA-IH to give the final estimate. If it is applied after (A) the combination, COCOA-IH is applied to the estimates given by the individual methods to each frame, and the estimates by COCOA-IH for each frame are then averaged to give the final estimate.

- Frame median: it is the same approach as the previous one but considering the median instead of the average operations to combine the

Table 5.3: Comparison in terms of angular error with the individual, single-image illuminant estimation algorithms in the state of the art on the Shi-Gehler dataset. As a pedex to all the Mean and Median angular values, it is reported its position in a hypothetical ranking.

	Method	Mean	Med.	Max
Param.	Bright Pixels (BP) [96]	$3.98_{(18)}$	$2.61_{(16)}$	
	Cheng et al. [45]	$3.52_{(16)}$	$2.14_{(13)}$	28.35
	Grey Pixel (edge) [186]	$4.60_{(20)}$	$3.10_{(19)}$	
Unsup.	Buzzelli et al. (gl. norm) [35]	$4.84_{(22)}$	$4.12_{(22)}$	20.80
	Buzzelli et al. (ch. norm) [35]	$5.48_{(23)}$	$4.81_{(23)}$	19.88
	Quasi-Unsupervised [22]	$3.46_{(14)}$	$2.23_{(14)}$	21.17
Supervised	Bayesian [72]	$4.70_{(21)}$	$3.44_{(21)}$	
	Spatio-Spectral (ML) [43]	$3.55_{(17)}$	$2.93_{(18)}$	
	Spatio-Spectral (GP) [43]	$3.47_{(15)}$	$2.90_{(17)}$	
	Natural Image Statistics [73]	$4.09_{(19)}$	$3.13_{(20)}$	
	Exemplar-based [95]	$2.89_{(11)}$	$2.27_{(15)}$	
	Chakrabarti (Empirical) [42]	$2.89_{(11)}$	$1.89_{(11)}$	
	Chakrabarti (End-to-end) [42]	$2.56_{(8)}$	$1.67_{(8)}$	
	Cheng et al. [46]	$2.42_{(7)}$	$1.65_{(7)}$	
	Bianco et al. [29]	$2.36_{(6)}$	$1.44_{(5)}$	16.98
	FFCC [15]	$1.78_{(2)}$	$0.96_{(1)}$	16.25
	Oh and Kim [130]	$2.16_{(5)}$	$1.47_{(6)}$	
	CCC (dist+ext) [14]	$1.95_{(4)}$	$1.22_{(4)}$	
	FC4 (AlexNet) [86]	$1.77_{(1)}$	$1.11_{(2)}$	
	DS-Net (HypNet+SelNet) [154]	$1.90_{(3)}$	$1.12_{(3)}$	
	Quasi-Unsupervised + Fine Tune [22]	$2.91_{(13)}$	$1.98_{(12)}$	19.90
	COCOA-IH (this work)	$2.66_{(10)}$	$1.78_{(9)}$	21.45
	COCOA-IH-fast (this work)	$2.64_{(9)}$	$1.86_{(10)}$	16.23

per-frame estimates.

- Gaussian weights with free standard deviation: it is an extension of the first approach, where the combination weights are not uniform anymore but are taken from a Gaussian distribution with a free standard deviation σ. The Gaussian distribution is centered on the last frame and therefore decreasing weights are given to the older frames. For simplicity, all the sequences are extended to a common length by adding the necessary number of dummy illuminant estimates at the beginning of each sequence.

- Gaussian weights with free standard deviation and center: it is an extension of the previous approach, in which also the center x_0 of the Gaussian is a free parameter. Similarly to the previous approach, all the sequences are extended to a common length.

- LSTM - Long Short-Term Memory: in this approach the temporal component is exploited using LSTMs. When LSTMs are used before (B) the combination, one LSTM is applied to each of the inputs of COCOA-IH and the resulting model is trained end to end. When the temporal component is exploited after (A) the combination, a single LSTM is used and the model corresponds to the COCOA-VH described in Section 5.2. LSTM (B) has been initialized with the same configuration as LSTM (A). The main architectural difference is in the input and output feature size that in LSTM (B) have been set to 3, corresponding to the dimensionality of the input estimations to be time processed.

The different approaches considered to exploit the temporal component are tested on the BCC dataset [139]. The numerical results of this comparison are reported in Table 5.4: it is possible to see that the best performance in terms of both average and median angular errors are obtained by the COCOA-VH which uses an LSTM to exploit the temporal component. More in general, it is possible to see how the approaches that exploit the temporal component after the combination (A) obtain better results than the corresponding versions that exploit it before (B) the combination: on average this improvement is 1.1 degrees on the mean angular error, 0.8 degrees on the median angular error and 3.2 on the maximum angular error, respectively corresponding to a 26.6%, 27.7% and 25.9% improvement.

Table 5.4: Comparison of different solutions to exploit the temporal component, tested on the BCC dataset. The "Time" column refers to before (B) or after (A) the combination of input methods.

Method	Time	Mean	Med.	95%-Quant
Frame average	B	4.20	3.15	12.32
Frame median	B	4.10	2.68	13.12
Gauss. weights (σ)	B	4.23	3.03	12.37
Gauss. weights (σ, x_0)	B	3.98	2.90	11.51
LSTM	B	2.77	2.06	8.46
Frame average	A	2.83	2.11	7.79
Frame median	A	2.88	2.05	9.12
Gauss. weights (σ)	A	2.88	2.17	7.82
Gauss. weights (σ, x_0)	A	2.67	1.91	8.08
LSTM (**COCOA-VH**)	A	2.61	1.66	8.81

5.4.4 State-of-the-art video illuminant estimation

In this experiment are compared the proposed COCOA-VH against state-of-the art video illuminant estimation methods on the BCC dataset. The focus here is on methods specifically designed for videos/image sequences (i.e. Prinet et al. [135], RCC-Net [137] and BCC-Net [139]), as well as two existing temporal extensions of supervised single-frame algorithms (i.e. T.GI for Grayness Index [138] and T.FFCC for Fast Fourier Color Constancy [15]).

The numerical results in terms of average, median and 95%-quantile angular error statistics are reported in Table 5.5. The results show how the proposed COCOA-VH ranks second in terms of both the average and the median error, surpassing more complex methods. In Figure 5.7 are reported the three sequences of the BCC dataset on which COCOA-VH obtains the three worst results. For each sequence have been drawn the plot of the illuminant estimated by each of the six combined algorithms on each frame of the sequence plotted as chromaticities in the ARC space [36] together with the final estimate by COCOA-VH and the ground truth. The plots show how in the initial frames the six estimates are closer to the ground truth and then start to diverge from it, thus causing the drift of the final COCOA-VH estimate. In Figure 5.8 are reported the three sequences of the BCC dataset on which COCOA-VH obtains the three best results. From the plots it is possible to

Table 5.5: Comparison in terms of angular error with the video illuminant estimation algorithms in the state of the art on the BCC dataset.

Method	Mean	Med.	95%-Quant
Prinet et al. [135]	7.51	6.94	20.70
Temporal extended GI (T.GI from [139])	4.73	2.96	17.42
Temporal extended FFCC [15]	3.35	1.70	17.41
RCC-Net [137]	2.74	2.23	8.21
BCC-Net [139]	1.99	1.21	6.34
COCOA-VH (this work)	2.61	1.66	8.81
COCOA-VH-fast (this work)	2.66	1.88	8.44

notice how these cases correspond to sequences on which the combined algorithms already provide a good illuminant estimate. Concerning the content of the sequences obtaining the worst and best results is possible to observe a strong similarity with those reported in Figures 5.5 and 5.6. This is not surprising since both COCOA-IH and COCOA-VH exploit the same set of input illuminant estimation methods and they have the same backbone architecture, just differing in the regression head.

As a further analysis the computational complexity of the compared methods have been measured, focusing on video illuminant estimation due to the critical role that efficiency assumes in this domain: fast online processing allows a direct feedback in the camera viewfinder, and fast offline processing enables handling large amounts of video data. Given the heterogeneous nature of the code available for the different methods, and the different hardware on which they run (i.e. CPU vs GPU), in order to perform a fair comparison, the number of floating point operations for each compared method have been calculated. In Figure 5.4 are plotted the average angular error reached by each method reported in Table 5.5 with respect to the number of operations. From the plot is possible to observe how the proposed methods are in the bottom left corner of the plot, providing the best trade-off between illuminant estimation accuracy and computational complexity, with COCOA-VH-fast being the one requiring the lowest number of operations, i.e. 16.6 millions of operations (M-Ops) of which just 0.56% are due to the actual non-linear combination. In practice, COCOA-VH and COCOA-VH-fast work at 21.97 FPS and 31.48 FPS respectively, the latter fully reaching the real-time threshold of 30 FPS [15, 123], with the bottleneck being the CPU-based im-

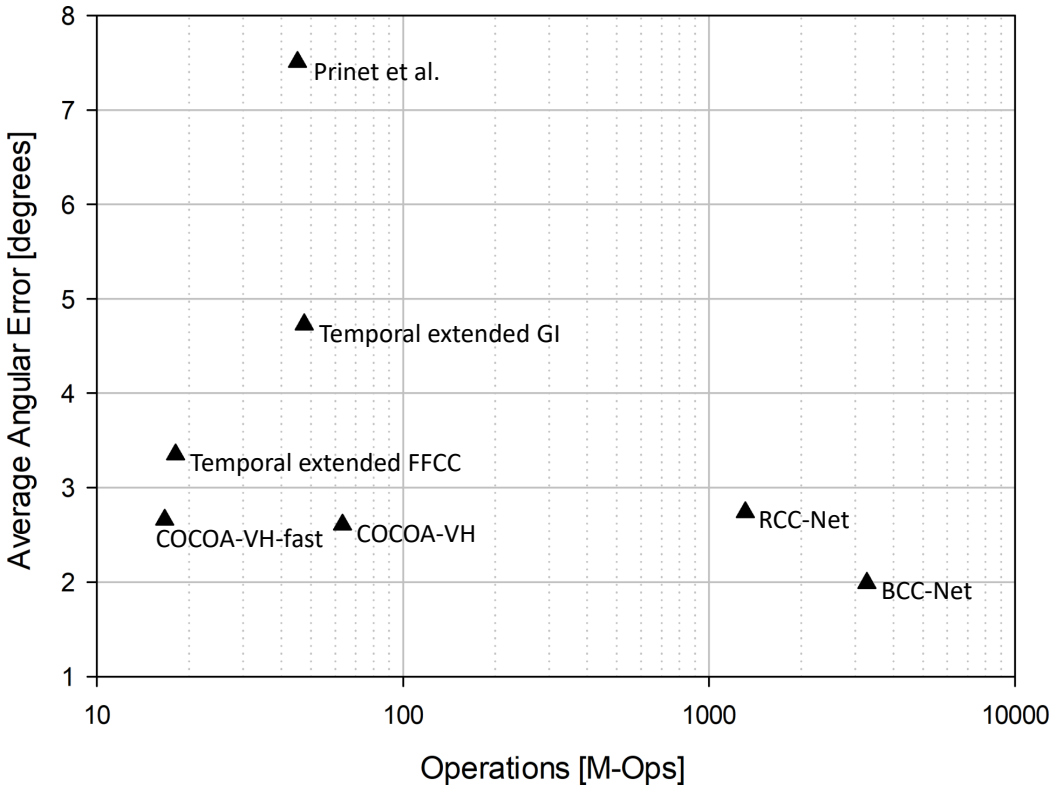


Figure 5.4: Plot representing the average angular error (in degrees) with respect to the computational complexity (in terms of millions of operations) of the methods reported in Table 5.5. The ideal point is in the bottom-left corner.

plementation of the input methods. A lower illuminant estimation error is obtained by BCC-Net [139], that requires a number of operations that is two orders of magnitude higher, i.e. 3277.1 M-Ops.

5.4.5 Sensitivity analysis

In this experiment a sensitivity analysis of COCOA-IH is performed, in order to understand how a change in one of the six inputs affects the final output.

The sensitivity analysis is performed exploiting the ARC color space [36], that has the property that euclidean distances correspond to angular errors. Each of the six input have been individually changed, by modifying the esti-

mate given by the corresponding algorithm from 0 to 0.1 radians (i.e. approximately 5.7°) in steps of 0.01 radians. The modification is performed along 36 directions, in order to cover the possible hues in 10°steps. The dataset considered is the Shi-Gehler reprocessed dataset [72], and for each possible input modification the average angular error is computed. The six surfaces obtained by considering all the possible input modifications of each input individually are reported in the top row of Figure 5.9. The bottom row of the same figure reports the level curves of these surfaces. In all the plots the corresponding crop of the ARC space is reported as a reference in order to understand the sensitivity with respect to different hues. The center point of all the six plots correspond to the case where no input is modified and thus the result correspond to the average angular error reported in Tables 5.3 and 5.2 for COCOA-IH (i.e. 2.66°).

From the plots reported it is possible to notice how in general there are inputs with respect to which COCOA-IH is more sensitive, i.e. the third and the fifth inputs respectively corresponding to GE2 and GW. This is also numerically confirmed in Table 5.6 where the average slope for each surface is computed. Furthermore is possible to observe how the sensitivity is not isotropic for any of the inputs, but the surfaces are approximately symmetric with an axis of symmetry passing close to the center of the plot and with a different direction for each of the inputs. The approximate direction of the axis of symmetry is reported for each surface in Table 5.6. In particular COCOA-IH is very sensitive to changes in the red-cyan direction for what concerns GE2 with an axis of symmetry approximately orientated at 80°, while the most sensitive direction with respect to GW is the green-purple direction with an axis of symmetry approximately orientated at 30°. It is also possible to observe how COCOA-IH has a very low sensitivity with respect to the first and the sixth inputs, i.e. SoG and WP. It is also possible to notice how there is a region for each input able to obtain a lower average angular error with respect to the one obtained when no change is applied to the inputs. This is due to the fact that in the classical three-fold subdivision of the Shi-Gehler dataset, the training and testing illuminants have a different distribution.

5.5 Summary

Computational color constancy has been addressed through the years with a wide variety of approaches, often relying on different assumptions over the

Table 5.6: Statistics of the sensitivity analysis of the COCOA-IH model with respect to the individual inputs: average slope, the higher the more sensitive is the model with respect to the corresponding input. Direction of axis of symmetry, that approximately corresponds to the direction of lowest sensitivity.

Input	Average slope	Axis of symmetry
SoG	1.4410	30°
GE1	2.9751	10°
GE2	6.1226	80°
gGW	2.3920	30°
GW	4.3267	30°
WP	1.4736	20°

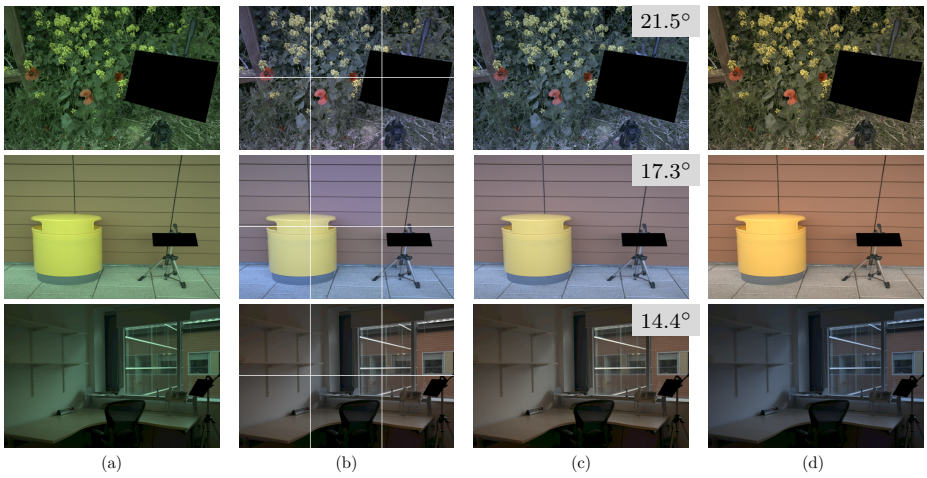


Figure 5.5: Visualization of the three images of the Shi-Gehler dataset on which COCOA-IH obtains the three worst results. Input image (a); collage image obtained from the six images respectively collected the illuminant estimated by each of the six individual algorithms (b); image corrected with the illuminant estimated by COCOA-IH, with the angular error overlaid in the top right corner (c); ground truth, i.e. image corrected with the ground truth illuminant (d).

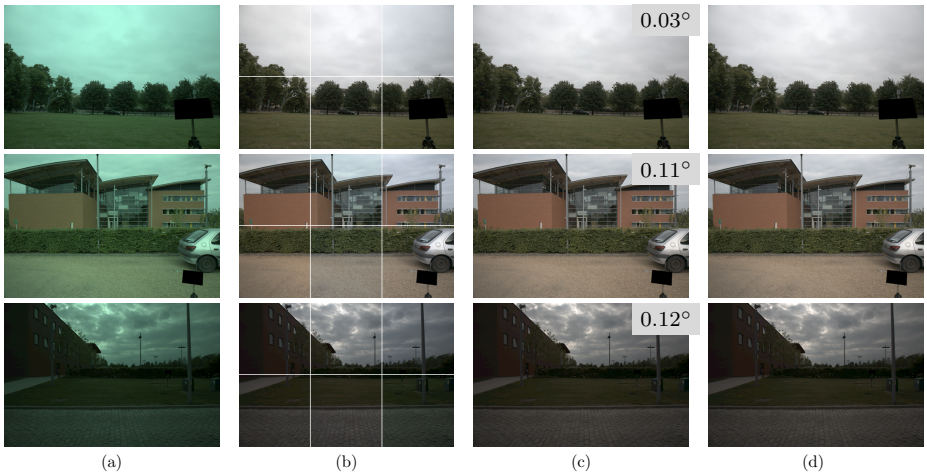


Figure 5.6: Visualization of the three images of the Shi-Gehler dataset on which COCOA-IH obtains the three best results. Input image (a); collage image obtained from the six images respectively collected the illuminant estimated by each of the six individual algorithms (b); image corrected with the illuminant estimated by COCOA-IH, with the angular error overlaid in the top right corner (c); ground truth, i.e. image corrected with the ground truth illuminant (d).

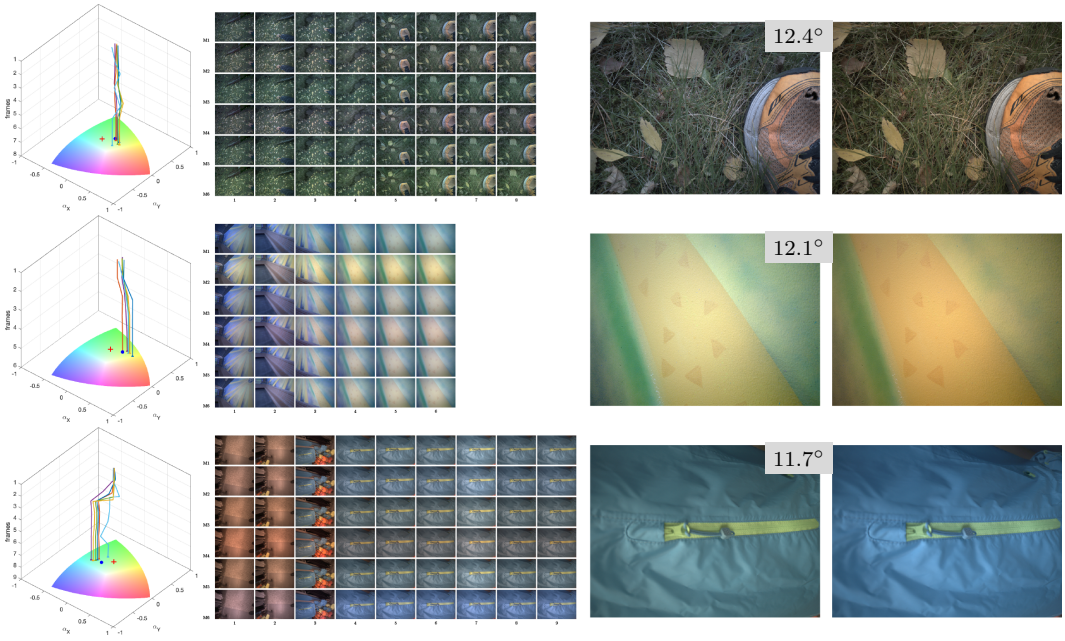


Figure 5.7: Worst 3 results of COCOA-VH on the BCC dataset. Column 1: plot of the estimates given by the different combined algorithms across the sequence; the blue dot represents the illuminant estimated on the shot frame by COCOA-VH, while the red cross represents the ground truth. Column 2: sequence frames corrected with the estimate given by the different combined algorithms, respectively SoG, GE1, GE2, GGW, GW, and WP. Column 3: shot frame corrected with the estimate by COCOA-VH. Column 4: shot frame corrected with the ground truth illuminant.

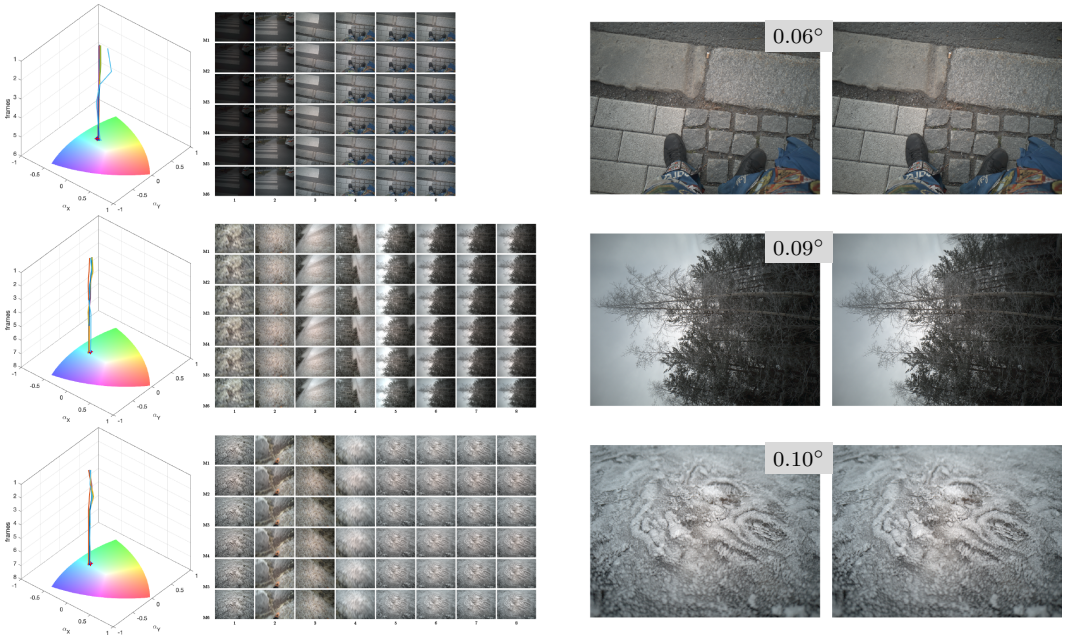


Figure 5.8: Best 3 results of COCOA-VH on the BCC dataset. Column 1: plot of the estimates given by the different combined algorithms across the sequence; the blue dot represents the illuminant estimated on the shot frame by COCOA-VH, while the red cross represents the ground truth. Column 2: sequence frames corrected with the estimate given by the different combined algorithms, respectively SoG, GE1, GE2, GGW, GW, and WP. Column 3: shot frame corrected with the estimate by COCOA-VH. Column 4: shot frame corrected with the ground truth illuminant.

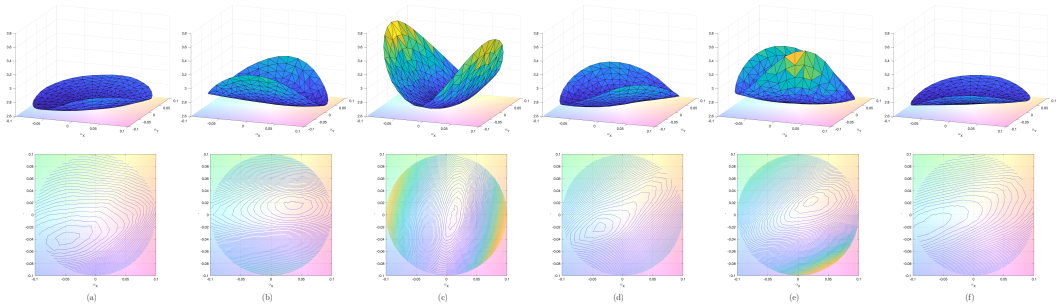


Figure 5.9: Sensitivity analysis of COCOA-IH with respect to the six inputs individually. From left to right: SoG (a), GE1 (b), GE2 (c), GGW (d), GW (f), and WP (g). Top row: surface representing how the average angular error on Shi-Gehler dataset changes when the corresponding input is modified. Bottom row: level curves of the corresponding surfaces in the top row.

input image. These approaches are increasingly computational demanding, memory demanding, and data greedy. In this work have been proposed a fusion strategy that efficiently exploits a variety of simple learning-free algorithms for computational color constancy, combining them in order to provide a lightweight solution that still achieves high performance. The proposed solution, which can be specialized to either the image domain or the video domain, has been thoroughly evaluated in a wide range of experimental setups on standard benchmark datasets. The proposed combination strategy for still images have been compared against other combining solutions achieving top performance, and reaching an illuminant estimation accuracy comparable to more sophisticated solutions. An exploration of different solutions have been done, in order to exploit the temporal component available when analyzing a full video sequence, and as a result a version of the model that exploits a LSTM module to handle varying-length videos have been experimentally defined. This solution has been tested against other algorithms for video color constancy, both in terms of angular error and computational complexity, achieving state-of-the-art performance. Knowing that adaptation to new devices is a real need in the application domain, an analysis on the reduction of the number of training images with respect to the standard dataset partition has been shown, demonstrating how the proposed solution is still able to effectively combine the input methods. Finally, two types of sensitivity analysis have been conducted: one aimed at interpreting the combination strategy learned by the proposed model, and another to understand how a

change in the inputs affect the output. As future developments, the idea is to further explore the possibilities of input combination when dealing with different camera sensors, as well as the combination of more complex input algorithms to further reduce the illuminant estimation error. Moreover, an in depth study of the datasets for auto white balancing alongside algorithm behaviour benchmark has been conducted and presented in [37, 32].

Chapter 6

Contrast enhancement algorithms optimization

The tone-mapping operation is one of the last operations in the camera pipeline, as described in Chapter 3 belonging to the group of the "image space rendering" operations. In general contrast enhancement has the purpose of improving the perceptibility of objects in the scene by enhancing the brightness difference between objects and their backgrounds [59]. Contrast enhancements are typically performed as a contrast stretch followed by a tonal enhancement, although these could both be performed in one step. A contrast stretch improves the brightness differences uniformly across the dynamic range of the image, whereas tonal enhancements improve the brightness differences in the shadow (dark), midtone (grays), or highlight (bright) regions at the expense of the brightness differences in the other regions.

The problem of image contrast correction has been treated with different approaches in the state of the art; gamma correction and histogram equalization [75] are an example of often used approaches. Other approaches may include transform based methods, exposure-based methods and image fusion based methods. Most of the existing approaches for image contrast enhancement rely on the values of one or more parameters to operate on images in order to perform the correction steps. These parameters are in general tuned manually on a set of different possible case scenarios or for specific images.

In this section a user-preferences based framework for contrast enhancement algorithms optimization is proposed. The proposed framework is based on the use of a logistic regressor, capable to model user preferences based on the concept of image acceptability defined by Jaroensri et al.[91]. The logistic

regressor score given to new images is used as objective function for bayesian optimization for the selection of the best parameters of different algorithms for image contrast enhancement. To perform the analysis of acceptability of images, a pretrained VGG-16 CNN is adopted for deep semantic feature extraction; the extracted features are used by the regressor model to perform the regression task, in order to assign a score value to the newly analyzed images. In order to prove the potential application of this approach to the most various contrast enhancement algorithm, a study of the performances of the optimization procedure with three different contrast enhancement algorithm is proposed. In the next section will be presented the framework, alongside the analysis of the data used for the user preference modeling, followed by experimental results performed on test data using a task specific selected metric.

6.1 Related Works

The image contrast correction has been widely studied through the years and several methods for adjusting image contrast have been developed. In general two groups of contrast enhancement algorithms can be identified: global correction algorithms and local ones. The first group of algorithms is made of approaches which globally enhance the content of the images, while the second one contains approaches that differently enhances each pixel with respect to the neighboring ones.

In the first group can be found approaches like Gamma correction and Histogram Equalization. Gamma correction is a simple exponential correction applied to each pixel of an image and which depends on the single parameter γ which corresponds to the exponent value of the function. Histogram equalization [75] techniques work on the image histogram by reshaping it into a different one with uniform distribution property in order to increase the contrast. Multiple versions of histogram equalization technique have been proposed in the years, trying to making it adaptive to the content of the image [132], trying to preserve original image brightness while enhancing the contrast [103, 173, 149, 102], or incorporating models of perception [106, 127, 179]. Other approaches in this first group are the Exposure-based methods [17, 146], which try to adjust the exposure level of an image using a mapping function between the light values and the pixel values of interested objects, and image fusion based methods [85, 113], which combine relevant information from multiple images taken from the same scene in order to produce a final more

informative one.

In the last years few approaches for image contrast enhancement exploiting machine learning have been presented. Here can be found approaches which exploits Neural Networks for the image enhancement [155, 5] and techniques for hyperparaemter selection and optimization of specific algorithms [100, 39].

Finally, a work from Jaroensri et al. [91] in 2015 proposes a way to model user preferences in order to determine when a processing operation in terms of contrast and brightness modification can be considered acceptable or not. This last work inspired the work presented in this section of the thesis.

6.2 Proposed Method

The proposed approach for the optimization of contrast enhancement algorithms is based on the use of a logistic regressor trained on user preferences data. A complete overview of the proposed approach is depicted in Figure 6.1. The first component of this framework is a logistic regressor model based on the work of Jaroensri et al. [91], which has the purpose of model user preferences in terms of image acceptability. The acceptability criteria has been defined on the basis of users subjective definition of image acceptability: in order to model user preferences, Jaroensri et al. [91] selected a random group of images from the Adobe fiveK dataset and modified them in terms of contrast and brightness. Those different versions of each image has been labeled by a pool of users, obtaining a dataset which can be adopted for the modelization of the concept of image acceptability from users point of view. The logistic regressor trained with this dataset has been used for the optimization process of a simple contrast enhancement algorithm. The trained logistic regresson gives a score which become the objective function to be maximized by the optimization process. The image features used by the regressor to perform the image classification are the ones extracted by a VGG-16 deep convolutional neural network [156] pre trained on ImageNet dataset [52]. Due to the recent achievements of deep neural networks, the idea is to exploit deep image representation given by this kind of models to drive the regression process, instead of relying on handcrafted features.

The second part of the proposed framework is the optimization process of the parameters of different algorithms for contrast enhancement. The optimization procedure adopts the user modeled definition of acceptability, represented by the regressor defined previously, to determine for each algorithm

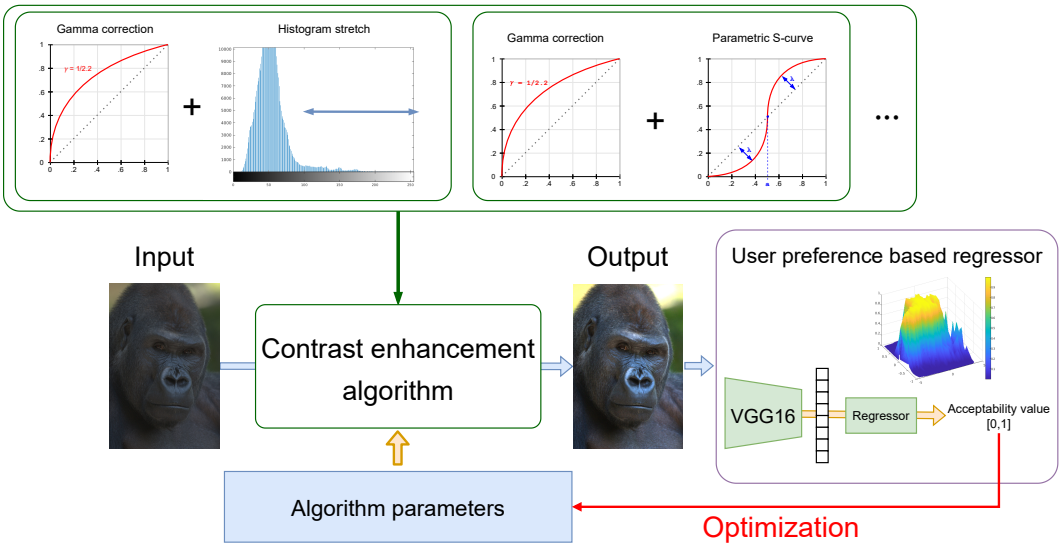


Figure 6.1: Overview of the proposed framework for contrast enhancement algorithms optimization.

the best set of parameters to perform the enhancement operation. Three algorithms have been selected in order to test the optimization procedure. Each algorithm has been optimized in two ways: an optimization by using a training dataset and an optimization per image, performed at processing time.

In this section a description of the logistic regressor model, the algorithms used and the optimization technique will be presented.

6.2.1 User preference based regression

Image processing operations do not necessary lead to images that can be considered good or acceptable. In particular this situation can occur more easily when the enhancement is performed by an automatic procedure, rather than the case in which the processing is guided by a human user. An example of images that can be considered acceptable or not after a processing operation is shown in Figure 6.2. Defining when an image can be considered acceptable is not an easy task, due to the subjective nature of the definition of "acceptable image": considering the examples reported in figure, for a human being it's easy to label the two images on the right as unacceptable due to the degradation of the content of the images, but at the same time its hard to define the degree of acceptability of the other two images on the left. One person

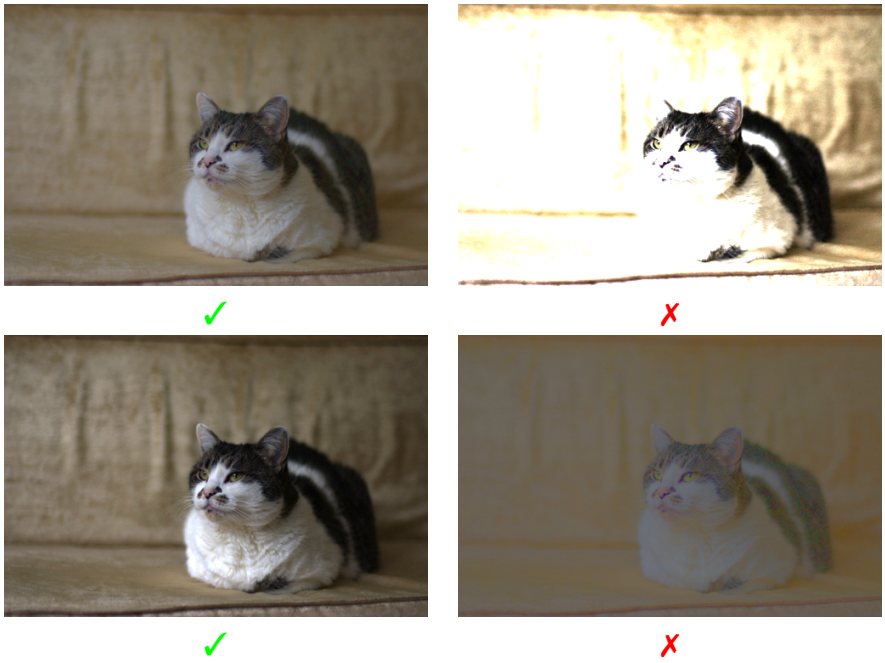


Figure 6.2: Example of images that can be considered as acceptable or not after a contrast enhancement operation. Images originally collected by Jaroensri et al. [91].

may prefer the upper one, due to the more naturalness of the colors, while another one can prefer the bottom one because of the higher contrast given by the processing step.

Starting from these observation, Jaroensri et al. [91] proposed a way to classify images in two classes, using a logistic regression model, based on user preferences. They first defined an image processing procedure which modifies contrast and brightness of images and then collected a dataset made of different versions of the same images, processed with different contrast and brightness parameters. To collect the data for the training of the logistic regressor, the MIT-Adobe Fivek dataset [38] has been used and for each image multiple versions using different combination of *contrast*/*brightness* parameters have been generated. These images have been then labeled by non expert people into two classes, acceptable and not acceptable. In their work, to make data related to each image more understandable, a representation in a Cartesian plane where the contrast and brightness parameters lies on the x and y axes is used. Figure 6.3 shows the scatter plot of the data points collected for an image

of the dataset: each point corresponds to a different version of the image. As can be seen in figure, is possible to identify a "region of acceptability", which vary from image to image. With this new labeled dataset they trained a regressor model, based on low level image content features, able to predict for new images the "region of acceptability" and automatically classify the impact of a processing step over images as acceptable or not. Starting from this configuration, a new logistic regressor has been trained with the purpose to use it as objective function of the optimization step, described in the next section. Instead of using low level image features (such as Fraction of Highlight and Shadow Clipping, Luminance Histogram, RMS Luminance Contrast etc...), a deep convolutional neural network have been adopted, in particular has been used the VGG-16 model. These new models permit a much faster training procedure and inference step with respect to the low level feature extractor previously used in the original work from [91]. The features obtained from the deep neural network are used to train a tree ensemble using Logitboost implementation from MATLAB 2021A. Different versions of the regressor have been trained in relation to the data augmentation operated over the dataset from the work of Jaroensri et al. [91]. This analysis is decribed in section 6.3.1.

6.2.2 Parametric contrast enhancement algorithms

Given the definition of image acceptability, described in the previous section, and given a model of user preferences capable to associate a score of acceptability to an image, is possible to design an optimization procedure in order to maximize the score of an image, given a certain enhancement algorithm.

The proposed approach for algorithm optimization can be applied to any kind of algorithm for contrast enhancement whose performance depends on one or more parameters. In order to prove the efficiency of the user preference driven optimization, three algorithms for contrast enhancement have been tested. The first algorithm consists of a simple combination of two global operators: first the image is processed using a gamma function (with parameter γ), and then an histogram stretching operation (with two parameters max_value and min_value) is performed over the output of the gamma function. The second one is a slightly different configuration which adopts a parametric S-curve function defined by Kang et al. [100], dependent on two parameters, λ, which determines the slope of the S-curve, and a which determines the flex point of

the S-curve. The formula used by Kang et al. to specify the S-curve is:

$$y = \begin{cases} a - a(1 - \frac{x}{a})^{\lambda} & if\ x \leq a \\ a + (1 - a)(\frac{x-a}{1-a})^{\lambda} & otherwise \end{cases} \tag{6.1}$$

The last algorithm used is the one called Local Contrast Correction (LCC) method by Schettini et al. [148], which is a local contrast enhancement algorithm based on the use of bilateral filter to determines areas of images to be lighten or darken. The parameters of this algorithm are α, which determines the exponent of the exponential function applied to the luminance channel of the input images, and the two sigma values σ_1 and σ_2 which are the parameters of the bilateral filter function. This last algorithm works in YCbCr color space.

The optimization of the algorithms parameters has been performed using bayesian optimization approach (implementation from MATLAB). The procedure selects a set of values for each parameter, and then performs the contrast enhancement step. The resulting image is then processed by the combination of the deep CNN, which extracts the deep semantic features of the image, with the trained logistic regressor, assigning an acceptability score between 0 and 1. The features extracted using the VGG-16 comes from the last convolutional layer, after which an average pooling operation has been applied in order to bring spatial resolution to dimension 1×1. The objective of the bayesian optimization is the set of parameters that maximize the score given by the logistic regressor at the newly enhanced image. Two versions of each algorithm have been optimized: one by optimizing over a training set and one which is the optimization per image. In he first case the optimization is performed offline, using a training set and optimizing the parameters on those data. The resulting set of parameters is then used for the corresponding algorithm at inference time. In the second scenario the opimization is performed directly over the images at processing time. This operation leads to a set of parameters specifically optimized for each new image.

6.3 Experimental setup

6.3.1 Dataset for user preferences modeling

In order to train the logistic regressor to be used as acceptability metric the dataset presented by Jaroensri et al. [91] has been adopted. The dataset is

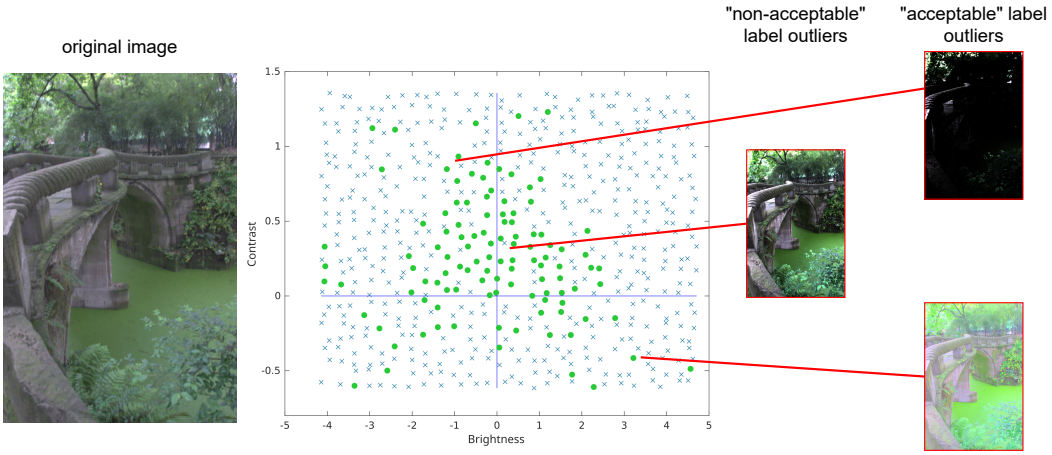


Figure 6.3: Example of possible outliers in the original version of the dataset. As can be seen there are multiple data points labeled in a misleading way with respect to the actual state of the image. Green dots correspond to images with "acceptable" label, blue crosses correspond to images with "non acceptable" label and the center of two blue axis correspond to the original image at coordinates $[0, 0]$.

made of binary human judgement of image quality of 500 images adjusted to various configurations using brightness and contrast settings. The images were taken from the MIT-Adobe FiveK Dataset [38]. The total amount of data points collected is 301320, of which 241148 constitute the training set while the remaining 60174 are used as test set. The original version of the dataset offers for each one of the original 500 images, around 600 processed versions. Those versions have been collected by first applying to the raw image data the white balance and saturation of one of the expert retouchers from the MIT-Adobe FiveK, then by performing contrast and brightness adjustments using the procedure described in the original work [91].

In order to use this dataset for the training of the logistic regressor, which is used as model of user preferences, two operations have been performed:

1. a data points cleaning procedure to remove outliers for the "acceptable" and "non acceptable" labels.

2. a data augmentation procedure to balance positive and negative labels.

Due to the presence of an high amount of outliers, a cleaning procedure has been performed, before augmenting the dataset. As can be seen in Figure

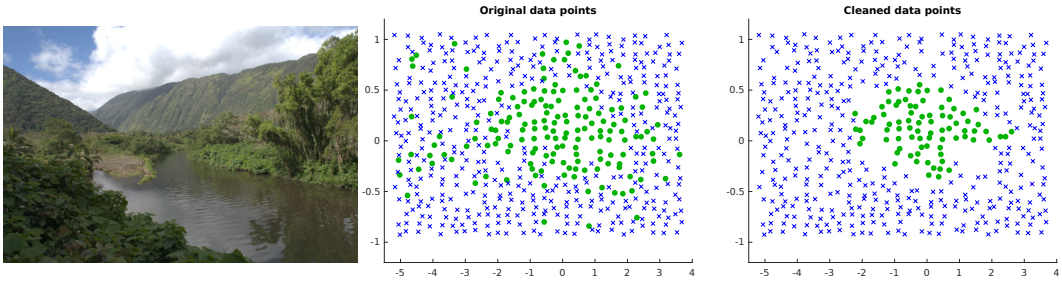


Figure 6.4: Example of data points distribution in contrast/brightness space, before (left) and after (right) the data cleaning procedure. As can be seen most of the outliers have been removed. Green dots correspond to images with "acceptable" label, blue crosses correspond to images with "non acceptable" label.

6.3, in the areas where most of the images are labeled as acceptable ones, few points with the non-acceptable label can occur. The same situation arises in the areas with high concentration of points labeled as "non-acceptable", having in this case images mislabeled as "acceptable". Looking at the images corresponding to those point it's easy to see how those points are misclassified (Figure 6.3, on the right). In order to train the regressor model those points have been removed from the training dataset. The cleaning procedure consists of a density analysis in the brightness/contrast space, performed by dividing the space in bins and counting the amount of positive labels in each bin. Using a fixed threshold the points have then been removed obtaining a new set of data points for each image. An example is shown in Figure 6.4.

Secondly, since in the original version the 600 data points per image collected by Jaroensri et al. present an high disparity in the labels (around 75% of the data points are labeled as not acceptable, while the remaining 25% are labeled as acceptable), a data augmentation procedure has been applied in order to balance the amount of positive and negative labels in the dataset. In the specific case, for each of the original 500 images, the points with positive label (acceptable editing) have been duplicated in order to reach the same amount of points with negative label (non-acceptable editing).

The results achieved with the different versions of the training dataset can be seen in Table 6.1. Due to the fact that the test set presents the same unbalanced nature of the training set in terms of labels, macro accuracy has been used to select the best configuration to be used for the optimization procedure.

Table 6.1: Analysis on the impact of the data augmentation and datapoint cleaning procedure. Micro and Macro accuracies are reported due to the unbalanced nature of the test set in terms of positive and negative labels.

Predictor	Data balancing	Outliers removed	Micro accuracy(%)	Macro accuracy(%)	Positive accuracy(%)	Negative accuracy(%)	Precision	Recall	F-Score
			78.85%	64.26%	35.97%	92.56%	0.607	0.360	0.452
VGG16	✓		75.01%	73.19%	69.66%	76.72%	0.489	0.697	0.575
	✓	✓	74.98%	73.33%	70.13%	76.54%	0.488	0.701	0.576

6.3.2 Dataset for optimization procedure test

The optimization procedure with the three algorithms has been performed using the the test set of the dataset proposed by Jaroensri et al. [91]. For each image the version corresponding to the origin of the contrast/brightness space has been selected in order to be used as input image. Those images, corresponding to the point $[0, 0]$ (which will be called $image_{[0,0]}$), are the raw images from the Adobe fiveK dataset, on which only few steps of the processing pipeline have been applied. In the original work the RAW images have been white balanced and saturation have been adjusted using the parameters given by one of expert from the Adobe fiveK dataset experts pool. This version is considered the starting point for the contrast enhancement operation.

Starting from the $image_{[0,0]}$ and a set of random values for the algorithm parameters, each of the three algorithms have been optimized using the result of the logistic regressor as objective function. The optimization has been performed using bayes optimization procedure for a total amount of 30 iterations. In the case of optimization on training dataset, at each iteration 120 random images are processed and evaluated. In the case of per-image optimization, the procedure processed the same image for the total amount of 30 iterations, always starting from the $image_{[0,0]}$.

6.3.3 Optimization evaluation metric

The evaluation of the results obtained from the optimization procedure has been performed using a full reference metric. In order to select the most suitable metric to evaluate image contrast and brightness variation, exploiting the procedure and the TID2013 dataset proposed by Ponomarenko et al. [133], a set of possible suitable metrics have been analyzed. The TID2013 is a dataset intended for evaluation of full-reference image visual quality assessment metrics. The TID2013 contains 25 reference images and 3000 distorted images.

Table 6.2: Results of the correlation test performed on the images and MOS provided in the TID2013 dataset. For each metric are reported the score of the Spearman and Kendall correlation indexes.

Metric	Spearman correlation	Kendall correlation
VIF-P	**0.85949**	**0.63792**
PSNRHMA	0.65654	0.47296
PSNRHA	0.64522	0.47038
FSIM	0.4719	0.35828
MSSIM	0.46838	0.35522
FSIMc	0.46804	0.34966
PSNRc	0.46085	0.30902
NQM	0.45996	0.29276
SSIM	0.45513	0.34408
PSNRHVS	0.44283	0.30799
PSNR	0.44142	0.30825
PSNRHVSM	0.43625	0.30076
WSNR	0.42387	0.29921
VSNR	0.35144	0.23416

Reference images are obtained by cropping from Kodak Lossless True Color Image Suite [62]. In order to determine which metric is the most suitable for contrast and brightness enhancement analysis, each of the metrics analyzed in the work of Ponomarenko et al. has been compared by only using the score given to the images distorted under the label "Contrast change". The scores given by each metric are compared using the MOS collected for each image by Ponomarenko et al. and by calculating the Spearman and Kendall correlation scores. The collected correlation scores are reported in Table 6.2. From this analysis the Visual Information Fidelity metric (called VIF-P) [152] has been selected as the most suitable for the evaluation of the optimized algorithms performances.

Since the VIF-P metric is a full reference metric, target images are needed for the evaluation of the algorithms performances. Since the acceptability is subjective concept, and since the dataset from Jaroensri et al. [91] provides for each image multiple enhanced versions but not a target one, it is necessary to select a reference image to perform the comparison. In order to select reference images for each input one, a simple selection procedure has been defined: considering only the positive labels, the version closest to the average contrast/brightness data point has been selected as "average user preferred image". These images selected using this procedure are used as target for the evaluation procedure. The higher is the value assigned by the metric to an image, the closer the analyzed image is to the reference one.

Table 6.3: Results in terms of VIF-P score and percentage of image improved over the test set from [91]. P-values obtained for the t-test have been provided to show statistical significance of the experiments.

Optimization	Method	Avg VIF-P	STD	p-value	Percentage of improved images
-	Original input	0.8162	0.1526	-	-
on dataset	Gamma correction + Histogram Stretch	0.8765	0.1537	0.0000	95%
	Gamma correction + S-curve	0.8166	0.1542	0.9401	52%
	Local Contrast Correction (LCC)	0.8645	0.1349	0.0000	80%
per image	Gamma correction + Histogram Stretch	0.8582	0.1593	0.0000	85%
	Gamma correction + S-curve	0.8405	0.1370	0.0069	66%
	Local Contrast Correction (LCC)	0.8720	0.1444	0.0000	84%

6.4 Experimental results

Here are reported the performances obtained by the three algorithms optimized. In table 6.3 are reported the average VIF-P score obtained on the test set before the contrast enhancement step and the score obtained by the output of each of the optimized algorithms. Two groups of scores are reported, corresponding to the optimization on dataset and the optimization per image.

As can be seen from the table, the application of the optimization procedure improves the quality of the output images with respect to the input ones. Analyzing in details the three algorithms, different behaviours can be observed. While for both the S-curve and LCC algorithms the per-image optimization brings higher performances with respect to the optimization on dataset, the behaviour with the gamma correction with histogram stretch operation is the opposite. However, with the only exception given by the S-Curve approach optimized on the dataset, the optimization procedure brings an improvement in terms of average VIF-P. Looking at the percentage of images of which quality improved, all of the algorithms have performed an improvement at least in 50% of the cases. The best result in this sense is given by the Gamma correction + Histogram Stretch algorithm which improved the quality of the 95% of the test images.

A more in detail picture of the distribution of differences of quality score between the target images and the ones processed by the three algorithms is shown in Figure 6.5. Here are reported the histogram of the differences between the enhanced images score and the input images scores. Has can be seen in this representation, Gamma correction + Histogram Stretch algorithm brings the most noticeable improvement, alongside the LCC algorithm in the

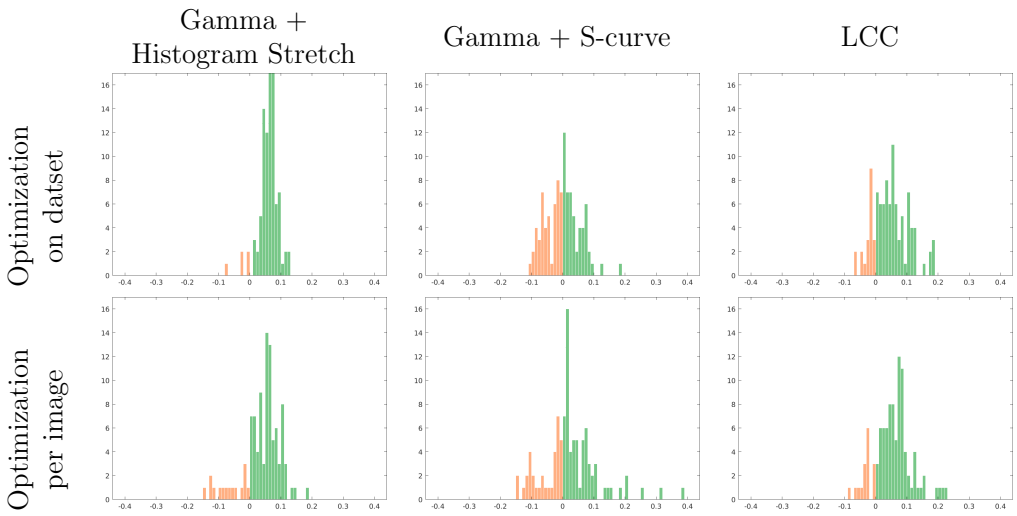


Figure 6.5: Distributions of the differences in VIF-P values between the enhanced images and the original input ones. From left to right are reported Gamma + Histogram Stretch, Gamma + S-curve and LCC algorithms. Green bars represent cases where the VIF-P difference is positive while orange ones represent cases where the VIF-P value of enhanced images is the same or lower with respect to the input ones.

per-image optimized version.

These results prove the effectiveness of using a user preferences driven optimization procedure to improve the contrast and brightness in images, and demonstrates how it can also be applied to different kind of algorithms.

6.5 Summary

In this section an optimization procedure for contrast enhancement algorithms, based on a model of user preferences has been presented. The proposed approach is composed of two main blocks: a logistic regressor, which models user preferences and which is used to guide the optimization process, and the actual algorithm to perform contrast enhancement. In order to model user preferences a deep convolutional neural network has been exploited, in the specific case a VGG-16, to extract the image features which have been used to train the logistic regressor. A dataset augmentation procedure has been used in order to clean the datset used for training and an evaluation of the performances has

been provided. With the proposed logistic regressor, three different contrast enhancement algorithms have been optimized, using Bayesian Optimization approach. The optimization has been performed in two ways: optimization on a training set and optimization per image. In the first case a single set of parameters is obtained, making the inference operation a simple application of the algorithm with the obtained parameters, while in the latter case the optimization is performed at inference time, giving for certain algorithms better results but increasing the inference time, due to the online optimization procedure.

In order to analyze the results of the optimized models, an image quality assessment metric has been selected. A procedure to determine which metric suits most for the task has been adopted and as a result of the application of this procedure the VIF-P metric has been selected and used for testing the optimized algorithms. Finally a procedure to select an average preferred image have been presented. Using the images selected with this procedure as target, an analysis in terms of average performance improvement has been provided, alongside a more detailed analysis of per image improvement. Experimental results show the effectiveness of the proposed optimization procedure, showing how for each considered algorithm, in both the optimization scenarios, an improvement in terms of VIF-P has been obtained.

Chapter 7

JPEG blind artifact reduction

Image compression corresponds to the last step in the digital camera processing pipeline, before storing the image collected. Image compression represents a very active research topic due to the high impact of the data in a large number of fields, from image sharing on the web to the most specific applications involving the acquisition of images and transfer to elaboration nodes. Specifically, image compression refers to the task of representing images using the smallest storage space possible.

Compression algorithms play a key role in saving space and bandwidth for the memorization and transfer of large amounts of images. Two different compression paradigms exist: the former is lossless image compression, where the compression rate is limited by the requirement that the original image must be perfectly recovered; the latter, more diffused, is lossy image compression, where higher compression rates are possible at the cost of some distortion in the recovered image. Among the lossy compression algorithms, the most diffused and used is the JPEG compression algorithm.

The JPEG compression algorithm first converts the original RGB image into YCbCr color space and processes the luma (luminance component) and chroma (chromaticity component) channels separately. It divides the luma channel of an input image into non-overlapping 8×8 blocks and performs the Discrete Cosine Transform (DCT) on each block separately while downsampling the chroma components with a bilinear filter. The DCT coefficients obtained from the luma channel are then quantized based on *quantization tables* and adjusted using the user-selected *Quality Factor*. The image is then reconstructed from the quantized DCT coefficients by using the inverse DCT. The described JPEG encoding operation introduces three kinds of artifacts in

the recovered images, related to the quality factor used for the compression:

i. blocking artifacts, which come from the recombination of the 8×8 blocks, that are independently compressed without considering the adjacent blocks;

ii. ringing artifacts, which are most visible along the edges and are related to the coarse quantization of the high-frequencies components;

iii. blurred low-frequencies areas, which is also related to the compression of the high-frequencies in the DCT domain.

The presence of these kinds of artifacts represents a problem since the general quality of the images is degraded resulting unpleasing for normal users and for generic applications (e.g. projection, print, etc.), or even useless for computer vision applications where the loss of information can be potentially critic for the task [53, 27].

With the purpose of reducing these artifacts, in the last years, a lot of JPEG artifact reduction algorithms have been proposed. These methods include both traditional image processing pipelines [114, 142, 174, 8, 61, 90, 172, 116] and machine learning approaches [54, 196, 41, 182, 70, 115, 197, 198, 162, 191], both making great steps in the restoration of corrupted images. However, these methods suffer from two main limits: the first one is that they need to train a different model for each possible Quality Factor (QF), making them not generally applicable to general images downloaded from the web unless the QF used for compression is known; the second one, is that the great majority of methods in the state of the art restores just the luma channel or do not fully exploit the knowledge about the JPEG compression pipeline.

To address these problems, in this section I propose a new method for the blind universal restoration of JPEG compressed images, based on machine learning, specifically on convolutional autoencoders. The proposed approach consists of two deep autoencoders respectively used for luma and chroma restoration, that are able to restore images independently from the quality factor used for the compression.

In the next sections the methodology will be described and an in depth analysis of the robustness of restoration results at different Quality Factors will be presented. Also a comparison with the state of the art approaches will be discussed.

7.1 Related Works

The task of JPEG compression artifacts removal has been faced in different ways in the past years. The existing proposed methods can be broadly classified into two groups: traditional image processing methods and learning-based methods.

The first group includes methods based on traditional image processing techniques working both in the spatial and in the frequency domain. For spatial domain processing, different kinds of filters have been proposed, with the intent of restoring specific areas of the images such as edges [114], textures [142], smooth regions[174], etc. Algorithms usually rely on information obtained by the application of the Discrete Cosine Transform (DCT) transform [8]. SA-DCT, proposed by Foi *et* al. [61], attempts to reconstruct an estimate of the signal using the DCT of the original image together with the spatial information contained in the image itself. However, SA-DCT is not capable to reproduce details like sharp edges or complex textures. To overcome this limit different restoration oriented methods have been proposed, like the Regression Tree Fields based method (RTF) [90]. The RTF uses the results of SA-DCT to restore images, taking advantage of a regression tree field model.

Following the success of the application of Deep Convolutional Neural Networks (Deep-CNNs) in image processing tasks, such as image denoising [196] and Single-Image Super-Resolution [55], Deep-CNNs have been applied with success to JPEG compression artifact removal task. The basic idea behind Deep-CNNs is to learn a function to map a set of images from an input distribution, to the desired output one [77]. In the artifact removal case, the objective is to map degraded images into another distribution without the presence of the noise. The trained neural network obtained at the end of the training process represents an approximation of the desired function for the translation of the images from a distribution to another one.

The first attempt with this kind of model has been done by Dong *et al.*[54] who proposed the ARCNN, a model inspired by SRCNN [55], a neural network for Super-Resolution. This first attempt has been followed by DnCNN [196], a CNN for general denoising task that has also been used on JPEG compressed images, and CAS-CNN [41], a model proposed by Cavigelli *et al.*, who presented a much deeper model capable to obtain higher quality images. Wang *et al.* proposed D3 [182], a deep neural network that adopts JPEG-related priors to improve reconstruction quality which obtained an improvement in speed and performances with respect with to the previous models.

In 2018 several new models for JPEG artifact removal have been presented, showing interesting improvements in the quality of the results. Liu *et al.* [115] proposed a Multi-level Wavelet CNN (MWCNN), a model based on the U-Net architecture [143], trained and used for multiple tasks: compression artifact removal, denoising, and super-resolution. Zhang *et al.* [197] developed DMCNN, a Dual-Domain Multi-Scale CNN, which gains higher results quality than the previous works, by using both pixel and frequency (*i.e.* DCT) domain information. Galtieri *et al.*[70] and Yoo *et al.* [191] tried to address the problem of JPEG compressed image restoration by employing a generative adversarial network (GAN)[76] for artifact removal and texture reconstruction. Lastly, two interesting methods have been proposed: S-Net, by Zheng *et al.* [198], a method based on a "greedy loss architecture" to train deeper models capable to outperform the previous state-of-the-art, and JBCBCR, the most recent method proposed by Chen *et al.* [44], which restores JPEG images in YCbCr space, exploiting the correlation between the information from both luma and chroma components of the images.

7.2 Proposed Method

The methods in the state of the art mainly suffer from two limits: the first one is that each machine learning model needs to know the JPEG compression Quality Factor (QF) of each input image to properly restore a compressed image; the second one is that the great majority of them are capable to restore only the luma channel without considering the chroma components. Only the two most recent methods try to restore also the colors of the images: S-Net [198] which works on RGB space, and JBCBR [44] which tries to exploit the distribution of the artifacts coming from the JPEG pipeline, working in YCbCr space.

In this work, a method able to overcome both the aforementioned problems is proposed. The first problem has to do with the way the models are trained: all of the previously existing methods make the implicit assumption that the compression quality factor QF that has been used to compress the input images is known at restoration time. In fact, most of the previous models present networks trained on datasets compressed on specific quality factors (the most common being QF $= 10, 20, 30$ and 40). This way of training the models leads to two limits:

- the models are capable to correctly restore only images at a specific

QF, with the consequence that specific training for each quality factor is needed;

- the QF used for the compression of the images is needed in order to select a model and correctly restore the images since each model is trained at a specific compression QF. This is usually an unknown information for images coming from unknown sources (e.g. downloaded from the web), thus largely limiting the usability of the models.

In order to overcome the necessity to know the compression quality factor, we train the model on a dataset containing images compressed at different QFs: this will make the model more generic and able to restore images taken in the wild, i.e. without knowing the actual QF used. This objective poses a challenge, since the training of such a quality independent model is much harder than training on a single quality factor: for example, the model has to learn if a strong edge present in the image is a JPEG artifact belonging to an image with a low QF and thus should be corrected, or a real edge belonging to an image with a high QF and therefore should be preserved. Preliminary experiments, in fact, showed that just training a state of the art method with images compressed at different QFs significantly deteriorates the restoration performance with respect to the same method trained for a single QF.

The second problem concerns the way the previous models restore the images: almost all of the previous state-of-the-art methods are trained on the luma channel (Y channel of the $YCbCr$ space) of the images. This approach is based on the fact that the JPEG compression algorithm applies the DCT to the Y channel, introducing ringing and blocking artifacts on the luma channel, while the other Cb and Cr channels are just sub-sampled the bicubic interpolation. The design and training of a model for the specific restoration of the luma component and its subsequent application for the restoration of the chroma components (as done for example by ARCNN [54]), introduces chromatic aberrations and artifacts in the final result. S-Net[198] and JBCBR [44] are the only methods considering this problem and instead of training a model for the restoration of just the luma component, they work respectively in RGB and YCbCr color spaces for restoring both luma and chroma.

To overcome this second limit and restore also the color information, similarly to Chen *et al.* [44] the knowledge of how the JPEG compression pipeline works has been exploited and so has been proposed the use of two models for the image restoration in $YCbCr$ space: the first model restores the Y channel; the second model then uses the result as a *Structure Map* (i.e. a guide) for

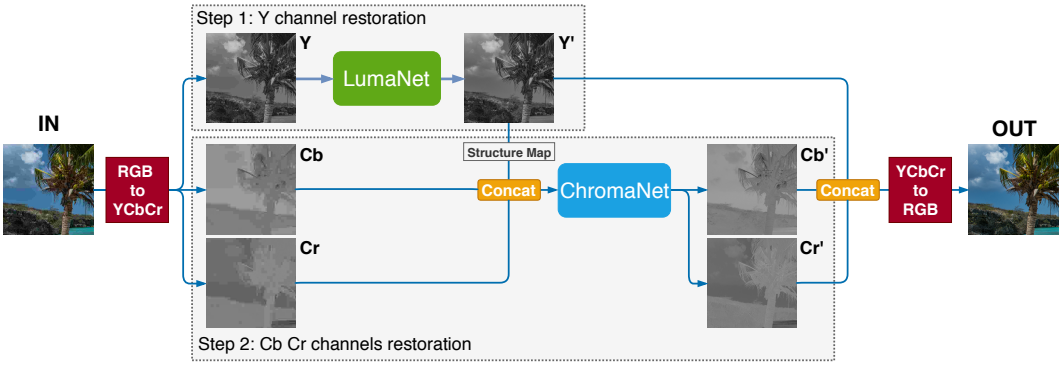


Figure 7.1: Schematic representation of the proposed method: the input image is first converted to *YCbCr* color space. The *Y* channel is restored with the LumaNet and the result *Y'* is concatenated with the original *CbCr* channels to restore *Cb'Cr'* with the ChromaNet. Restored *Y'Cb'Cr'* channels are then converted back to RGB color space.

the restoration of the chroma components. A schematic representation of the proposed method is depicted in Figure 7.1. The input RGB image is converted into YCbCr color space and the Y channel is separated from the Cb and Cr channels. The Y channel is restored with a dedicated network, and the result is channel-wise concatenated with the original Cb and Cr channels. This stack is processed with a second network that produces as output the restored Cb and Cr channels using the restored Y channels as a guide. The restored Y, Cb, and Cb channels, the former coming from the first network and the latter ones coming from the second network, are channel-wise concatenated and converted from YCbCr to RGB to produce the final output.

7.2.1 Luma and Chroma Restoration Model

The vast majority of learning based methods for JPEG compression artifact removal in the state of the art [54, 196, 41, 182, 115, 197] focus exclusively on the luma component of the images. Generally, these methods perform the compression artifact removal working on the *Y* channel of the images, after converting them in *YCbCr* color space. These approaches do not take into consideration the chroma aspects of the images, generating results with aberrations in RGB space and low perceptual quality.

The JPEG compression algorithm, when operating with very low compression quality factors (e.g. QF $\leq$ 20) tends to change the colors of the input

images in two different ways: hue change and spatial location change. As can be seen in Figure 7.2, in the compressed version of the *Cb* and *Cr* channels, as expected, the color resolution is reduced and also, for some elements, the color position does not correspond to the one in the original uncompressed image.

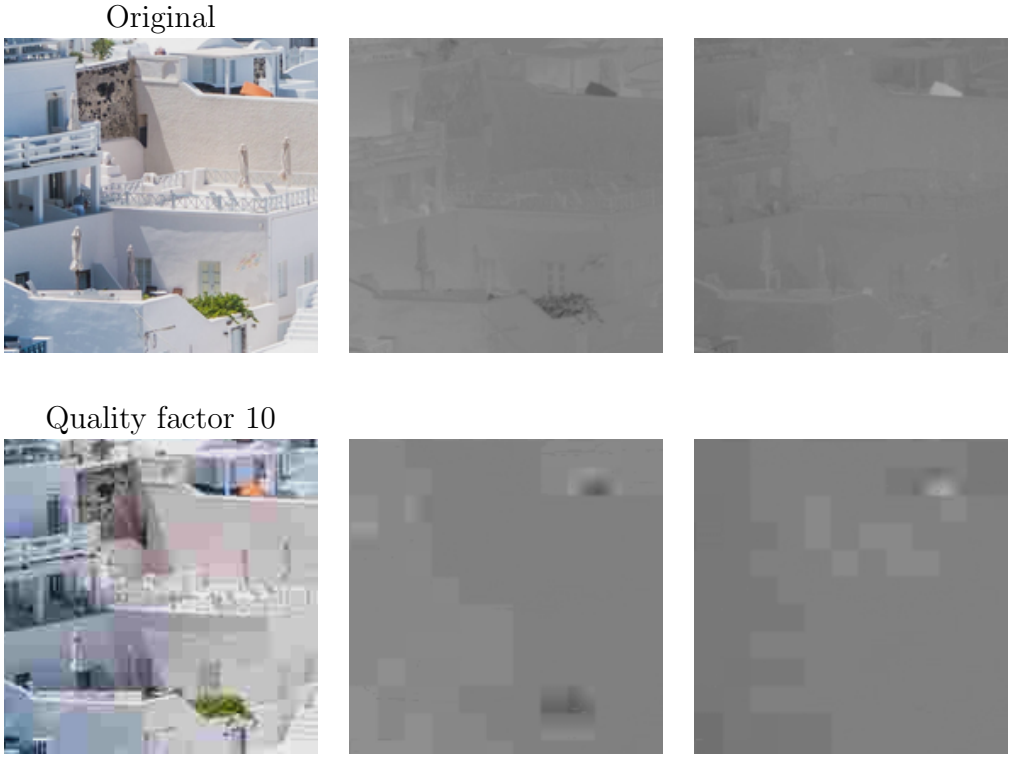
In the last years, only two models tried to restore the images considering also the chroma components. These methods are S-Net [198] and JBCBR [44]. While the first one tries to restore the information contained in the images in RGB space, the second one exploits the YCbCr space, the same used for the compression by the JPEG algorithm.

Keeping the above considerations in mind the method has been designed for restoring both luma and chroma components of the compressed images (see Figure 7.1). The method consists of two steps: the first step, after the conversion of the input image into *YCbCr* color space, involves the restoration of the *Y* channel alone, using a first model named LumaNet, and produces *Y'* as output. The second step concatenates *Y'CbCr* along the channel dimension and uses a second model named ChromaNet, to restore the *CbCr* channels. This second step uses *Y'* as a map of the structures present in the image (i.e. a sort of guide) to condition the second network to recover the color hue and contours, and produces *Cb'Cr'* as output. The final output is obtained by concatenating *Y'Cb'Cr'* and converting them back to RGB. Both LumaNet and ChromaNet are two different deep CNN Autoencoders both exploiting a new revisited version of the Residual Blocks [82].

7.2.2 Deep Residual Autoencoder Architecture

Autoencoder architectures have been widely used in image processing tasks like image-to-image translation [89], Super-Resolution [193], image inpainting [185] and rain removal [136]. Autoencoders for image processing tasks generally present a structure made by three parts: the encoder, which extracts features from the n-dimensional input (usually one or three channels); a central part, that performs feature processing; and the final decoder, which decodes the processed features into the output image having the desired dimensions. Figure 7.3 shows a schematic representation of the proposed model, while a more detailed description of its architecture is reported in Table 7.1.

The encoder, which consists of two convolutions followed by Leaky ReLU activations, is followed by a central part for feature enhancement consisting in a sequence of *Residual-in-Residual Dense Blocks* (RRDB) [178], a modified version of the well known residual blocks originally introduced in the ResNet



(a) Color (c) Cb channel (d) Cr channel

Figure 7.2: Visual example of how the JPEG compression algorithm, when operating with very low compression quality factors, changes the colors of the input images in two different ways: hue change and spatial location change.

architecture[82], that have been shown to perform well in other image processing tasks, e.g. image super-resolution [112, 178]. The RRDBs blocks combine multi-level residual learning and dense connection architecture: the RRDBs are designed without the use of the Batch Normalization and the application of the residual learning on different levels. The RRDBs are shown in Figure 7.4: each RRDB is made of five *Dense Blocks*, which use only convolutions with Leaky ReLUs activation and dense skip connection structures, combined together with other skip connections. Finally, the decoder is designed in a symmetrical way with respect to the encoder part.

The same architecture has been used for both the networks for luma and chroma restoration, but with some differences:

- different depth in terms of number of RRDBs used in the central part;

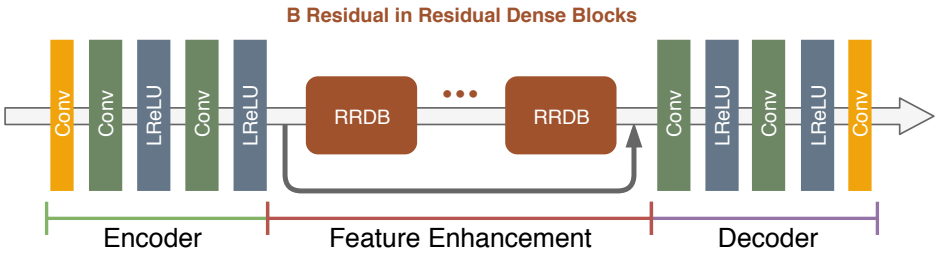


Figure 7.3: Graphical representation of the architecture of the autoencoders used for both the luma and chroma restoration.

- different feature extraction from the input in the encoder part.

For the restoration of the luma (Y channel), the number of central RRDBs is set to five, while for the *CbCr* restoration the number of RRDB is decreased to three. The second and more important difference is in the first layer of the *CbCr version* of the network, which is a *3-dimensional convolutional layer*. Considering that the input of the *CbCr-Net* is the concatenation (along the channel dimension) of the restored *Y'* channel with the *Cb* and *Cr* channels, we decided to use a 3D convolution to make the model capable to correlate information about color and structures with the use of the same kernels for all the information coming from the three input channels. The output of this second network are the two restored *Cb* and *Cr* channels, which are then concatenated with the restored *Y'* channel, in order to obtain the complete restored image.

In order to improve the quality of the generated results, as well as to make the training process more stable, the proposed architecture includes the following design choices:

- removal of Batch Normalization (BN) layers from the Residual Blocks;
- use of a *residual scaling* parameter in each Residual Block;
- initialization of the model weights using a scaled version of the Kaiming initialization[81].

The removal of the batch normalization layers has been proved, in image Super-Resolution [112] and image deblurring [129] tasks, to increase the performances for the generation of images in terms of quality indexes (PSNR and SSIM [181]). The removal of the BN layers, which improve the stability of the

Table 7.1: Detailed architecture of the autoencoders used for both the luma and chroma restoration. The number of RRDBs is $B = 5$ for the Y-Net and $B = 3$ for the CbCr-Net.

	Layer	Filter size, Stride, Padding	output channels
	Conv2D	1x1, 1, 0	64
Encoder	Conv2D	3x3, 1, 1	64
	LReLU	-	64
	Conv2D	3x3, 1, 1	64
	LReLU	-	64
		RRDB x B	
Decoder	Conv2D	3x3, 1, 1	64
	LReLU	-	64
	Conv2D	3x3, 1, 1	64
	LReLU	-	64
	Conv2D	1x1, 1, 0	1 / 2
	Tanh	-	1 / 2

training and the generated image appearance, makes, on the other hand, the training of deep networks more difficult. To solve that issues two solutions have been proved to work well: the so-called residual scaling (in the model set to 0.2), to scale each residual in order to not magnify the input image in a wrong way, and a small weight initialization, obtained by the application of the Kaiming initialization, presented by He *et al.*[81], scaled by a factor 0.1. As can be seen in Figure 7.4 the residual scaling is applied to the higher level of the residual learning architecture, i.e. on the output of each dense block and at the end of the RRDBs.

7.3 Experimental Setup

The training of the proposed method leads to two different Deep-CNNs respectively for the restoration of the luminance and chroma components of JPEG compressed images at generic quality (i.e. QFs). In order to evaluate the results, the proposed model have been compared with the state of the art in four different experimental setups:

1. *known QF* luminance restoration: comparison with the state-of-the-art

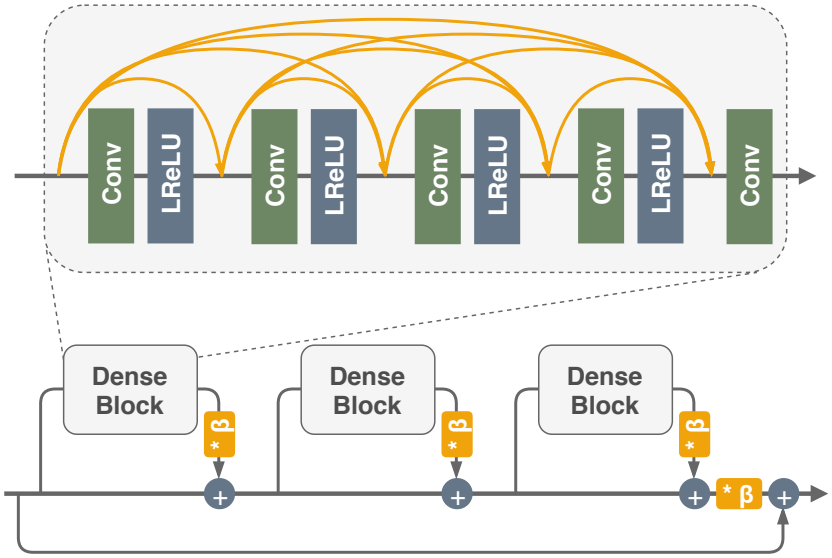


Figure 7.4: Schematic representation of the architecture of the Residual-in-Residual Dense Block (RRDB) [178].

methods which work only on the Y channel of the input images;

2. *unknown QF* luminance restoration: comparison to test the ability of the models to restore images at intermediate QFs never seen during training;

3. restoration of areas with high and low details density: evaluation of the performances of the state-of-the-art methods and the proposed one over specific areas of the images, by dividing the images in patches classified on high-to-low frequency (DCT domain) and high-to-low detail density;

4. color restoration: evaluation of the color restoration capability of the model on the images converted in RGB space after the processing.

7.3.1 Dataset

The dataset used for training is the DIV2K dataset, a collection of high-quality images (2K resolution), presented during the NTIRE2017 challenge [7] for image restoration tasks. This dataset is made of a total amount of 900 images: 800 are used for training while the remaining 100 are used for validation. The complete dataset contains also 100 images for testing. The

ground truth of this last part has not been released after the challenge, and therefore are not used in this paper.

With the purpose of increasing the amount of different texture and pattern to show to the model during training, the DIV2K dataset have been combined with the Flickr2K dataset [165], a collection of 2650 high-quality images (same resolution as the DIV2K) collected from Flickr website.

In order to train the models on different quality factors, for each image in the dataset have been compressed with 10 different compression levels, corresponding to the quality factors between QF = 10 to QF = 100, with step 10. The images have been compressed with the MATLAB standard library function. In the training phase of the model, as a pre-processing operation, the compressed images are read and converted into YCbCr space using the Python Scikit-Image library (v0.14.0). The compressed version of the training dataset contains 8000 images. The same operation has been applied to the Flickr2K dataset for a total amount of 34k training images.

The evaluation of the model, for the luminance channel restoration, has been done on the LIVE1[181], Classic-5, BSD500 [11] and Kodak Lossless True Color Image Suite[62], four benchmark datasets widely used for JPEG artifact removal algorithm evaluation. For the evaluation of the behavior of the models with the *unknown compression quality factor* we adopted the SDIVL [50], a dataset proposed for Image Quality Assessment task.

The evaluation of the color channels restoration has been done using the Kodak Lossless True Color Image Suite[62], in the same way that has been done by Chen *et al.* [44].

7.3.2 Evaluation metrics

The globally adopted metrics for the evaluation of the quality of images in artifact removal tasks are PSNR, PSNR-B [190] (which focus the evaluation on the blocking artifacts) and SSIM [181] indexes. For all of these three measures, a higher value means better results. The PSNR and PSNR-B indexes give information about the quality of the images in terms of noise and perceived quality, with PSNR-B taking into consideration also the blocking artifacts; SSIM index is an indicator of the quality of edges and structures contained in the image. For all the three indexes considered a higher value means that the content and the structures in the reconstructed image are more similar to the ones in the target image.

7.3.3 Training Details

All the training phase has been done on an NVIDIA Titan V GPU with 8 GB of memory using PyTorch framework at version 0.4.1. The mini-batch size has been set to 8 and each input image has been cropped to a patch size of 100×100 pixels. During the experiments we tried to train the network with different crop sizes (32×32, 50×50, 100×100 and 400×400), observing how training deeper networks with bigger patch size gives a boost on performances over both PSNR and SSIM indexes.

We also explored the use of different numbers of RRDBs in the model: we observed how with deeper models, using this specific kind of residual blocks, the results got better and better, increasing the PSNR and SSIM values on the validation set. The final structure uses five RRDBs for the Y channel restoration model and three RRDBs for the CbCr model, where each convolution has 64 filters. We found this configuration to be the best one, with respect to the patch size, the amount of RRDBs, the number of filters and the limits due to the memory offered by the used board.

We trained the model using Adam optimizer [104] with $\beta_1 = 0.9, \beta_2 = 0.999$, with learning rate initialized at 2×10^{-4} decreased after 200 epochs of training by a factor of 2. The training has been performed using the L1 Loss since allows us to achieve better PSNR results and to make the training more stable.

7.4 Experimental Results

7.4.1 Restoration with known compression Quality Factor

We compared the proposed model with the state-of-the-art models ARCNN[54], CAS-CNN[41], D3[182], and the more recent DMCNN[197], MemNet[162], MWCNN[115], ARGAN[70], S-Net[198] and JBCBR [44].

Since the state-of-the-art methods operate only on the Y channel of the images, in order to make a fair comparison, we used only the result coming from the application of the LumaNet, without any integration of data from the color components. The metrics are evaluated on the Y channel recovered by the first network with the corresponding target images, using the MATLAB standard libraries, over five different compression qualities: $10, 20, 40, 60, 80$. For each method, on all the datasets considered, we report the results taken

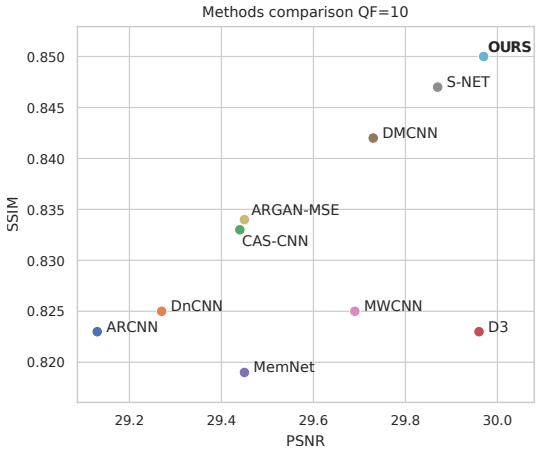


Figure 7.5: PSNR-SSIM comparison of the state-of-the-art-models and the proposed method. For both metrics higher value means better visual results.

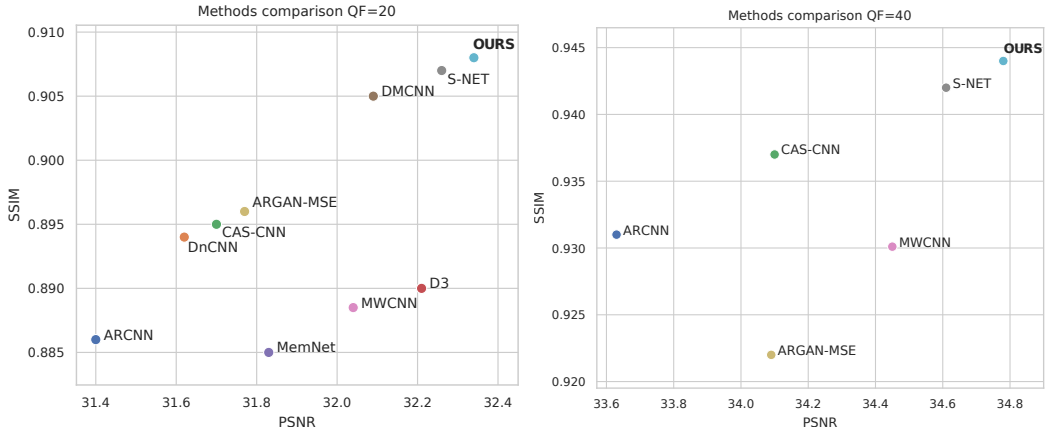


Figure 7.6: PSNR-SSIM comparison of the state-of-the-art-models and the proposed method. For both metrics higher value means better visual results.

from the corresponding publication, except for ARCNN and MWCNN which provide the source-code, that are then used for the evaluation. Since the training of the proposed methods leads to a single model that can be used for all the quality factors, we used the same model for the evaluation at all the qualities previously mentioned. All the state-of-the-art methods compared, instead, have a different trained model for each QF considered.

Table 7.2, 7.3, 7.4 and 7.5 respectively report the comparison on the LIVE1, BSD500, Classic-5, and the Kodak Lossless True Color Im-

Table 7.2: Comparison on test set LIVE1: for the methods in the state of the art a different model is trained for each QF considered. The proposed method uses the same model for all the QFs.

	Quality	ARCNN [54]	DnCNN [196]	CAS-CNN [41]	D3 [182]	DMCNN [197]	MemNet [162]	MWCNN [115]	S-NET [198]	ARGAN-MSE [70]	ARGAN [70]	CED-GT [191]	Proposed model
PSNR	10	29.13	29.19	29.44	29.96	29.73	29.45	29.37	29.87	29.47	27.65	26.54	**29.98**
	20	31.40	31.59	31.70	32.21	32.09	31.83	31.58	32.26	31.81	29.99	29.33	**32.34**
	40	33.63	33.96	34.10	-	-	-	34.17	34.61	34.17	31.64	-	**34.78**
	60	-	-	35.78	-	-	-	-	-	-	-	-	**36.47**
	80	-	-	38.55	-	-	-	-	-	-	-	-	**39.31**
PSNR-B	10	28.74	-	29.19	29.45	29.55	-	28.85	-	29.13	27.63	26.51	**29.61**
	20	30.69	-	30.88	31.35	31.32	-	30.83	-	31.29	29.69	29.32	**31.76**
	40	33.12	-	33.68	-	-	-	33.33	-	33.42	31.17	-	**33.96**
	60	-	-	35.10	-	-	-	-	-	-	-	-	**35.51**
	80	-	-	37.73	-	-	-	-	-	-	-	-	**38.26**
SSIM	10	0.823	0.812	0.833	0.823	0.842	0.819	0.832	0.847	0.833	0.777	0.767	**0.851**
	20	0.886	0.880	0.895	0.890	0.905	0.885	0.891	0.907	0.897	0.864	0.854	**0.908**
	40	0.931	0.924	0.937	-	-	-	0.936	0.942	0.937	-	0.903	**0.944**
	60	-	-	0.954	-	-	-	-	-	-	-	-	**0.960**
	80	-	-	0.973	-	-	-	-	-	-	-	-	**0.976**

Table 7.3: Comparison on test set BSD500: for the methods in the state of the art a different model is trained for each QF considered. The proposed method uses the same model for all the QFs.

	Quality	ARCNN [54]	DnCNN [196]	CAS-CNN [41]	D3 [182]	DMCNN [197]	MemNet [162]	MWCNN [115]	S-NET [198]	ARGAN-MSE [70]	ARGAN [70]	CED-GT [191]	Proposed model
PSNR	10	29.10	-	-	-	29.67	-	29.50	29.82	29.05	27.31	26.00	**29.92**
	20	31.25	-	-	-	31.98	-	31.34	32.15	31.23	28.48	28.62	**32.23**
	40	33.55	-	-	-	-	-	33.23	34.45	33.45	30.98	-	**34.61**
PSNR-B	10	28.75	-	-	-	-	-	28.60	-	28.64	27.31	25.97	**29.41**
	20	30.60	-	-	-	-	-	29.84	-	30.49	29.03	28.58	**31.39**
	40	32.80	-	-	-	-	-	31.04	-	32.34	30.16	-	**33.34**
SSIM	10	0.819	-	-	-	0.840	-	0.835	0.844	0.806	0.749	0.731	**0.847**
	20	0.885	-	-	-	0.904	-	0.889	0.905	0.877	0.841	0.825	**0.906**
	40	0.929	-	-	-	-	-	0.928	0.941	0.923	0.884	-	**0.943**

AGE SUITE datasets for all the three metrics considered. As can be seen, the proposed solution outperforms the state of the art on all the metrics. With the proposed model we obtained improvements with respect to the state-of-the-art methods on both general perceptual quality (PSNR/PSNR-B) and structure reconstruction (SSIM) on the first two datasets. On the third and fourth ones, we obtain improvement in both PSNR-B and SSIM, with comparable results with respect to the best method in terms of PSNR.

Since each index focuses on different aspects of the restoration quality, each index alone is not capable to summarize all the aspects of a good reconstruction. Therefore, we also compare the methods in a graph style-view, reported in Figures 7.5 and 7.6 to correlate the two indexes. In order to obtain a more pleasing perceived quality, both the metrics must obtain high values. It is easy from this kind of view to see how the proposed method outperforms the current state-of-the-art models even if a single model is used for all the QFs.

Table 7.4: Comparison on test set Classic-5: for the methods in the state of the art a different model is trained for each QF considered. The proposed method uses the same model for all the QFs.

	Quality	ARCNN [54]	DnCNN [196]	CAS-CNN [41]	D3 [182]	DMCNN [197]	MemNet [162]	MWCNN [115]	S-NET [198]	ARGAN-MSE [70]	ARGAN [70]	CED-GT [191]	Proposed model
PSNR	10	29.04	29.4	-	-	-	29.69	**29.68**	-	-	-	-	29.67
	20	31.16	31.63	-	-	-	31.90	31.78	-	-	-	-	**31.89**
	40	33.34	33.77	-	-	-	-	**34.05**	-	-	-	-	34.04
PSNR-B	10	28.75	-	-	-	-	-	29.06	-	-	-	-	**29.35**
	20	30.6	-	-	-	-	-	30.95	-	-	-	-	**31.43**
	40	32.8	-	-	-	-	-	33.20	-	-	-	-	**33.33**
SSIM	10	0.811	0.803	-	-	-	0.811	0.828	-	-	-	-	**0.829**
	20	0.869	0.861	-	-	-	0.866	0.878	-	-	-	-	**0.882**
	40	0.91	0.9	-	-	-	-	0.916	-	-	-	-	**0.917**

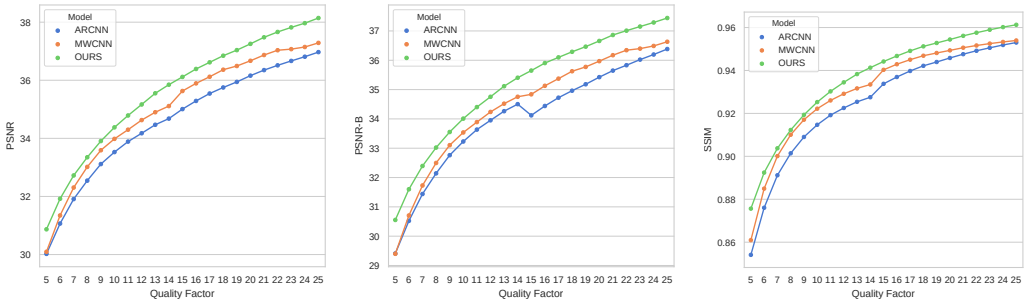
Table 7.5: Comparison on test set Kodak Lossless True Color Image Suite: for the methods in the state of the art a different model is trained for each QF considered. The proposed method uses the same model for all the QFs. The values marked with the symbol (*) are taken from [44] while the other ones are obtained using the codes officially released by the corresponding authors, and the evaluation code from [54].

	Quality	D2SD [116]	ARCNN [54]	DnCNN [196]	MemNet [162]	MWCNN [115]	JBCBR [44]	Proposed model
				Y				
PSNR	10	30.28*	30.01 / 30.56*	30.75*	30.96*	30.82 / **31.19***	31.03	31.10
	20	32.45*	32.31 / 32.78*	33.09*	33.29*	33.10 / **33.47***	33.31	33.44
PSNR-B	10	-	29.88	-	-	30.63	30.82	**30.93**
	20	-	32.06	-	-	32.81	32.99	**33.17**
SSIM	10	-	0.818	-	-	0.836	0.846	**0.847**
	20	-	0.881	-	-	0.896	0.902	**0.904**

7.4.2 Restoration with unknown compression Quality Factor

Another kind of evaluation has been done about the capability of the models to recover images at compression quality factors never seen during training. In most of the real use-cases, the JPEG compression quality factor previously applied to an image is not known: it is then important that a model can recover the images without this prior information. On the other hand, if we are able at least to estimate the compression quality factor of the input compressed image, following the previous approaches we should train new models for each specific quality factor needed, or use the model trained for the closest QF to the desired one.

We compare the proposed model with the two state-of-the-art models for which the code is available (i.e. ARCNN and MWCNN) in a specific selection



PSNR PSNR-B SSIM

Figure 7.7: Comparison on QFs not seen during training. For ARCNN and MWCNN the models trained for QF=10 and QF=20 are tested on QF in the range [5, 25]. The proposed model is trained for QF in the range [10, 100] with steps of 10, and is tested on the same intermediate QFs not seen in training.

of cases. Since previous models have been trained on specific quality factors, and the proposed one has been instead trained over quality factor from 10 to 100 in steps of 10, without the use of images with QFs in between, we decided to test the model robustness on "never seen" artifacts. In order to perform the evaluation coherently, for the state-of-the-art algorithm we used the pretrained models for the nearest quality factor, for example, if the input image has been compressed with QF = 17 we used the models trained for QF = 20. The evaluation has been done only on the Luminance channel restores only with the LumaNet, in the same way, that has been done for the known QFs. For this evaluation we adopted the SDIVL dataset: for each image of the testset, we applied all of the compression factors in the interval $5-25$. The evaluation is done in the same way it has been done in the previous section, by extracting Y channel and measuring PSNR, PSNR-B, and SSIM indexes.

In Figure 7.7 are shown the results of the models on the SDIVL with all the quality factors compression. As can be seen in those graphs the poposed model shows a more stable behavior: the model is capable to restore images at different QFs with more coherent and smooth behavior in relation to the increase of the QF, in comparison with the other methods. Moreover, the previous state-of-the-art models have difficulties to restore images at quality factors distant from the trained one. It is particularly interesting to see how the other models have difficulties to restore images at higher qualities with respect to the QF used in training, in terms of structures in the images (Figure 7.7), due to the more complex textures never seen by the models during the training

Table 7.6: Comparison on test set LIVE1 by subdividing the image patches on the basis of the frequency content in five classes from high to low.

	ARCNN [54]			MWCNN [115]			Proposed model		
Frequency	PSNR	PSNR-B	SSIM	PSNR	PSNR-B	SSIM	PSNR	PSNR-B	SSIM
high	27.53	27.26	0.782	27.61	27.24	0.792	**28.18**	**27.88**	**0.807**
medium-high	25.00	24.66	0.685	25.24	24.67	0.700	**25.64**	**25.18**	**0.729**
medium	24.61	24.27	0.734	24.50	23.82	0.740	**25.37**	**24.91**	**0.773**
medium-low	25.92	25.49	0.794	25.91	25.24	0.803	**26.73**	**26.21**	**0.827**
low	27.08	25.93	0.840	26.72	25.27	0.849	**27.81**	**26.52**	**0.864**

Table 7.7: Comparison on test set LIVE1 by subdividing the image patches on the basis of the detail density in five classes from low to high.

	ARCNN [54]			MWCNN [115]			Proposed model		
Edges frequency	PSNR	PSNR-B	SSIM	PSNR	PSNR-B	SSIM	PSNR	PSNR-B	SSIM
high	23.20	22.94	0.667	23.42	22.82	0.683	**23.87**	**23.45**	**0.716**
medium-high	24.69	24.39	0.721	24.91	24.31	0.735	**25.42**	**25.02**	**0.763**
medium	25.68	25.22	0.758	25.94	25.26	0.772	**26.41**	**25.86**	**0.794**
medium-low	26.83	26.12	0.805	27.01	25.95	0.817	**27.47**	**26.61**	**0.832**
low	29.17	28.22	0.884	27.45	26.28	0.888	**29.97**	**28.99**	**0.897**

Table 7.8: Color restoration comparison on test set Kodak Lossless True Color Image Suite. Evaluation of Cb and Cr channels restoration in terms of PSNR.

Quality	JPEG	SA-DCT [61]	JBF [172]	EJBF [172]	JBCBR [44]	Proposed model
			Cb			
10	36.14	37.83	37.38	37.39	39.16	**39.30**
20	39.02	40.69	40.34	40.45	41.99	**42.22**
			Cr			
10	36.00	37.57	37.16	37.20	38.92	**39.04**
20	38.99	40.47	40.01	40.24	41.64	**41.89**

phase.

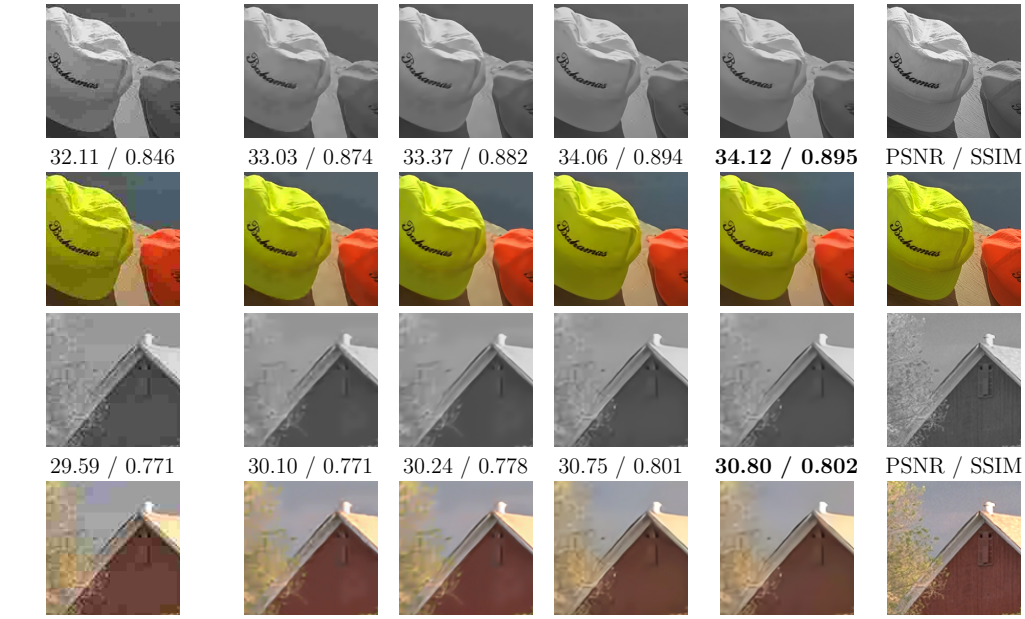


Figure 7.8: Visual comparison of image restoration result. The first and third lines show the Luma channel (Y) restored by the models with the associated PSNR and SSIM values, computed on the whole image; the second and forth lines show the RGB colored version. For the models that can only recover the Y channel (identified by the * symbol), the Cb and Cr channels are taken directly from the original high quality corresponding ground truths crops.

7.4.3 High and low frequency areas restoration

In order to better understand if the proposed method performs better than approaches in the state of the art only on certain image types, a further experiment have been conducted: the images from LIVE1 testset, compressed at $QF = 10$, have been divided into 64×64 patches and have been classified into five categories. The categories are obtained by equally diving the patches into five bins with respect to both frequency and detail density. Patch frequency is computed as the weighted average of the 2D Fourier Transform normalized magnitude. Patch detail density is computed as the 2D average of the result of the Canny edge detection. The results for the considered evaluation metrics over the five categories of the frequency and detail density are respectively reported in Table 7.6 and 7.7. From the results reported it is possible to notice that the proposed method consistently outperforms the state of the art on all

the frequency and detail density categories.

7.4.4 Color Restoration

The final evaluation is focused on the color restoration capability of the models. The comparison has been done evaluating the Cb and Cr channels of the recovered images, in the same way that has been done by Chen *et al.*[44]: the chroma component of the restored images from the Kodak Lossless True Color Image Suite, with QF = 10 and QF = 20, have been used in the comparison. From the results reported in Table 7.8 in terms of PSNR it is possible to see that the proposed model obtains better results than the other methods. This is remarkable since, analogously to what has been done for the previous experiments, the compared methods trained a different method for each QF considered, while the proposed method uses a single model for all QFs. A visual comparison of color image restoration results is reported in Figure 7.8. In particular, it is possible to see how the methods that are trained to recover full color images obtain much better visual results even on the luma channel alone. In order to not perform an unfair visual comparison, for the methods designed to recover just the luma channel, the Cb and Cr channels are taken from the original uncompressed image.

7.4.5 Model complexity

In this section is presented a comparison of model complexity. In particular, have been compared the inference time for a single $512 \times 512 \times 3$ on an NVIDIA Titan V GPU, the PSNR score and the model size in terms of learnable parameters. The results are reported in Figure 7.9. They are divided into two plots: the bottom one reports the comparison for the restoration of the Y-channel, the top one reports the comparison for the restoration of the Cb and Cr channels. For some methods, two sizes are reported: the full circle represents the size for a single model (i.e. trained for a single QF) while the empty circle represents the size of ten models, simulating the fact that at test time more models are needed to cover all the possible QFs. The plots show that the proposed solution compares favorably with respect to the state of the art on all the aspects considered, showing also a very good trade-off between PSNR and model size.

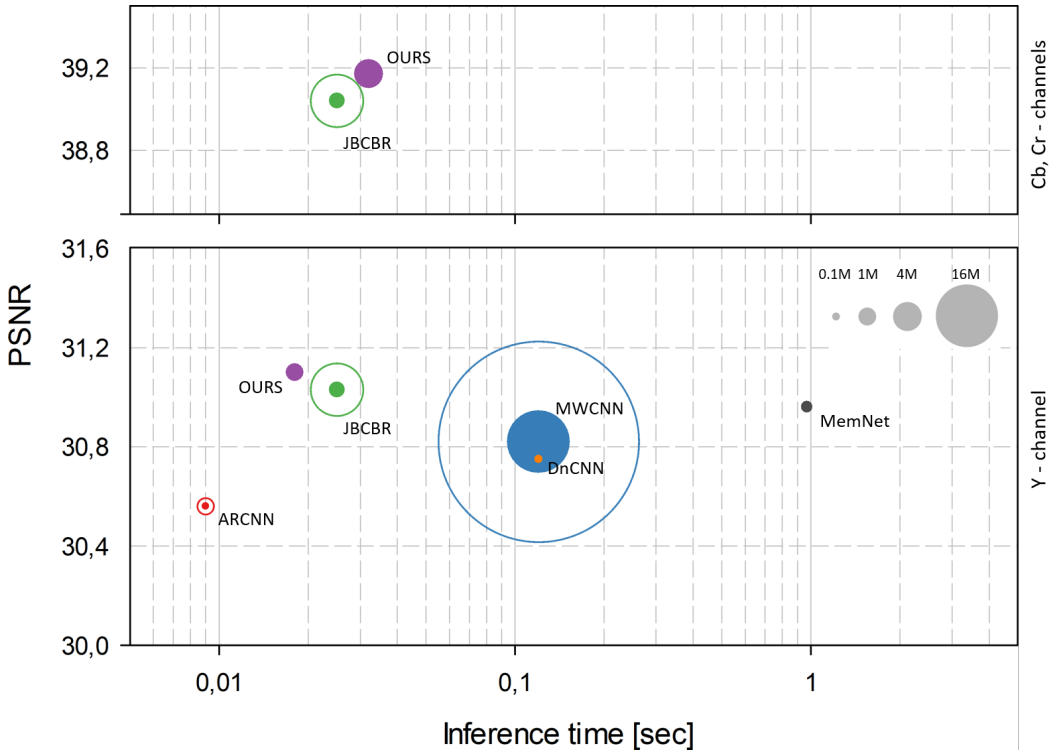


Figure 7.9: Inference time for a $512 \times 512 \times 3$ image on a NVIDIA Titan V GPU. In the top plot the average PSNR on the Cb and Cr channels is reported, in the bottom plot the PSNR on the Y channel is reported.

7.5 Summary

In this section a deep residual autoencoder exploiting Residual-in-Residual Dense Blocks (RRDB) to remove artifacts in JPEG compressed images have been presented. The proposed method is blind and universal, i.e. it is independent from the QF used. The proposed model operates in the YCbCr color space and performs a two-phase restoration of JPEG artifacts: in the former phase, a first autoencoder exploiting 2D convolutions is used to restore the luma channel of the input image. In the latter phase, the restored luma is stacked along the channel dimension with the chroma channels of the input image; then, a second autoencoder employing 3D convolutions uses the restored luma channel as a guide to restore the chroma channels.

The main contributions of this work are: i) the design of a blind universal method for the restoration of JPEG compression artifact that is independent from the QF used; ii) the design of a model trainable end-to-end that fully exploits knowledge about JPEG compression pipeline; iii) a thorough comparison with the state of the art on four standard datasets at fixed QFs; iv) an analysis of robustness of restoration results at QFs not used for training.

Extensive experimental results on four widely used benchmark datasets (i.e. LIVE1, BDS500, CLASSIC-5, and Kodak) show that the proposed model is able to outperform the state of the art with respect to all the evaluation metrics considered (i.e. PSNR, PSNR-B, and SSIM). This result is remarkable since the approaches in the state of the art use a different set of weights for each compression quality, while the proposed model uses the same weights for all of them, making it applicable to images in the wild where the QF used for compression is unknown. Furthermore, the proposed model shows greater robustness than state-of-the-art methods when applied to compression qualities not seen during training. Since preliminary experiments with the same architecture proposed showed good results for the restoration of other artifacts (i.e. noise removal, in the CVPRW NTIRE2019 challenge [2]), as future work is to investigate its extension to other single and multiple distortions [49]. To this end, techniques that are able to better interpret and understand what the model has learned, such as what has been done in the framework of image classification [128, 19], should be studied to be applied also in the image processing domain.

7.6 Model adaptation to other artifacts: sRGB noise

One last step related to the model proposed for JPEG artifact reduction is the one of trying to adapt it to other kinds of artifacts. Here in particular the proposed solution has been adapted for image denoising. The main purpose is to prove the potential use of such a model, with very limited changes, on another image restoration task.

The proposed method works on YCbCr noisy images and gives as output restored YCbCr images, so to adapt it to the denoising task, the pre-processing step converts the input RGB images into YCbCr color space, and the post-processing step converts the result back into the RGB color space. The entire solution is made of two autoencoder neural networks: the first one is used for the restoration of the luma channel (Y channel), while the second one restores the chroma components (Cb and Cr channels) of the images, using the restored luma channel as a "structure map" to guide the reconstruction. Differently from the original method that was designed for JPEG restoration, for the denoising task for both the first and second network the number of Residual-in-Residual Dense Blocks (RRDBs) has been set equal to $B = 5$, the input mini-batches size equal to 8, made of 100×100 pixel crops taken from the training dataset. To increase the number of structures and textures seen by the network, online data augmentation (random flipping and rotation) has been applied to the input training crops, during the training phase. The overview of the model is shown in Figure 7.10.

7.6.1 Training description

The dataset used for the training of this model is the SIDD-Medium dataset [1] that consists of 320 noisy images in both raw-RGB and sRGB space with corresponding ground truth and metadata. Each noisy or ground truth image is a 2D array of normalized raw-RGB values (mosaiced color filter array) in the range [0, 1] in single-precision floating point format. The metadata files contained dictionaries of Tiff tags for the raw-RGB images. The Tiff data has not been used in any way since the main purpose is to prove the potential use of the previously defined model for JPEG restoration in this new scenario. The validation data consisted of 1280 noisy image blocks (i.e., croppings) from both raw-RGB and sRGB images, each block is 256×256 pixels. The testing

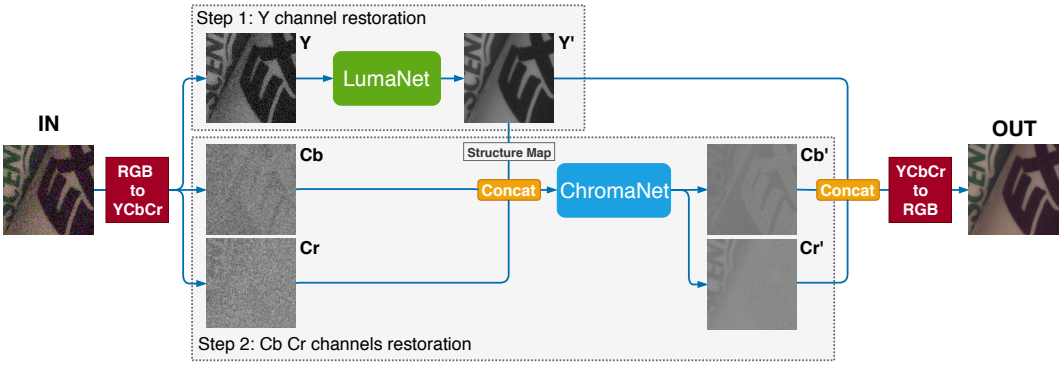


Figure 7.10: Overview of the denoise model. The approach is an adaptation of the JPEG artifact reduction proposed method to gaussian noise artifact removal.

data consisted of 1280 noisy image blocks different from the validation block but following the same format as the validation data.

To train this model, mini-batch size has been set equal to 8; each minibatch is made of 100×100 pixel crops taken from the training dataset augmented with online data augmentation random flipping and rotation. The model has been trained with Adam optimizer, with a starting learning rate of 1×10^{-4}, decreased by a factor 10 after 240k and 300K iterations. The model has been developed in Pytorch v1.0.0 on a Nvidia Titan V with 12 GB dedicated RAM.

Snapshot ensemble technique has been used for the final testing phase, combining the results of the best epoch of training of the proposed model, combined with the results of two other epochs near to the best one in terms of quality indexes. The results obtained by the best model have been averaged with the results coming from two other checkpoints of the same model coming from previous epochs with very close performances in terms of PSNR and SSIM over the validation set. The ensemble operation has given a very small boost on the performances in terms of PSNR index (0.005dB).

7.6.2 Experimental Results

To test the model adaptation performances, the proposed approach has been used to participate in a Workshop challenge at CVPR 2019. The model has been tested on sRGB images.

The proposed model competed with 220 different approaches and ended in position 15 of the leader board. In table 7.9 are reported the respective PSNR

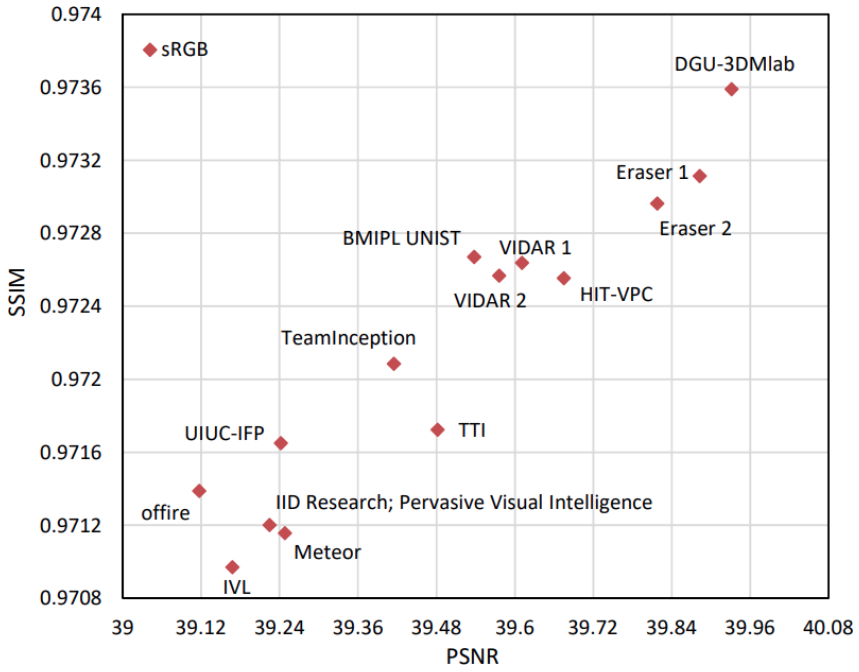


Figure 7.11: Results of the NTIRE 2019 challenge. The plot represents the final top 15 containing the best algorithms for single image noise removal of sRGB images. The proposed model is IVL labeled one.

and SSIM values achieved by the proposed approach, while in Figure 7.11 is reported a scatter plot showing the performances of the different approaches in terms of PSNR and SSIM.

As can be seen from table 7.9, the proposed approach for JPEG artifact reduction, once adapted for the new denoising task, achieves very good results with respect to the best model in the ranking. In terms of SSIM index, the difference is only of a total amount of 0.0026, while for the PSNR is of $0,764$ db. This behavior, even if the performances do not reach the results achieved by the best method in the ranking, shows how is possible to adopt the proposed per channel approach to similar kinds of artifacts, obtaining competing results with respect to ad hoc designed approaches. Another interesting achievement of the proposed approach is the result in terms of running time (expressed in seconds per megapixel), which is significantly below the execution time of the other proposed methods, with exception of "IID Research; Pervasive Visual Intelligence".

Table 7.9: Results and rankings of methods submitted to the sRGB denoising track of NTIRE 2019 workshop.

Team / Method	PSNR	SSIM	Runtime (s/Mpixel)
DGU-3DMlab	39.932(1)	0.9736(1)	0.5577
Eraser	39.883(2)	0.9731(2)	-∼2
Eraser	39.818(3)	0.973(3)	3.416
HIT-VPC	39.675(4)	0.9726(7)	-
VIDAR	39.611(5)	0.9726(5)	0.903
VIDAR	39.576(6)	0.9726(6)	0.903
BMIPL UNIST	39.538(7)	0.9727(4)	3.132
TTI	39.482(8)	0.9717(9)	~ 2
TeamInception	39.415(9)	0.9721(8)	1.136
Meteor	39.248(10)	0.9712(13)	0.13
UIUC-IFP	39.242(11)	0.9717(10)	10.73
IID Research; Pervasive Visual Intelligence	39.225(12)	0.9712(12)	0.0283
IVL	39.168(13)	0.971(14)	0.02
offire	39.117(14)	0.9714(11)	3.83

Part III

Addressing external artifacts

The second type of artifacts and defects which can affect image quality is the one corresponding to the elements external to the camera pipeline. As seen in Section 4.2, image quality is not only dependent on the operations that are done in the processing pipeline, but also on external factors such as the presence of elements in the acquired scene or dependent to the light and the lenses used by the capturing system.

In this part of the thesis, the artifacts treated are the ones caused by weather conditions, such as rain and haze, which can affect the overall quality and usability of an image. In particular, the work is focused on the reduction of artifacts coming from taking a picture of a scene during a rainy weather. In this scenario, images are usually affected by the presence of specific kinds of artifacts, which occludes information making images less beautiful and usable. Moreover, to analyze also the usefulness aspect, a study of the impact of artifacts and enhancement operation over the images has been done. Two studies are presented: one related to the semantic segmentation task and another one on optical character recognition. These two studies are described in Sections 8.3 and 8.4, after the definition and proposal of an autoencoder convolutional neural network for rain and rain-induced haze removal. The last chapter is instead focused on the problem of raindrop removal, which is related to the reduction of defects related to the presence of external elements which lies over the lenses or a transparent surface in front of the camera, and which cannot be controlled in any way by the user at shooting time and must be addressed in post-processing steps.

Regarding the relation with the digital processing pipeline, since those processing operations are not meant to be integrated directly in the digital processing pipeline, computational complexity and hardware-related constraints are not anymore strict as for the methods in part II. This fact permits the design of methods that rely on deep learning and that may need more computational power to obtain the desired result.

Chapter 8

Rain streak reduction & downstream tasks

In the last years, low-level image processing has improved a lot with the introduction of Convolutional Neural Networks, permitting to outperform classical handcrafted methods in most tasks such as Super-Resolution, Image Denoising, Image Colorization, Image Dehazing and Deraining. Those methods are intended to enhance the input images that suffer from problems of different nature in order to improve the quality and visibility as perceived by humans or for subsequent automatic systems like automatic object detectors, etc. In this section, the focus of the restoration is Image Deraining, where the objective is to remove rain from images taken during bad weather conditions, more specifically in situations where the visibility is occluded by rain streaks and haze. Note that the case in which there are raindrops over the camera lenses has not been considered in this specific work but will be treated separately, specifically is the focus of the work in section 9.

8.1 Related works

In the last years, a lot of CNN based models for single image deraining have been presented. One of the first attempt in rain removal from single image was presented by Eigen et al. [57]: this first approach was focused on the removal of dirt and rain from a glass surface between the camera and the scene content. Fu et al. [64], in 2017, designed a Convolutional neural network for rainstreaks removal from single images: to train that model they adopted a synthetic rainy dataset, spefically generated for the task. Again,

Fu et al. [65] proposed another approach, based on the concept of residual learning, which consent a lighter training procedure of models for rainy image restoration. Yang et al. [188] presented a CNN model for rain detection and removal from single images: this approach is based on a step of detection of rainstreaks present in the images and then a removal process, based on the result of the previous detection step. Quian et al. [136] presented the first generative network, trained using adversarial training, for raindrop removal, based on the concept of attention maps. Zhang et al. [194] proposed instead a CNN model based on the estimation of rain density in the image in order to drive the restoration process in relation to the amount of distortion presents in the images. Again, Zhang et al. [195] proposed one year later another approach for rainstreak removal, based on Conditional Generative Adversarial Network, trained by using newly synthetic rainy dataset. Finally, Li et al. [111] presented a benchmark of all the current state of the art models for the deraining task, considering the different existing datasets and also the possibility to improve detectors' performances with different methods.

Inspired by this last work, the work presented in this section aims to see the effect of this kind of processing on rainy street view images in order to improve the performance of possible subsequent downstream task on the processed images.

8.2 Proposed method for rainstreaks removal

Inspired by the results obtained by Convolutional Neural Networks and in particular GANs in low-level image processing tasks such as Super Resolution, Image Colorization, Image Inpainting, Noise Removal, etc... the decision has been the one of using a U-Net style architecture trained using a discriminative network in a conditional Generative Adversarial Network framework.

8.2.1 Network Details

The structure of the DeRaining CNN is based on the U-Net [143] architecture, with the addition of skip connections as done for Pix2Pix network [88]. The architecture is shown in Figure 8.2a. Based also on recent works related to image enhancement, some changes have been made to the classical U-Net architecture, in order to reduce the introduction of artifacts and improve the quality of the final results:

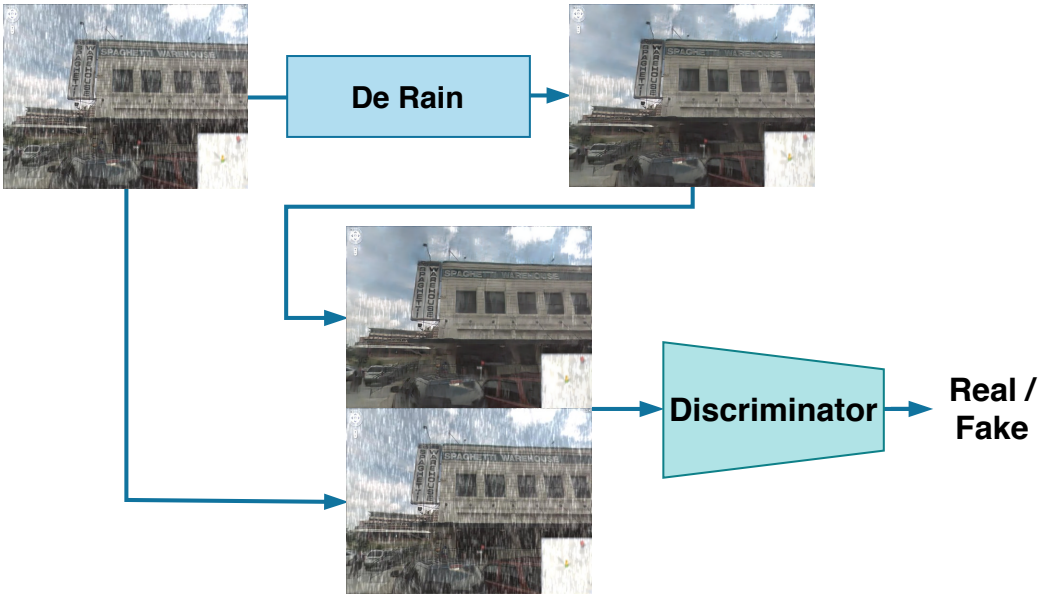


Figure 8.1: Training system with Conditional patchGAN.

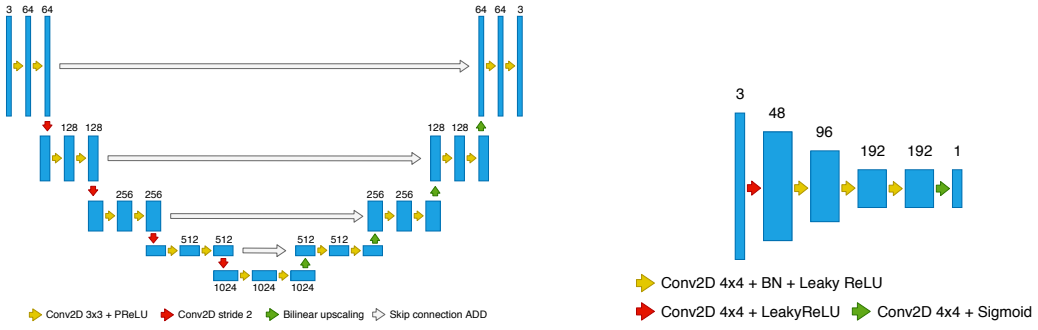
- The normalization layers have been removed from the model, in order to avoid the generation of artifacts, as done in [112] and [129].

- Max-pooling operation have been replaced with convolutions with 2-pixel stride to reduce feature spatial dimensions, without losing useful information for the restoration process.

- A combination of bilinear upsampling with 2D Convolution has been adopted, to reduce artifacts coming from the application of the deconvolutional layers in the decoder part of the network.

A patchGAN discriminative network have been used in order to train the model in a GAN framework, trained with a Conditional GAN training approach [125], using both generated and input images as input to the discriminator to better classify fake and real images, similarly to the discriminator used for Pix2Pix.

The architecture is shown in Figure 8.2b while a scheme of the entire training method is shown in Figure 8.1.



(a) (b)

Figure 8.2: (a) U-Net style architecture of the generative network. The max pooling layers have been replaced with convolutions with strides > 1 and the upscaling operation is performed with Bilinear Interpolation combined with convolutions. (b) PatchGAN style discriminative network architecture.

8.2.2 Loss Function

The loss function used to train the model is defined as:

$$Loss = \lambda_e * L_e + \lambda_{adv} * L_{adv} + \lambda_p * L_p. \tag{8.1}$$

which is the combination of three loss functions, weighted by three different weight values $\lambda_e, \lambda_{adv}, \lambda_p$

Given an image pair $\{x, y\}$ with C channels, width W and height H (i.e. $C \times W \times H$), where x is the input image and y is the corresponding target, the three loss functions are defined as follows.

The per-pixel Euclidean loss, defined as:

$$L_e = \frac{1}{CWH} \sum_{c=1}^{C} \sum_{w=1}^{W} \sum_{h=1}^{H} ||\phi_E(x^{c,w,h}) - y^{c,w,h}||_2^2, \tag{8.2}$$

where $\phi_E(\cdot)$ is the learned network for rain removal.

The Perceptual loss [94] defined as distance function between features extracted from the target and output images, using the pre-trained VGG network:

$$L_p = \frac{1}{C_i W_i H_i} \sum_{c=1}^{C_i} \sum_{w=1}^{W_i} \sum_{h=1}^{H_i} ||\phi_E(x^{c,w,h}) - V(y_B^{c,w,h})||_2^2, \tag{8.3}$$

where $V(\cdot)$ represents a non-linear CNN transformation (VGG16 network). Finally, the original *GAN* loss described as:

$$L_{adv} = \mathbb{E}_{x,y}[\log D(x, y)] + \mathbb{E}_x[\log(1 - D(x, G(x)))], \tag{8.4}$$

where $G(\cdot)$ is the trained generative network for image de-raining.

8.2.3 Training data

In order to train the model for the rain removal task, a dataset made of *input-target* images created with synthetic rain masks has been adopted. The dataset has been presented by Zhang et al. [195]: the entire dataset is made of 700 images, where 500 images have been randomly taken from the UCID dataset [147] and 200 images are randomly chosen from the BSD-500 training set [10]. The validation has been performed with the images from the test set made with 100 images, 50 from the UCID dataset and 50 from the BSD-500 dataset [10].

For each image, a rain mask has been chosen (from a set of 10 different ones) and applied. This operation has been manually done by Zhang et al. [195] using Photoshop. Moreover, in order to test the models with "real" rainy images, Zhang et al. collected a set of 50 natural images.

Since for the training phase the number of images is limited, both image flipping and rotation have been used in order to augment the dataset. All of the images have been cropped to a common size of 256×256, in the case in which the images were bigger, and upscaled to that dimension, in the case in which the images were smaller.

8.2.4 Training details

The model has been written in PyTorch v1.3.1 and trained on an Nvidia Titan V GPU. The training has been done with batch size 8 for a total amount of 1K epochs. The model has been trained using Adam optimizer [104] with a starting learning rate of 10^{-5} for both generative and discriminative networks. For the balancing of the loss, in order to stabilize the training, λ_e, λ_p and λ_{adv} have been respectively set to values 1, 0.1 and $6.6 * 10^{-3}$.

8.2.5 Synthetic Rain Augmentation

To reproduce semi-realistic rainy images, to be used for testing the model in different downstream tasks, a procedure for the generation of random rainy masks to apply over the target images has been defined. Differently from [195], instead of using Photoshop to generate a limited number of masks to randomly apply to the images, a MATLAB procedure to create a random rainy mask generator has been designed: for each image generates a new mask, based on some parameters randomly selected in ranges that have been defined empirically, with respect to the original approach from [195] and the objective of obtaining semi-realistic rainy images. The pipeline is represented in Figure 8.3.

Starting from an sRGB image, the process first generates a raindrop mask, by choosing four parameters: d_1 rain density, σ_1 Gaussian filter dimension, l_1 streak length, α_1 falling angle. After that, a rain streaks map is generated using two parameters previously chosen for the first mask: l_1 streak length and σ_1 falling angle, and two other ones chosen at this step: d_2 rain density and α_2 Gaussian filter dimension. Eventually, an optional haze mask is generated. These three masks are then applied to the image in order to obtain the rainy version of the original input image.

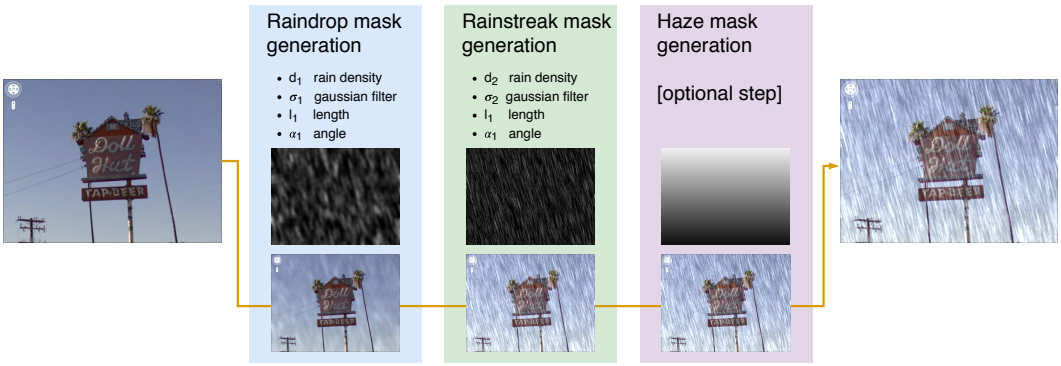


Figure 8.3: Steps of the pipeline designed for the synthetic rain generation

8.3 Semantic Segmentation

The automotive field has seen a strong expansion in recent years, where a crucial role is played by perception systems for autonomous vehicles and for assisted driving. The development of computer vision techniques in this field

potentially allows for a decrease of the production costs due to the exploitation of inexpensive hardware, i.e. RGB cameras in place of depth sensors. Large benchmark datasets such as the CityScapes dataset have proven to be extremely valuable in developing and testing automotive-related solutions, such as networks for monocular depth estimation [30] and for semantic segmentation [122] of street scenes.

The specialized literature is mostly focused on either improving the model accuracy, or in reducing the computational complexity of the involved models. Relatively little effort has been put into investigating and quantifying the impact of meteorological conditions over the method performance, including phenomena that alter the image quality such as haze, rain, and changes in illumination conditions. This is mostly due to the lack of appropriate datasets, i.e. real-life photos acquired in bad weather conditions, and annotated for computer vision tasks such as semantic segmentation. For this reason, in fact, the analysis will resort to synthetic rain augmentation over annotated datasets for the quantitative experiments presented in this section.

Valada et al. [168] presented a multi-stream deep neural network that learns features from multimodal data, and adaptively weights different features based on the scene conditions. The authors assessed semantic segmentation on the synthetic dataset Synthia [144] (which includes rainy scenes), but did not offer a direct comparative evaluation on the presence and absence of rain-induced artifacts. Khan et al. [101] created an entirely artificial dataset for semantic segmentation in different atmospheric conditions, to be used as training data for the task. In this section will be argued that although synthetic data generation is essential in producing an adequately large database, introducing synthetic rain artifacts over real images would instead offer the grounds for an evaluation that is closer to a real-case scenario. Porav et al. [134] focused on the removal of rain droplets, which by definition refer to the artifacts introduced by a wet glass on a clear day. As such these are only partially representative of real-case scenarios. Halder et al. [79] developed a physics-based data augmentation technique, used to train more robust models for semantic segmentation, although they offer no insights on the benefits of rain-removal techniques. Recently, Li et al. [111] defined a unified benchmark for images perturbed by rain streaks, rain drops, and mist, and tested different methods for rain removal, including among the evaluation criteria the impact over vehicle detection.

8.3.1 Dataset and evaluation metrics

Experiments have been performed on the Cityscapes [51] dataset: a set of urban street images annotated with pixel-wise semantic information. It is composed of 5000 high-resolution images (2048×1024) out of which 2975, 500 and 1525 images belong respectively to train, validation and test subsets. Annotations include 30 different classes of objects, although only 19 are typically used for training and evaluation, plus a background class:

- road
- sidewalk
- building
- wall
- fence
- pole
- traffic light
- traffic sign
- vegetation
- terrain
- sky
- person
- rider
- car
- truck
- bus
- train
- motorcycle
- bicycle
- (background)

The dataset is characterized by a vast diversity of scenes, with images taken from different cities all with good or medium weather conditions.

Two metrics are used for model validation: avereage of class-wise Intersection over Union (IoU, also called Jaccard Index) and average class-wise Accuracy. These are computed as:

$$\text{IoU} = \frac{TP}{TP + FP + FN} \tag{8.5}$$

$$\text{Accuracy} = \frac{TP}{TP + FN} \tag{8.6}$$

Where TP, FP and FN are, respectively, the number of True Positive, False Positive, and False Negative pixels.

8.3.2 Experimental Results

In this section are analyzed the performance of the semantic segmentation model in relation to the condition of the data involved (i.e. "clean", rainy, and rain-removed). The evaluation of the model has been done considering the condition of data used for training as well as for validation.

Three version of the Cityscapes dataset have been considered for the analysis:

- **Clean images**: the original images from the Cityscapes dataset.
- **Rainy images**: images obtained using the synthetic mask generation algorithm starting from the "clean" images.

- **Rain-removed images**: images obtained by removing the rain from the rainy images version of the Cityscapes, using the rain reduction algorithm.

This analysis has been done with the purpose of studying how the level of degradation in data can affect the performances of the segmentation algorithm in inference time, and how it can affect the learning process of the model.

Table 8.1 and 8.2 report respectively the mean Accuracy and mean of class-wise Intersection over Union.

Table 8.1: Accuracy of semantic segmentation on the Cityscapes validation dataset: table shows the results in relation to the training data used for the semantic segmentation network.

Accuracy		Test data		
		Clean	Rainy	Rain removed
Training data	Clean	72.88%	24.73%	41.57%
	Rainy	35.34%	35.75%	34.66%
	Rain removed	69.75%	67.00%	67.96%

Table 8.2: Intersection Over Union of semantic segmentation on the Cityscapes validation dataset: table shows the results in relation to the training data used for the semantic segmentation network.

IoU		Test data		
		Clean	Rainy	Rain removed
Training data	Clean	62.59%	15.00%	27.50%
	Rainy	29.48%	29.31%	27.85%
	Rain removed	58.03%	56.28%	57.64%

As can be seen from the tables, for what concerns the segmentation with the model trained on "clean" (rain-free) images, the rain removal step helps to improve the performance with respect to direct segmentation of rainy images. While, as expected, with the "clean" validation images the segmentation algorithm performs better than the other cases. It is interesting to notice how the rain-removal pre-processing operation brings to an accuracy improvement of 16.84% and mIoU of 12.50% between the rainy images and the rain-removed ones.

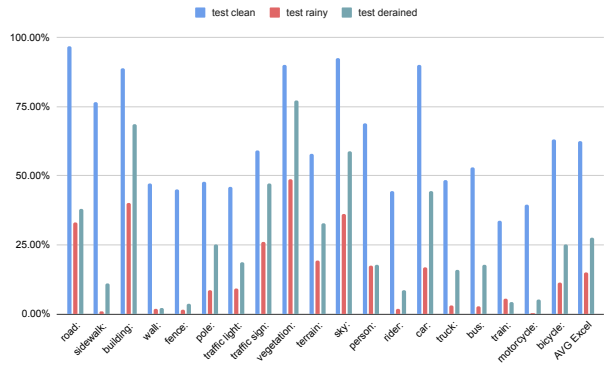
For what concerns the other two training cases, the one with the rainy training set and the one with the rain-removed training set, it's possible to observe a different behavior. In both of the cases, independently from the validation dataset used, the performance of the segmentation model in terms of accuracy an IoU does not change significantly. This behavior can be related to the amount of information present in the images used for training. In the first of these two cases, the network trained with the rainy images is not able to perform better than 36% in terms of accuracy, and 30% in terms of mIoU. Since during the training phase, part of the information in each image is always occluded or corrupted, the model is not capable to learn correct feature extraction for the classification of some specific classes. Looking at the per-class mIoU analysis in Figure 8.4b, some of the classes have mIoU of 0% or values very near to zero. As can be seen, for the three validation sets the situation is the same for all the classes, behavior that shows how the limited capability of the network to correctly segment element of the images is related to the missing information during training time, and not related to the type of validation data.

Similar behavior can be observed with the model trained with the rain-removed images. In this case, the performance improves in terms of accuracy and IoU due to the partially restored information in the training set, after the use of the rain-removal model. However, the general behavior is the same as the previous case: even if the model is tested with "clean" images, is not able to perform better than the other two validation set cases, due to the missing knowledge in the training images.

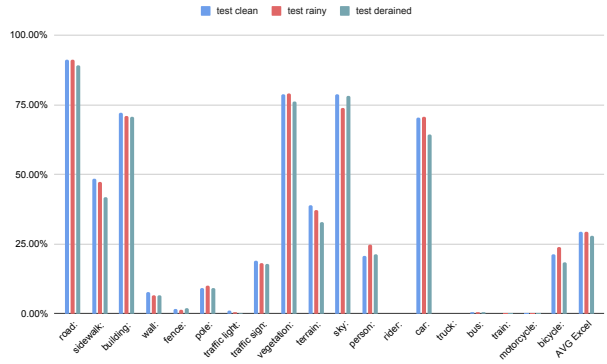
In Figure 8.4c it is possible see the per-class analysis: it is easy to observe the same behavior of the model trained with the rainy images, but it is also possible to observe an improvement in the segmentation of classes that were not recognized by the model trained with rainy images. This improvement is related to the enhancement of the training data due to the pre-processing step over the training set. Aside, it is interesting how the training with the rain-removed images has improved the results of the segmentation model with respect to the one trained with "clean" images.

8.3.3 Visual inspection

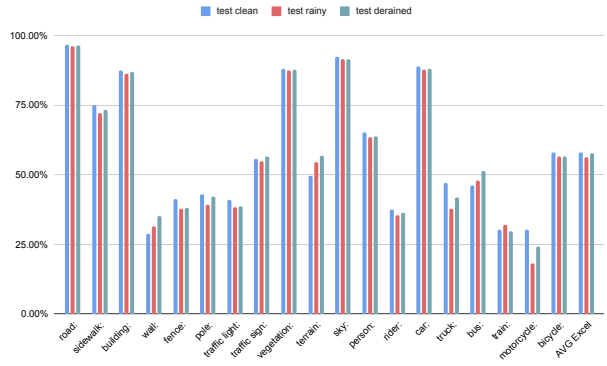
Here is presented a visual inspection of the impact of rain and rain-removal techniques over semantic segmentation. Figure 8.5 clearly shows the deterioration in prediction accuracy introduced by rain-related artifacts (row c).



(a) Clean training set



(b) Rainy training set



(c) Rain-removed training set

Figure 8.4: Mean intersection over union value for each class of the Cityscapes test set

The strong texture of rain streaks completely changes the interpretation of the road and sidewalk areas, which are mistaken as an obstacle (wall/fence). This phenomenon occurs despite the strong intrinsic bias of the "road" class, that appears in the plurality of training data pixels, and that occupies a consistent area throughout different images. On top of this, small regions such as the traffic signs and far-away vehicles are completely missed.

By processing the rain-augmented image using our rain-removal network, it is possible to partially restore the accuracy of semantic segmentation in some of the areas. As Figure 8.5d shows, the segmentation of the small cars is almost completely recovered, and part of the road and sidewalk are correctly identified. A qualitative and subjective evaluation of rain removal on the RGB image shows arguably less impressive results, suggesting a disconnect between perceived quality and usefulness for computer vision.

The best results, however, are obtained by retraining the semantic segmentation network using images that were processed with the rain-augmentation pipeline and subsequent rain-removal. In this case, the prediction in the same scenario, as depicted in Figure 8.5e shows an excellent restoration of several details, although some imperfections remain in the top-right corner of the example image.

For the sake of completeness, a qualitative evaluation is performed over two out-of-dataset real-life pictures, depicted in Figure 8.6. Specifically, are reported semantic segmentation results over the original rainy images trained with the "clean" version of the Cityscapes dataset (column a), and results over rain-removed images trained with the rain-removed version of Cityscapes (column b). These two extreme cases show the significant improvement in segmentation quality that can be obtained by the joint application of our rain removal network both on training data and inference data. The final results still show some imperfections, which can be attributed to the different nature of the image data when compared to the training set, both concerning rain appearance, as well as the general content and format of the pictures.

8.4 Optical Character Recognition (OCR)

Optical character recognition (OCR) is defined as the automatic conversion of images of typed, handwritten or printed text into machine-encoded text. OCR technology is used in a wide variety of scenarios, from the automatic detection of text from scanned document containing printed text as well as handwritten

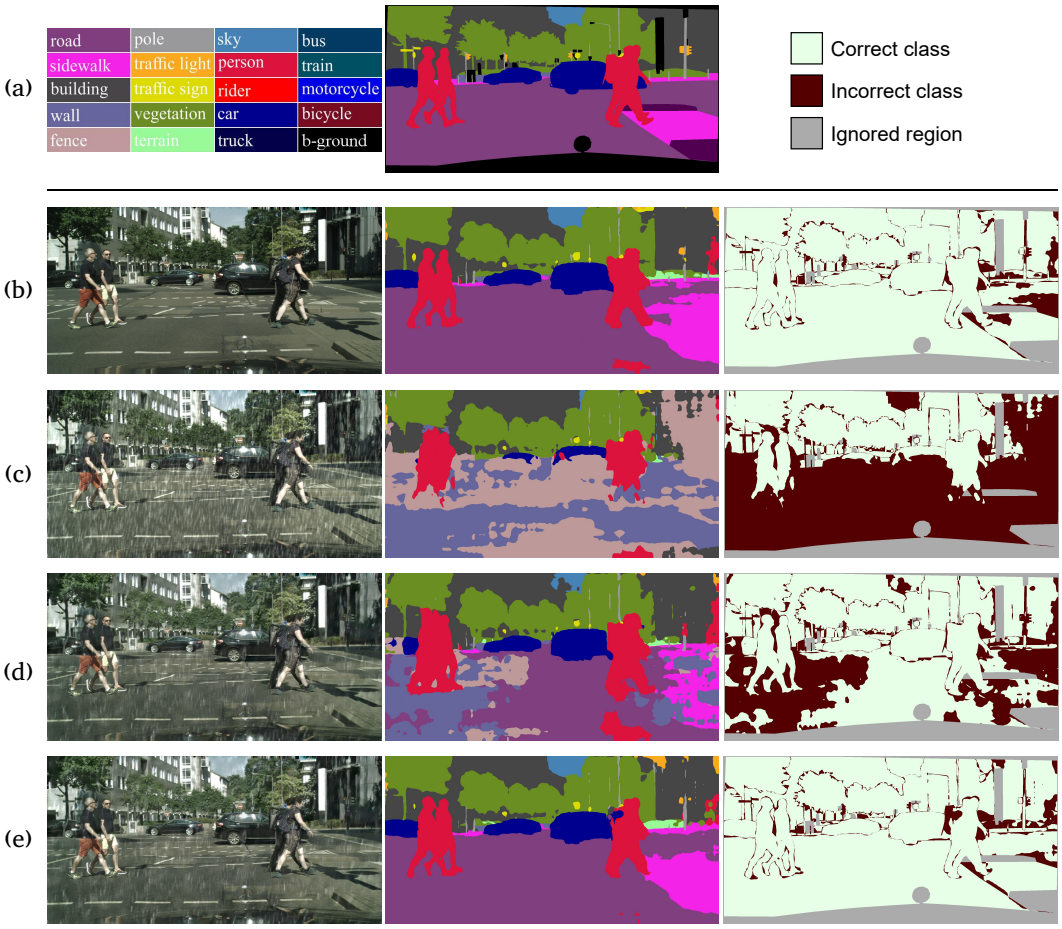


Figure 8.5: Impact of rain on semantic segmentation. Row (a) presents the color coding for semantic segmentation, the ground truth for the analyzed image, and the legend for error visualization. Rows (b) to (e) show, respectively, the prediction on the original "clean" image, on the image with artificial rain, on the image with rain removed, and once again on the image with rain removed but using a semantic segmentation model trained on images processed for rain and subsequent rain removal.

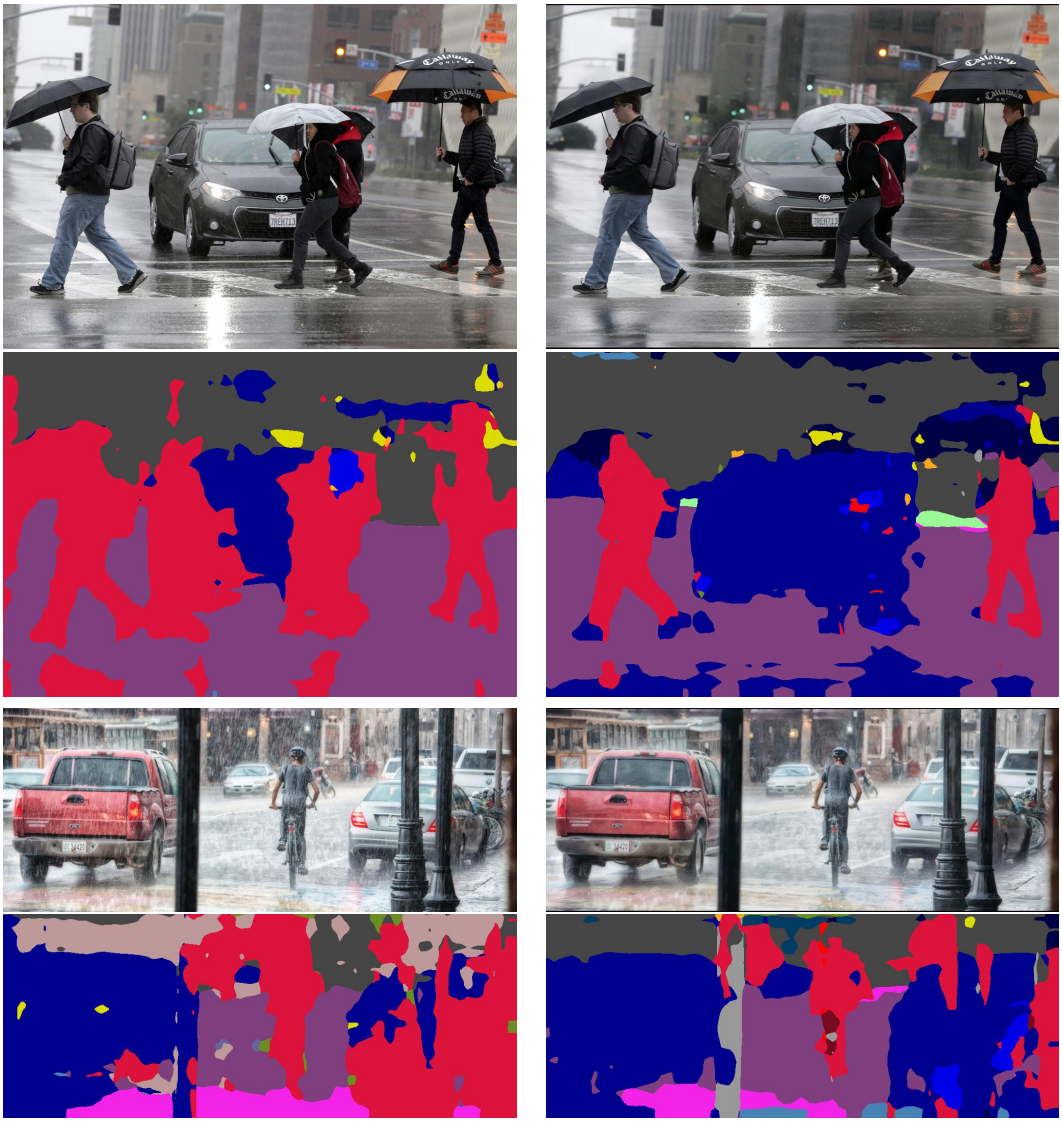


Figure 8.6: Visual assessment of rain (column a) and rain-removal (column b) over real case images, using semantic segmentation trained respectively on "clean" images, and images processed for rain-removal. Original images credit Nick Út, and Genaro Servín.

sentences, to text recognition from photographs of documents, maybe taken with smartphone cameras, or to recognition of text in photographed scenes.

In this second scenario, the focus is to analyze the impact of occlusions such as rain streaks and rain induced haze in processes like optical character recognition. To do so, the proposed model for rain streaks removal has been tested on data specifically collected and labeled for the OCR performances analysis, from two different points of view: perceived quality and text recognition accuracy, before and after the image restoration step.

8.4.1 Rainy Street View Images Synthesizing

In order to test the capability of the processing operation to improve the results of OCR methods, the performances of the proposed model have been tested using images of street scenes containing text areas. To this end, the Street View Text Dataset[176], which contains 350 images taken from Google Street View with high variability in text from signs, have been selected.

Since none of these images has been taken in bad weather conditions (such as rainy days, presence of haze or snow) each image have been synthetically augmented with the procedure described in section 8.2.5. A set of parameters are randomly chosen in a range of possible values, empirically defined, in order to obtain the most realistic rainy images possible. The resulting dataset has been called Rainy Street View Text Dataset (R-SVTD). Some examples of the R-SVTD are shown in Figure 8.7.

8.4.2 Quality Comparison

The first comparison has been done using the most commonly used full reference image quality metrics, i.e. PSNR and SSIM. In Table 8.3 is reported the comparison of the proposed method with other four methods in the state of the art: Fu et al. CNN[64] and DDN [65], Yang et al. JORDER[188] and Zhang et al. [194].

As can be seen from the table, the proposed method shows better results in terms of both the image quality metrics considered: with respect to the state of the art methods, the proposed solution achieved an improvement of +1.5328 dB in terms of PSNR and +0.0027 in terms of SSIM, while with respect to the rainy input images the improvement of quality corresponds to +3.9642 dB and +0.0911 respectively for PSNR and SSIM.

Figure 8.7: Some images from the R-SVTD after the application of the random rain mask with MATLAB. To improve the quality of the images, the mask has been created by combining synthesized streaks and haze.

Table 8.3: Comparison of the methods in terms of PSNR and SSIM indexes for the Rainy Street View Text Dataset.

	PSNR	SSIM
Rainy	20.8128	0.7794
CNN [64]	17.6142	0.6196
DDN [65]	23.0897	0.8678
JORDER [188]	18.5631	0.7522
DID-MDN [194]	23.2442	0.8343
Proposed method	**24.7770**	**0.8705**

8.4.3 OCR test

Due to the lack of the possibility to make a quantitative comparison of the model in terms of accuracy in text detection and recognition, the decision has been the one of adopting the OCR system provided by Google Cloud Vision API for a visual comparison. In Figure 8.8 there are some images and their text detection results before and after the application of the proposed deraining method.

As can be seen from the examples reported, the proposed deraining method tends to improve the results of the OCR. In most of the cases, the OCR is able to detect text areas that were not detected before, even if the text recognition is not always completely correct. This improvement can be seen mainly in the case of *heavy rain* conditions while in general in the other cases the improvement is not that significant since the OCR used is capable to correctly detect the text area. In those cases, the proposed deraining method improves the recognition of few letters with respect to the rainy version. In the 42% of the cases, i.e. 147 out of 350 images from the Rainy Street View Text Dataset, the rain removal processing step improved the results in terms of both text detection and recognition.

8.5 Summary

In this section the rainstreaks removal task has been presented alongside with the definition of a Convolutional Neural Network model, based on the Pix2Pix model, for image restoration. Alongside with the classical image quality eval-

Figure 8.8: Some results over the Rainy Street View Text Dataset with the relative bounding boxes and detected texts.

uation of the results, the proposed method has been analyzed in terms of improvements in performances of other downstream tasks that may benefit from the presence of a pre-processing step: Semantic Segmentation and Optical Character Recognition. In order to perform the analysis a pipeline for synthetic rain generation has been defined in order to augment the Cityscapes dataset and the Street View Text dataset, obtaining different versions of the datasets for both training and testing.

Concerning the Optical Character Recognition task analysis, the proposed model has been compared on a dataset composed by street view scenes, to which have been added synthetically generated rain. This dataset has been called Rainy Street View Text Dataset (R-SVTD). The comparisons with the other state of the art methods shown that the proposed model outperforms the previous obtained results in terms of PSNR and SSIM indexes, with a respective improvement of +1.5328 dB and +0.0027. Tests on the R-SVTD dataset showed how the model is capable to restore the structures of the degraded images in order to improve the results of an OCR model used after the restoration.

Regarding the Semantic Segmentation task analysis, the proposed rain removal model has been trained in three different conditions: with "clean" images, with images with artificial rain streaks, and with images processed for removal of the artificial rain streaks. In the first case the experiments show how the application of rain-removal on rainy images gives benefit for the segmentation step of a model trained in optimal image conditions. The other experiments, regarding the impact of the degraded information in the images used for training the model, shows how the application of an enhancement algorithm can improve the performance of the model at inference time. We observed an improvement of 34% of accuracy and 30% of mIoU between the model trained with the degraded images and the one trained with the enhanced ones.

In the end have been obtained promising results that encourage to continue working in this direction with a major focus on optimization of those methods, specifically for those kind of tasks limited by the nature of the images.

As future step, is necessary to put the attention on some points that have been highlighted by the experiments done. At the moment the model is trained for the reconstruction of general content images since the training has been performed on those kinds of contents. A first step can be related to the training of a model for the removal of rain in relation to the specific content or information to restore, related to the subsequent downstream task. A second

point is related to the fact that not in all of the cases the processing operation gained some improvement. In some cases, the models tend to introduce artifacts. For example, in the OCR task analysis, the text that was originally well recognized, change for some letters because of the wrong enhancement during the processing. Putting attention on that fact, the next step should be related to the reduction of undesired artifacts in the enhancement operation. Another possible route to follow is the one that considers the use the results of the downstream tasks as objective functions for the training of the models, with the purpose to obtain CNNs to specifically improve the results related to the next step of image analysis or processing.

Chapter 9

Raindrop Removal From Camera Lenses

Adverse weather conditions negatively impact the perceived visibility of a scene. In a street-driving scenario, for example, rain droplets adhering to the glass surface of a car windshield might occlude crucial elements such as obstacles, pedestrians, or traffic signs, and are generally distracting to the driving experience. In addition to hindering human vision, rain-induced artifacts are also found to affect computer vision: several works in the the scientific literature quantify the benefits of digital rain removal on a wide variety of tasks, ranging from object detection [111], to semantic segmentation [199], to optical character recognition [201]. The problem of image regression, in its most general formulation, has been addressed with a wide variety of approaches through the years: from handcrafted solutions [184], to the more recent exploitation of Convolutional Neural Networks (CNN) [105]. The latter has involved general-purpose methods for image-to-image translation [88], as well as domain-specific architectures such as the ones described in Section 9.1. Compared to other artifacts such as rain streaks and rain mist, rain droplets impose a significant and specific set of challenges, such as large occlusion areas and a wide variety of appearances, as show in Section 9.2. This work is specifically focused on raindrop removal by designing a Laplacian encoder-decoder neural network. The proposed solution allows to control the image reconstruction process by producing the different levels of a Laplacian pyramid decomposition of the expected clear (i.e. rain-free) image. This approach avoids relying on commonly-used attention maps which are inherently limited by misalignements between the rainy and clear image, a phenomenon observed

by [9]. The proposed model is trained with multiple losses, evaluating the partial reconstruction at each level of the pyramid. In the training procedure, the derivation tree has been modified in order to prevent redundant gradient flow for pyramid levels that impact more than one loss component. This novel formulation, and its integration with the Laplacian decomposition, was found to be optimal after comparative evaluation with several other alternatives, which are reported in the experimental results. Future developments of the proposed method are also suggested based on an in-depth analysis of the relationship between Laplacian levels and rain removal.

9.1 Related works

The digital removal of rain-induced artifacts has been actively studied through the years, resulting in an extensive scientific production. A recent review by [189] presented a comprehensive analysis, ranging from solutions based on explicit raindrop-appearance models ([71, 121, 188]), to data-driven ones (typically relying on deep learning: [64, 195]). [175] produced a similar overview of existing approaches, with particular attention to rain removal in video sequences, providing direct links to papers, source codes, project pages, datasets, and metrics. Rain-related artifacts can be organized in three macro categories according to [111]: rain droplets, rain streaks, and rain mist. An image is said to be affected by "rain droplets" (also referred to as raindrops) when the scene is observed through wet glass: typically, a car windshield right after it rained. This particular interpretation of the problem is the one addressed in this paper, therefore an analysis of corresponding state of the art solutions is provided in the current section. The term "rain streaks" refers instead to the visual artifacts of a scene directly observed when rain is currently pouring [21, 124, 93]. In this case, the terminal velocity of falling rain produces motion-blurred rain streaks overimposed to the image. Finally, the task of rain-streak removal is often treated in conjunction with the correction of "rain-induced mist": in the same scenario, in fact, rain streaks that are far away from the camera are not individually discernible, and produce a global appearance equivalent to that of airborne water [56]. Additionally, some recent works have focused on the digital removal of snow flakes, training regression models either through adversarial techniques [92], or more traditional learning procedures [118].

The specific field of raindrop removal is relatively recent, compared to rain streak and mist removal. [184] developed an handcrafted approach to the

problem: they focused on droplet detection, by analyzing color, texture, and shape statistics of raindrop images. Based on these features, their solution is to produce a first set of candidate raindrop regions, which is subsequently pruned through a learning-based verification algorithm. The authors then resorted to existing image inpainting solutions in order to restore the selected image areas. A relevant contribution to the field has then been given by [136], who in 2018 published a high-quality dataset that has since become the *de facto* standard for this area of research. The authors also introduced a so-called "attentive generative network", trained in an adversarial configuration. They injected a visual-attention map to both the generative and discriminative component of the network, in order to focus the image processing mainly on corrupted areas. However, whenever attention maps are designed to target explicit raindrop masks (a function of the difference between rainy image and clear reference), they are inherently limited by misalignements between the two images and moving objects, as observed by [9]. They consequently developed a physically-accurate computer-graphics engine to augment images with artificial raindrops. Such technique allowed them to exploit existing datasets unrelated to rain removal, in order to train a model that is able to simultaneously locate and remove raindrops in a self-supervised manner. Their solution, based on a conditional generative adversarial network, is mainly developed for application to video sequences by exploiting motion cues. [140] devised a so-called "double attention mechanism" to guide the learning and inference of a Convolutional Neural Network in the task of raindrop removal. Their approach relies upon the generation of a shape-driven attention map, to locate raindrops based on a-priori knowledge on their shape properties. Such attention map was applied using a channel recalibration mechanism, to properly weight the intermediate activations of their neural model. [80] released a dataset of images augmented with physics-based synthetic raindrops, as well as the associated raindrop masks. They defined a neural network for raindrop detection which explicitly models the refraction and blurring components of the raindrop itself. [150] explicitly modelled the blur level of rain droplets using a soft mask populated through an iterative procedure, and fuse it with the input image through an attention mechanism. They also exploit multi-scale analysis based on the observation that different scale versions of a rainy image have similar raindrop patterns. In developing the final solution, experiments with different strategies have been performed, and eventually have been defined a neural network that, while not relying on attention maps, still produces competitive or even superior results when compared to such methods.

More specifically, the proposed approach to the digital removal of rain droplets leverages a Laplacian decomposition of the input image, in order to address the problem at different scales. Decomposing the input image with various representations has been successfully exploited in the past for rain streak removal while not for raindrop removal. [99] applied an image decomposition based on morphological component analysis, specifically resorting to bilateral filtering. They decomposed the image into a low-frequency and high-frequency part, and focused on processing only the high-frequency component: they exploited dictionary learning and sparse coding to further decompose it into rain and non-rain components, in order to effectively remove the former. Similarly, [161] also devised an approach that relies on image decomposition for dictionary-based removal of rain streaks, but embedded and formulated the decomposition-basis selection as an optimization problem instead of exploiting bilateral filtering. [66] focused on reducing the computational complexity of Convolutional Neural Networks for rain streaks removal by representing the input image as a Gaussian-Laplacian pyramid, and by designing a so-called "Lightweight Pyramid Network" (LPNet) based on a recursive and residual structure.

This is the first time that image decomposition is exploited for raindrop removal. In Section 9.2 and Section 9.3.3 is shown that this application is particularly appropriate, as the variety of appearances of rain droplets can be individually handled by exploiting the Laplacian-based image decomposition.

9.2 Proposed method for raindrop removal

Raindrops adhering to a transparent surface in front of the camera (like a car windshield, or the camera lens itself) degrade the quality of the information contained in the picture to different extents, depending on the camera focus:

1. In-focus raindrops. The degradation is manifested as blur over specific areas of the image, affecting both low and high frequencies.

2. Out-of-focus raindrops. Two main effects can be identified:

    - A degradation related to the refraction phenomena introduced by the convex shape of the drop, that affects the low-frequencies.
    - A drop contour degradation that is manifested as artifacts in the high frequencies.

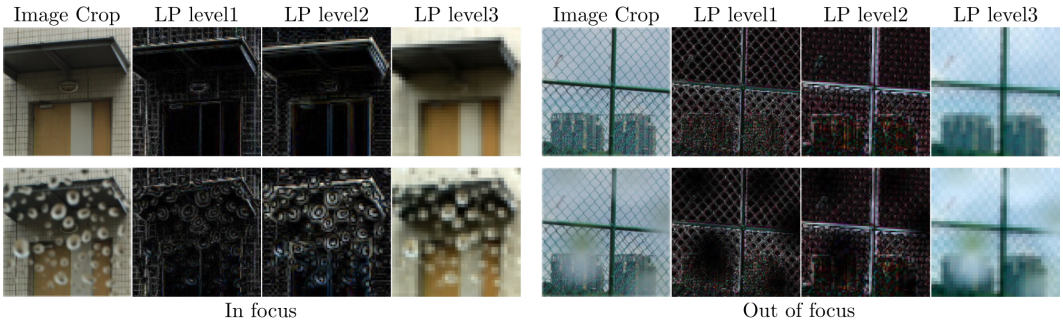


Figure 9.1: Different kinds of raindrop and their impact on the overall image. The ones in camera focus tend to introduce artifacts related to the sharp edges of the single raindrops in combination with the refraction phenomenon. The ones out of focus tend to remove information where the drops are located, by blurring the corresponding image areas.

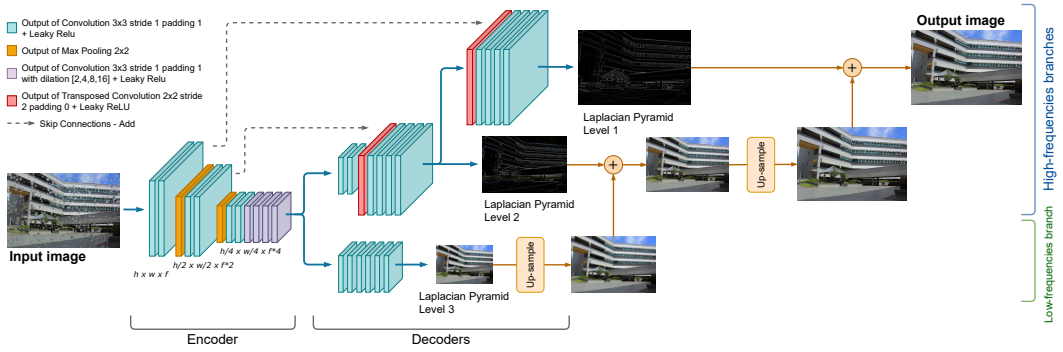


Figure 9.2: Architecture of the proposed Laplacian Raindrop Removal CNN. The number of features in output after the first convolution is set to $f = 64$. The output of the different levels is combined by up-sampling the lower levels and summing them to the higher frequencies, in order to obtain the final output.

An example of in-focus and out-of-focus raindrops can be seen in the second row of Figure 9.1. To model the degradation distribution over different frequencies of the input image, the approach exploits the image Laplacian pyramid decomposition [34], whose effect is also depicted in figure.

9.2.1 Laplacian-based image restoration

Given an input image I_{rainy}, the proposed encoder-decoder network G is designed and trained to generate the corresponding levels $\hat{y}_i$ of its Laplacian pyramid decomposition, free of rain artifacts:

$$\hat{Y} = \{\hat{y}_1, \hat{y}_2, ..., \hat{y}_N\} = G(I_{rainy}) \tag{9.1}$$

where $\hat{y}_N$ is the tallest level of the Laplacian pyramid, corresponding to the low frequencies component. The final recomposed output $I_{derained}$ is then computed as:

$$I_{derained} = L_{\hat{Y}}(1) \tag{9.2}$$

$$L_Y(j) = \begin{cases} y_N & \text{if } j = N \\ y_j + upsample(L_Y(j+1)) & \text{otherwise} \end{cases} \tag{9.3}$$

The proposed network architecture, depicted in Figure 9.2, is divided in two main components: an encoder for the input I_{drop}, and a novel decoder composed of multiple output branches, in relation to the specific formulation of Laplacian pyramid levels.

The design of the encoder is partially inspired from the U-net model by [143], with some relevant variations in the convolution and general structure. More specifically, the encoder is a sequence of two CONV-lReLU-CONV-lReLU layers with a MaxPool operation, to extract features and to reduce the spatial dimension. The activations are not reduced to spatial dimensions 1×1 as in the original U-Net architecture, but only reduced down by a factor of 4 (given by the presence of the two MaxPooling operation), to avoid losing spatial information in the encoded features, which serves an important role in image restoration.

The deepest part is a sequence of two CONV-lReLU blocks and four CONV-lReLU blocks with dilation ([192]): these last six blocks of layers have been added with respect to the original U-Net encoder structure, to increase the model receptive field without reducing further the spatial feature dimensionality. The dilation spacing increases as a power of two from the first layer to the last one (2, 4, 8, 16). The depth of output features after the first convolution is set to $f = 64$, and the following ones are derived as indicated in Figure 9.2.

The decoder, which addresses the actual restoration of the information at different frequency bands, has been designed in relation to the number N of levels of the Laplacian pyramid which have to be reconstructed. In the

experiments, N has ebeen set to 3. The decoder is composed of branches of two types: one dedicated to restoration of low-frequencies, two dedicated to high-frequencies. The low-frequencies branch is a concatenation of six CONV-lReLU with a final CONV(1×1) layer with a Sigmoid activation function to map from the features space to the RGB color space. The output of this branch corresponds to the deepest level of the Laplacian pyramid, which is a low-resolution version of the rain-free image, and which will be combined with the highest levels generated by the model according to Equation 9.1. The high-frequencies branches are designed to restore the details and the fine structures in the image. The corresponding Laplacian pyramid levels all share common characteristics: values centered around zero and a general appearance that is not as intelligible as that of the lower frequencies. For this reason, the structure of this part of the model is composed of multiple sub-branches that incrementally enhance the features from the deepest to the highest level of the laplacian pyramid. Each higher branch is an extension with respect to the previous level in the Laplacian pyramid, i.e. it takes the features decoded by the preceding level to restore its own. The decoder blocks are composed of four CONV-lReLU layers plus a transposed convolution, to upsample the features for the higher Laplacian level, and a CONV(1×1) with a Tanh activation function to map from feature space to RGB color space.

9.2.2 Laplacian loss function

Given a target rain-free image I_{clear}, the corresponding Laplacian pyramid levels Y are extracted to be compared with the restored output $\hat{Y}$. The proposed loss function $Loss_d$ reconstructs the restored image up to each level, and compares it with the corresponding reconstructed target using the L1 norm ($||\cdot||_1$):

$$Loss_d = \sum_{i=1}^{N} ||L_Y(i) - L_{\hat{Y}}(i)||_1 \tag{9.4}$$

Instead of directly comparing the generated frequencies with the target ones, the proposed approach reconstructs the image up to the specific level at which the comparison takes place. This strategy, experimentally validated in Section 9.3.2, is motivated by two purposes:

- Levels balance: the comparison at each branch is always performed on complete RGB images. This guarantees a magnitude of error similar

between the different branches, without the necessity to re-weight to the different components of the loss for regularization purposes.

- Reconstruction context: instead of comparing images composed only of details taken out of their original context, the comparison using the reconstructed level can highlight differences in relation to the context in which the details are located. This is expected to help the training in detecting structures introduced by raindrops, in contrast to the ones coming from elements of the actual scene.

It should be noted that, with the formulation expressed in Equation 9.4, the evaluation of every Laplacian level impacts all the lower levels during the gradient back-propagation. Therefore, higher levels effectively influence the learning process multiple times. In order to prevent this phenomenon, the flow of gradients during the training process has been modified, by inhibiting the back-propagation on the lower branches, and maintaining it only for the current branch.

9.3 Experiments

9.3.1 Experimental setup

To train and test the proposed model for raindrop removal, the dataset and metodology adopted are the ones presented by [136]. This dataset has been collected with the same procedure from [57] by placing a glass panel in front of a camera, and taking pictures before and after spraying the glass with water. The dataset contains a total amount of 1119 pairs of images depicting different outdoor scenes. The dataset is divided into three main folders: the *train* folder, containing 861 pairs of images, and two test folders: *test_b* (249 image pairs) and *test_a* (58 image pairs, a subset of well-aligned images from *test_b*). The folder *test_a* is commonly used for methods assessment and comparisons, as done by [136, 140, 131, 80, 134]. The folder *test_b* (without the images contained in *test_a*) is commonly used for internal validation.

The proposed encoder-decoder network has been trained with images from the *train* folder, cropped at dimension 256×256 pixels. The crops have been collected using a sliding window with overlap equal to 128 pixels, generating a total amount of 20664 training samples. To further augment the dataset, online flipping and rotation (90°, 180°, 270°) have been randomly applied to

the images at training time. For validation and test, *test_b* and *test_a* folders have been used, as done by [136], [140] and [80]. The model is written in PyTorch v1.4.0, trained using an NVIDIA Titan V GPU with 12 GB of RAM. The optimizer adopted is the Adam optimizer [104] with $\beta_1 = 0.9, \beta_2 = 0.999$ with a starting learning rate $lr = 2 \cdot 10^{-5}$ decreased by a factor $10\times$ after 300 epochs of training, and weight decay set to 10^{-8}.

Quantitative evaluation is performed using two full-reference quality assessment metrics: Peak Signal-to-Noise Ratio (PSNR) [180] and Structural Similarity Index Measure (SSIM) [181], both computed on the luminance channel of images in the YCbCr color space. To be noted that SSIM was proven by [12] to be better correlated with human opinion scores, compared to PSNR.

9.3.2 Evaluation of alternative training configurations

The Laplacian loss function $Loss_d$ defined in Section 9.2.2 based on the proposed encoder-decoder network have been compared with a baseline devoid of any Laplacian decomposition, and with two alternative loss functions that do exploit the decomposition, but combine the resulting levels in different ways. All four configurations, depicted in Figure 9.3, exploit the L1 norm to perform the output-target comparisons, and are described in the following:

- Configuration *a*: a single loss for a classical encoder-decoder model without Laplacian decomposition. Here the decoder defined in Section 9.2 has been completely replaced with a specular version of the proposed encoder.

- Configuration *b*: a single loss exploiting the proposed Laplacian encoder-decoder, but comparing only the final fully-reconstructed output with the target image.

$$Loss_b = ||L_Y(1) - L_{\hat{Y}}(1)||_1 \tag{9.5}$$

- Configuration *c*: multiple sub-losses exploiting the proposed Laplacian encoder-decoder, evaluating the output of each level individually, but without the intermediate reconstruction.

$$Loss_c = \sum_{i=1}^{N} ||y_i - \hat{y}_i||_1 \tag{9.6}$$

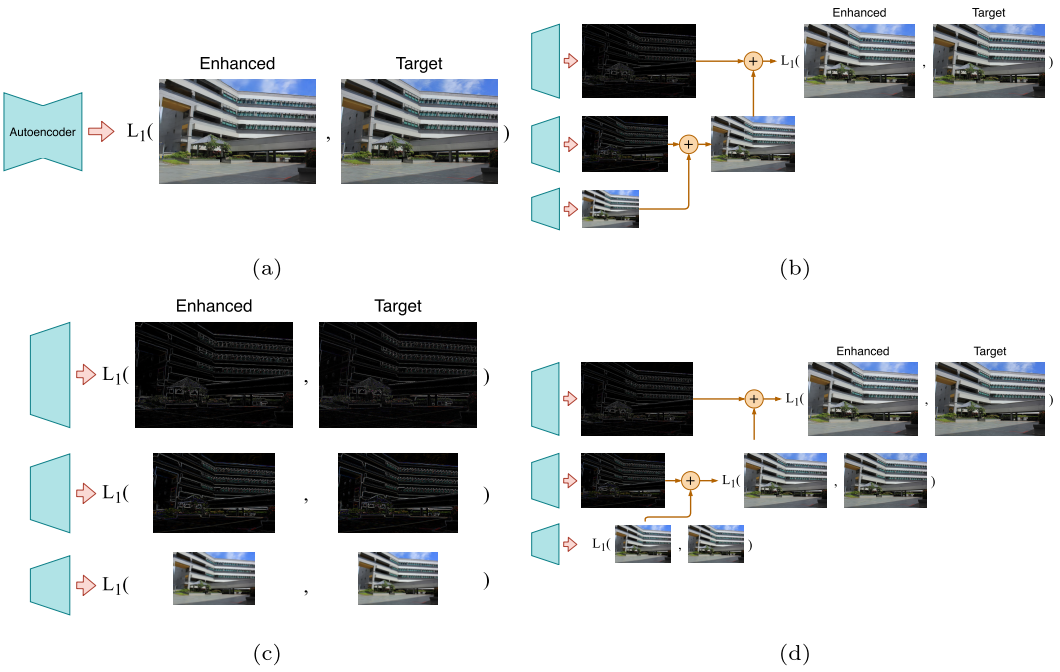


Figure 9.3: Analysis of different training configurations for encoder-decoder network: (a) comparison between the output of a classic encoder-decoder model and the target, (b) comparison between the reconstructed output and the target, (c) comparison between each level output and the corresponding target level, (d) comparison between each reconstructed level of the pyramid and the corresponding target.

- Configuration *d*: multiple sub-losses exploiting the proposed Laplacian encoder-decoder, reconstructing the image at each level. This is the definitive configuration, combining the sub-losses according to Equation 9.4.

Table 9.1 reports the results of the correspondingly-trained models in terms of PSNR and SSIM. It is possible to observe how the use of a loss function that is aware only of the final result (configuration *b*), is not enough to fully exploit the power of the Laplacian decomposition. Such a model, without a control on the output of the single levels, obtains worse results with respect to a single encoder-decoder trained with the same distance function and with no Laplacian decomposition (configuration *a*). Configuration *c* compares the results of each branch with the corresponding target version, but without the

Table 9.1: Study on the training configuration: results achieved training the proposed model using the different loss function configurations. Evaluation performed on *test_a* from the dataset by [136].

Training configuration	PSNR	SSIM
a: Single loss, classical encoder-decoder	30.14	0.9198
b: Single loss, reconstructed image	29.76	0.9200
c: Multiple sub-losses, individual levels	30.56	0.9252
d: Multiple sub-losses, reconstructed levels	**31.12**	**0.9297**

Laplacian reconstruction step for the corresponding levels. In this case, an improvement is obtained with respect to both single-loss configurations *a* and *b*. In this configuration, the training of each branch is directly related to the reconstruction of a certain frequency band: each level is thus focused on the restoration of certain details, without considering the other branches' contribution to the final restored image. However, due to the different nature of the images generated at the different branches (low frequencies and high frequencies), the magnitude of the sub-losses during training is different. Without any weighting-based regularization, therefore, this configuration is potentially suboptimal. The final version (configuration *d*) evaluates the results of each branch, with respect to the lower levels results. Instead of simply comparing the branches' output, the image is first reconstructed up to the interested level, and only then the loss is calculated. In this way, it is possible to compare the results of each layer with the corresponding targets, and at the same time, the losses at the different levels have similar magnitude. Moreover, with this kind of image evaluation the details introduced by each branch are compared with the target in relation to the general context of the image in which they are located, instead of comparing only the map of details modified by the neural network. This helps the neural network to better identify the presence of raindrops that must be removed, in comparison with textures coming from the original scene.

9.3.3 Laplacian decomposition assessment

The impact of Laplacian decomposition on rain removal have been assessed, by decomposing *test_a* rainy images as described in Section 9.2, and replacing

Table 9.2: Effect of replacing each Laplacian level with its perfect ground truth version. For the "Rainy" columns, "base images" refers to the original images in *test_a* from [136]. For "Derained", "base images" refers to the output of the proposed rain removal network.

	Rainy		Derained	
	PSNR	SSIM	PSNR	SSIM
Base images	24.10	0.8511	31.12	0.9297
Perfect level 1	24.61 (+2.1%)	0.9181 (+7.9%)	32.12 (+3.2%)	0.9781 (+5.2%)
Perfect level 2	24.60 (+2.1%)	0.8749 (+2.8%)	31.22 (+0.3%)	0.9402 (+1.1%)
Perfect level 3	28.95 (+20.1%)	0.8885 (+4.4%)	33.92 (+9.0%)	0.9389 (+1.0%)

each level independently with the corresponding clear version (i.e. the ground truth). For each resulting version, both PSNR and SSIM metrics are computed and compared with the full ground truth images. The results are shown in Table 9.2 with the "Rainy" columns.

PSNR reports the greatest potential advantage when resolving the problem at low frequencies (level 3). Conversely, the higher frequencies (level 1) have potentially the greatest impact on SSIM, which was in fact specifically designed to capture structural similarity. To be noted that the upper bound of SSIM is 1, while PSNR has no upper bound. This first evaluation provides an indication of how different errors are distributed across multiple levels. It is also possible to observe that the proposed solution, reported as the "Derained" columns for the base images, outperforms all the individual level replacements for the rainy images, showing that it effectively brings an improvement at more than one level.

It's then possible to quantify the upper bound of improving the current solution one level at a time, i.e. by determining the potential impact of perfectly restoring either of the levels from derained images. This is done, once again, by replacing each level individually with the corresponding one from the ground truth images. The results are shown in Table 9.2, with the "Derained" columns, and in Figure 9.4 for a view of the entire distribution. For SSIM, the largest possibility for improvement still appears to be working on level 1 (high frequencies). Interestingly enough, for PSNR can be observed a large potential improvement by working both on level 3 and level 1. In general, this suggests to focus on details at high frequencies, which would have a positive impact on both evaluation metrics, and which is left as a direction for future

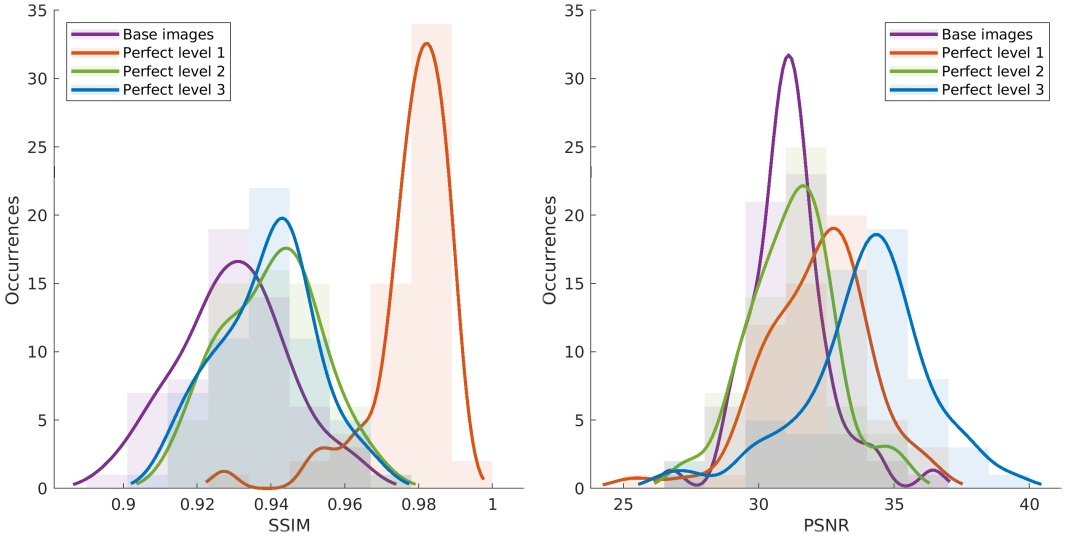


Figure 9.4: SSIM and PSNR distributions corresponding to replacing each Laplacian level of derained images ("base images") with a perfect version from the ground truth. The comparison is always performed with the full ground truth. Kernel Density Estimation [145] is applied to the distributions to facilitate interpretability.

research.

9.3.4 Comparison with the state of the art

Here a comparison of the proposed Laplacian encoder-decoder network is done with state of the art methods for raindrop removal: [57], AttentiveGAN by [136], [140], [80], DURN by [117], [131], [150] and an image-to-image general purpose method, named Pix2Pix [88]. The comparison, done in terms of standard measures PSNR and SSIM on *test_a* and *test_b* from [136], is reported in Table 9.3, while Figure 9.5 presents a visualization in Cartesian representation (PSNR on x-axis and SSIM on y-axis). For both metrics, the higher, the better restoration.

As it can be seen in Table 9.3, the proposed method outperforms the state of the art solutions on the standard test set *test_a* in terms of SSIM, and achieves comparable performance for PSNR. [9] report the results of their method on the *test_a* set of the same dataset in terms of PSNR and SSIM values as 31.94 and 0.945 respectively, thus obtaining good performance. However,

Table 9.3: Quantitative evaluation of methods for raindrop removal on *test_a* and *test_b* from the dataset by [136]. Results on *test_b* are reported from [150]. Best result in bold, second-best underlined.

	test_a		*test_b*	
Method	PSNR	SSIM	PSNR	SSIM
[57]	28.59	0.6726	-	-
Pix2pix - [88]	30.14	0.8299	23.50	0.7150
AttentiveGAN - [136]	**31.57**	0.9023	24.92	0.8090
[131]	30.72	0.9262	-	-
[140]	31.44	<u>0.9263</u>	-	-
[80]	30.17	0.9128	-	-
[134]	<u>31.55</u>	0.9020	-	-
DURN - [117]	31.24	0.9259	25.32	0.8173
[150]	31.47	0.9235	<u>25.35</u>	**0.8197**
Proposed method	31.12	**0.9297**	**25.40**	<u>0.8185</u>

Table 9.4: Comparison of average inference time for different methods. The proposed model was evaluated on an NVIDIA Titan V GPU. Other values are reported from [150], evaluated on an NVIDIA RTX 2080Ti GPU.

	Pix2Pix	AttentiveGAN	DuRN	Shao et al.	Proposed method
Time (s)	0.025	0.034	0.018	0.141	0.054

since their solution was trained on different data, the results are not directly comparable. To further analyze the performance of the proposed model on a larger dataset, the proposed solution have been tested on the *test_b* set from [136], comparing it with the results from other methods as reported by [150]. As can be seen in Table 9.3, the proposed model outperforms the state of the art in terms of PSNR, while the SSIM index reaches comparable results with the model by [150]. Furthermore, it is interesting to notice the performance drop of AttentiveGAN [136], which can be associated with sub-optimal generalization effectiveness, observed when testing the model with a

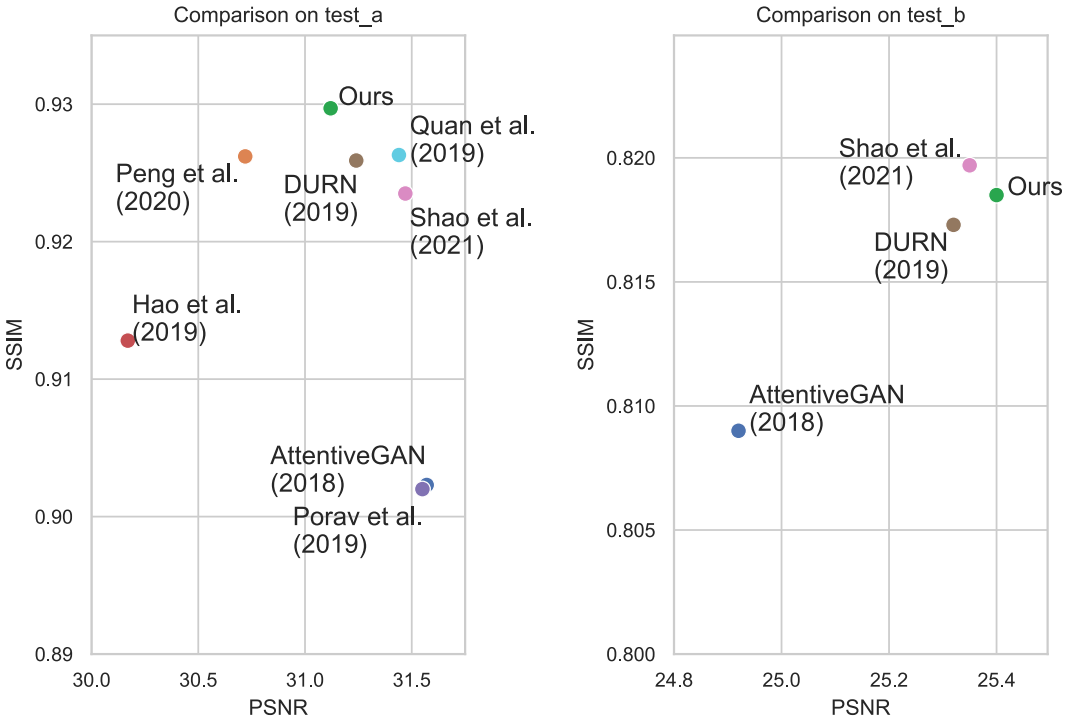


Figure 9.5: PSNR-SSIM comparison of the state-of-the-art-models and the proposed proposed method on *test_a* and *test_b* from [136]. For both metrics a higher value means better visual results.

higher number of images. In this sense, the proposed solution shows a more stable behaviour, being capable to better generalize, and perform generally better than the best performing ones on the smaller set *test_a*.

An additional term for comparison is to account for the average running time during the inference phase. In Table 9.4 is reported the timing assessment performed by [150] on various methods, using an NVIDIA RTX 2080Ti GPU, and compare these with the proposed model, evaluated using an NVIDIA Titan V GPU, which is an inferior hardware configuration. The observations that can be derived are limited by the differences in the experimental setup, however, it is possible to observe that the proposed solution is faster than the method by [150], while being in the same order of magnitude as the other compared methods.

Concerning a visual comparison of the models, in Figure 9.6 and 9.7 are reported some processed images from the *test_a* set. The comparison was performed against the methods from [136] (AttentiveGAN) and [140], whose

code and models are publicly available. The images were selected in order to highlight a variety of scene and droplet types. It is possible to observe that the proposed encoder-decoder network produces a satisfactory restoration of homogeneous areas, as well as regions occluded by large out-of-focus droplets, while maintaining little-to-no artifacts related to refraction phenomena. To further prove the effectiveness of the proposed solution, a test of the model on an out-of-dataset scenario has bee performed. In Figure 9.8 are shown two frames from a video sequence captured using a car dash camera, during a storm. Once again the proposed approach is compared with AttentiveGAN by [136] and the model by [140]. As can be seen, the proposed solution is able to remove raindrops from the input images, which is particularly evident in the second reported frame, at the same time preserving details (trees from the first image) and avoiding the introduction of color artifacts (trees from the second image).

9.4 Summary

In this last part of the thesis is presented an encoder-decoder neural network for adherent raindrop removal, motivated by the perspective of improving the visibility of an acquired scene. The described neural architecture takes advantage of image decomposition, by generating the Laplacian pyramid levels of a rain-free version of the input image. This formulation deconstructs a problem that is inherently characterized by a variety of appearances, and allows the proposed model to potentially address each frequency band with a different strategy. Moreover, a loss function has been specifically designed in order to takes into account the different nature of each Laplacian level, and has been also shown its suitability in a comparison against other possible formulations of the loss function itself. The effectiveness of this solution was also demonstrated with respect to existing state of the art methods for raindrop removal, which are outperformed in terms of structural similarity on a standard test set.

To provide a direction for future research, investigative experiments have been conducted to understand what components of the image offer the greater chances at improving the model performance. The conclusion of these experiments is that both SSIM and PSNR measures would benefit significantly by focusing on the lowest level of the Laplacian pyramid, i.e. by improving the reconstruction of high frequencies. In general, the current model could be fur-

ther developed by exploiting alternative representations, and its efficacy could be evaluated on other types of weather-related artifacts, including rain streaks and snow.

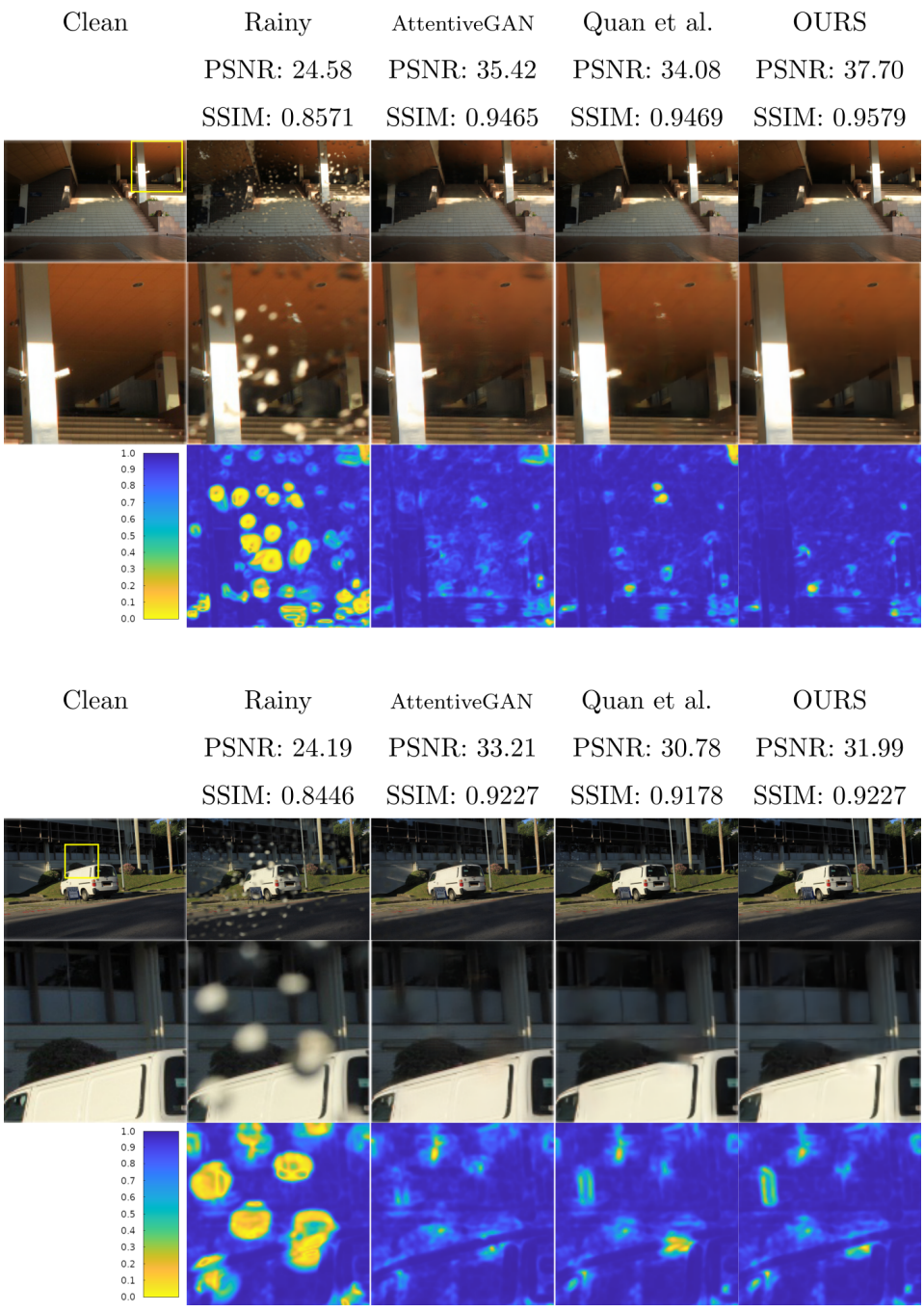


Figure 9.6: Visual comparison of methods for raindrop removal. The proposed proposed model correctly restores information on uniform areas and near edges coming from the original scene. Zoomed crops and the corresponding SSIM maps are reported to facilitate the results interpretation.

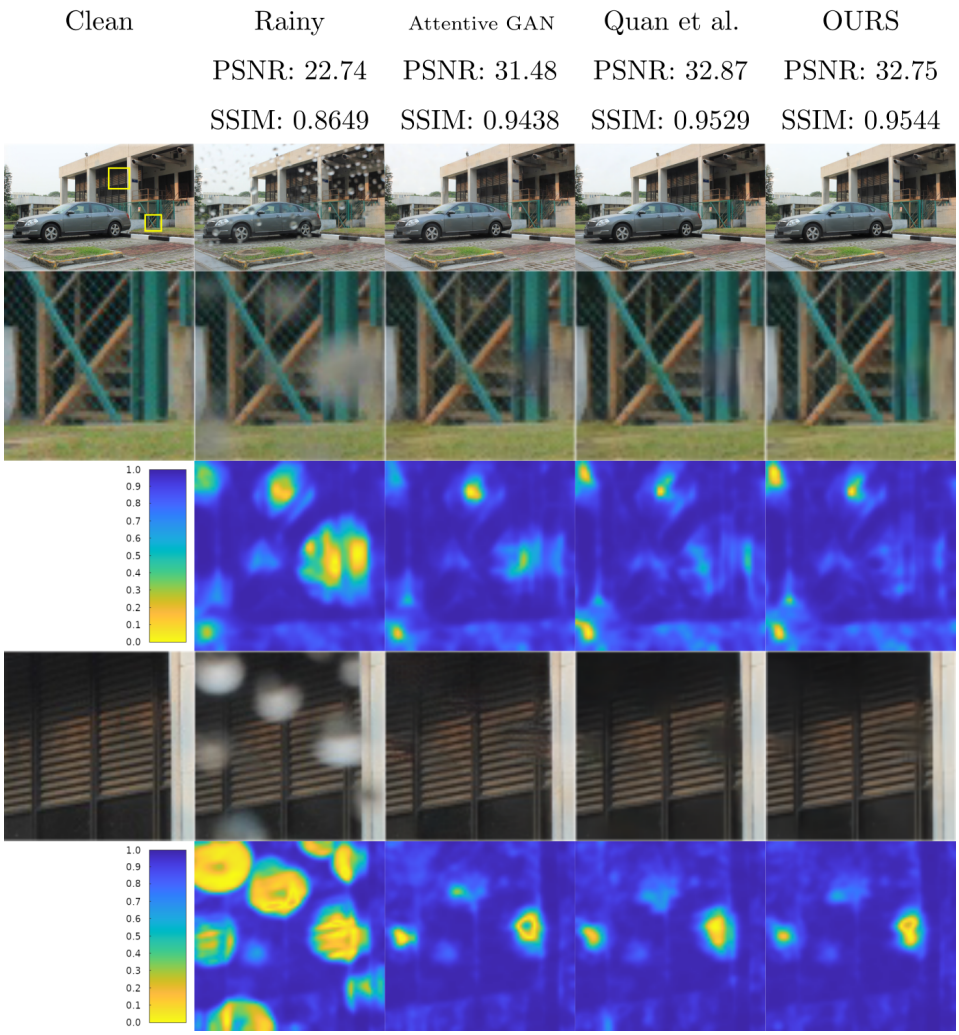


Figure 9.7: Visual comparison of methods for raindrop removal on heavily-textured areas. The proposed model correctly reconstructs some of the complex structures occluded by out-of-focus raindrops. Zoomed crops and the corresponding SSIM maps are reported to facilitate the results interpretation.

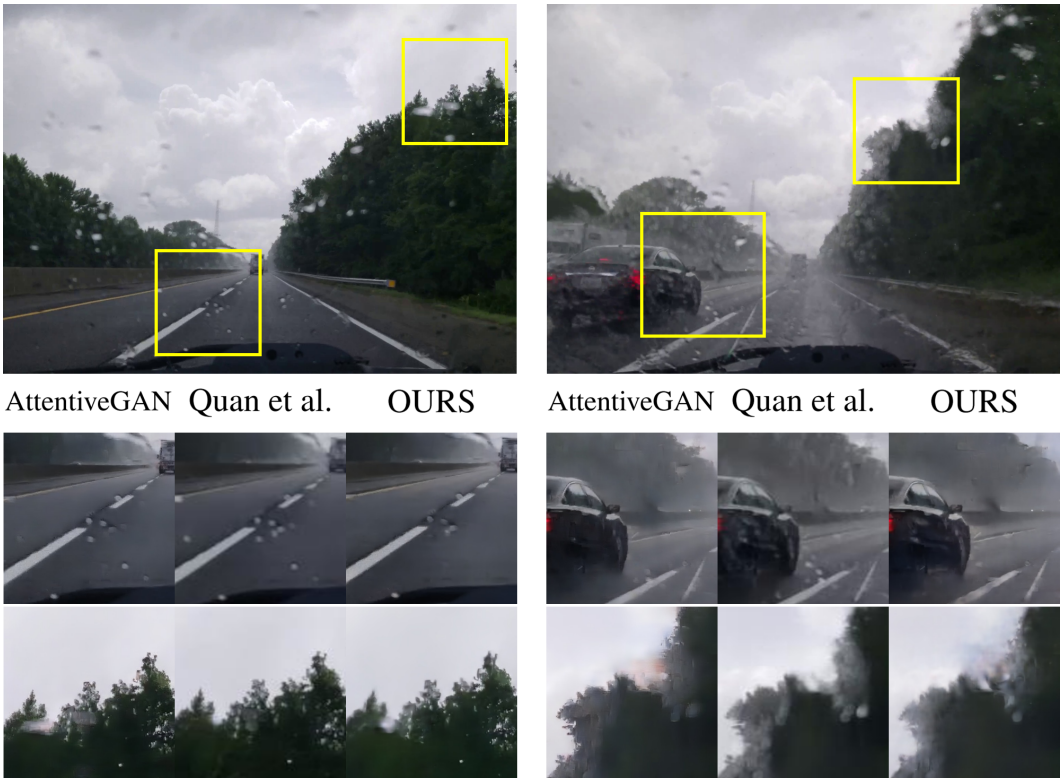


Figure 9.8: Results of rain removal on out-of-dataset images acquired with a car dash camera during a storm. The proposed model is able to restore images from a real-world scenario, removing raindrops and restoring details and structures corrupted by the presence of raindrops. Image credit Eli Christman (https://flic.kr/p/285SQMa).

Chapter 10

Conclusions

This thesis had the objective of identify, design and test new and more robust modules for image processing and restoration that can improve the quality of the acquired images, in particular in critical scenarios such as adverse weather conditions, poor light in the scene etc... To accomplish this objective, an analysis of the digital camera processing pipeline and the possible artifact and defects that can affect images taken by cameras has been done. The result of the analysis led to the division of the image artifacts into two main groups: "in camera generated artifacts" and "external artifacts".

Regarding the first group, four different modules of the camera pipeline have been identified and so new approaches exploiting machine learning have been designed in order to substitute the original camera modules or correct the image processed by those steps in the camera pipeline. For the second group of identified artifacts, new machine learning approaches to post-process the images have been designed, alongside in-depth analysis on the impact of image processing approaches.

In camera generated artifacts

The first module is the Auto White Balancing one. In Chapter 5 a new approach based on the concept of combinational technique exploiting simple multi-layer perceptron is presented. The proposed model has been designed with the purpose of obtaining a good compromise in terms of performance and efficiency, due to the constraint of obtaining a procedure that can potentially be integrated into digital processing pipelines. The proposed approach exploits the assumption over the content of the images made by classical illuminant estimation approaches, such as Gray World, White Patch, Shades of Gray, etc..,

by combining their different estimations into a new, more precise one. The combination procedure is performed using a multi-layer perceptron model, trained by backpropagation procedure using supervised learning techniques. Two versions of the approach have been presented, one focused on performances and another one focused on efficiency. These proposed configuration has proved to give comparable performances with more heavy and complex approaches in the state of the art. The proposed framework for combination, due to its lightweight and efficient nature, has also been extended to temporal color constancy, by the introduction of the use of long short term memory (LSTM) modules. The extension has been tested on benchmark datasets for temporal color constancy, achieving comparable results with respect to the state-of-the-art approaches with high improvements in terms of efficiency.

The second module is described in Chapter 6 and covers the contrast correction and enhancement step of the digital processing pipeline. Here has been presented an approach for contrast enhancement algorithms parameter optimization, based on deep semantic features extraction and user preferences in terms of image aesthetic. Exploiting a deep convolutional neural network, pretrained on a large image classification dataset, and a logistic regressor, user preferences have been modeled on the basis of a dataset proposed by previous work from Adobe [91]. To perform the training of the model two different procedures, one for cleaning the data points used and another one to augment the entire dataset have been designed and used. This model of user preferences on processed images has been adopted as the objective function for the optimization of three different algorithms for single image contrast enhancement. The experimental results show the potentiality of the proposed framework, by improving each algorithm performances in two different configurations of optimization: an optimization based on a training dataset, which gives for each algorithm a set of possible parameters which can be used directly on new images, and a per image optimization procedure, where the parameters are optimized for each new image. These two configurations show also how the optimization procedure can be adopted for the optimization of efficient modules that can be directly integrated into the digital processing pipeline, and as optimization of algorithms in post-processing scenarios.

The third and fourth artifacts considered are the JPEG compression artifacts and the camera noise. In Chapter 7 a new approach based on autoencoders neural network for blind JPEG artifact reduction has been presented. The design of this model, which is intended to be used as a post-processing step for image enhancement, is based on a combination of autoencoder neural

networks which mimic the JPEG compression procedure in a backward way. The proposed approach restores the luminance and chromaticity components in two separate steps, first by enhancing luminance information, then using the restored information as a guide map to perform the chromaticity component restoration. The training of this model has been performed with a dataset of images compressed with different compression factors, to make the model blind and capable to restore any kind of JPEG compressed image. Experimental results on standard datasets show how the proposed approach outperforms the state-of-the-art existing methods and that the proposed solution can generalize to test images that are compressed with compression factors never seen in the training phase. With the main purpose of proving the potential use of such a model on other image restoration tasks, with a very limited amount of changes, the proposed JPEG compression artifact reduction approach has been adapted to camera sensor noise removal. The adapted model has been presented at the NTIRE 2019 workshop challenge, competing with more than 200 different methods and being included in the short list of the best methods. Experiments show how the proposed adaptation can actually compete with ad hoc designed models for noise removal, by simply using a different dataset and with small modifications in the model architecture.

External artifacts

With the term "external artifacts" are referred all of the elements that can affect final image quality that come from the scene and are not directly related to digital processing steps. In this thesis have been considered the artifacts related to images taken in rainy weather conditions. In particular have been considered raindrops, rain streaks and rain-induced haze.

The first approach designed to treat artifacts belonging to this group is described in Chapter 8. In this chapter, a method for the removal of rain streaks and rain-induced haze from single images has been proposed and analyzed. The main focus of this first chapter in this group is the analysis of image processing methods using downstream computer vision tasks : optical character recognition (OCR) and semantic segmentation. To perform the analysis, a Generative Adversarial Network for rain streak removal has been proposed, alongside a dataset augmentation procedure to generate synthetic rain. In the OCR case scenario, the proposed model for rain removal has been adopted to reduce rain in images from street views containing text: the images obtained after the enhancement step have been analyzed using the canonical metrics

for image perceived quality assessment and by analyzing the performances of an OCR algorithm on both rainy and cleaned images. In the second case, images from the Cityscape dataset have been augmented with synthetic rain and then restored with the adopted model for rain reduction. Those images have been processed by a convolutional neural network for semantic segmentation and the obtained results have been analyzed in terms of accuracy and intersection over union. The experimental results show how the application of image processing operation can improve not only the perceived quality of the images but also the usability aspect, giving benefits in terms of performances of downstream tasks.

Finally, in Chapter 9 is proposed a new approach for raindrop removal from camera lenses (or glass surfaces in front of camera) based on autoencoder neural network and image frequency decomposition. The model proposed in this last chapter has been designed in order to process information at different frequency bands, by restoring different levels of the Laplacian pyramid decomposition of the input images. In this work, a new architecture has been proposed, alongside multiple loss function configurations for the training of such a model. Here an analysis of the impact of the different loss functions has been done, and the final proposed method has been tested on standard datasets. The experimental results showed how the proposed approach based on the exploitation of the frequency analysis can obtain comparable or even better results, in comparison to other more complex models which rely on external information to perform the restoration process, such as attention maps or raindrop location maps.

In this thesis, the main focus of the optimization procedures presented were the single modules of the camera pipeline. A future step can be the one of generating digital processing pipelines optimized in order to have different behavior for different kinds of scenarios: the final usage of the images can be considered as the final objective of the processing operation. It is possible to imagine optimizing not only the modules in relation to the single type of artifact, but also the entire processing pipeline with respect to the general use case scenario.

Bibliography

[1] A. Abdelhamed, S. Lin, and M. S. Brown. A high-quality denoising dataset for smartphone cameras. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[2] A. Abdelhamed, R. Timofte, and M. S. Brown. Ntire 2019 challenge on real image denoising: Methods and results. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019.

[3] T. Acharya and W. Metz. Scaling algorithm for efficient color representation/recovery in video, May 22 2001. US Patent 6,236,433.

[4] J. E. Adams Jr, J. F. Hamilton Jr, and J. A. Hamilton. Removing color aliasing artifacts from color digital images, Oct. 12 2004. US Patent 6,804,392.

[5] M. Afifi, K. G. Derpanis, B. Ommer, and M. S. Brown. Learning multi-scale photo exposure correction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9157–9167, 2021.

[6] M. Agarla, S. Bianco, L. Celona, R. Schettini, and M. Tchobanou. An analysis of spectral similarity measures. In *Color and Imaging Conference*, volume 2021, pages 300–305. Society for Imaging Science and Technology, 2021.

[7] E. Agustsson and R. Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, July 2017.

[8] N. Ahmed, T. Natarajan, and K. R. Rao. Discrete cosine transform. *IEEE transactions on Computers*, 100(1):90–93, 1974.

[9] S. Alletto, C. Carlin, L. Rigazio, Y. Ishii, and S. Tsukizawa. Adherent raindrop removal with self-supervised attention maps and spatio-temporal generative adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 0–0, 2019.

[10] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 33(5):898–916, 2010.

[11] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 33(5):898–916, 2011.

[12] I. Bakurov, M. Buzzelli, M. Castelli, R. Schettini, and L. Vanneschi. Parameters optimization of the structural similarity index. In *London Imaging Meeting*, volume 2020, pages 19–23. Society for Imaging Science and Technology, 2020.

[13] I. Bakurov, M. Buzzelli, R. Schettini, M. Castelli, and L. Vanneschi. Structural similarity index (ssim) revisited: A data-driven approach. *Expert Systems with Applications*, 189:116087, 2022.

[14] J. T. Barron. Convolutional color constancy. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 379–387, 2015.

[15] J. T. Barron and Y.-T. Tsai. Fast fourier color constancy. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 886–894, 2017.

[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, 2003.

[17] S. Battiato, A. Bosco, A. Castorina, and G. Messina. Automatic image enhancement by content dependent exposure correction. *EURASIP Journal on Advances in Signal Processing*, 2004(12):1–12, 2004.

[18] S. Battiato, A. R. Bruna, G. Messina, and G. Puglisi. *Image processing for embedded devices*. Bentham Science Publishers, 2010.

[19] D. Bau, B. Zhou, A. Khosla, A. Oliva, and A. Torralba. Network dissection: Quantifying interpretability of deep visual representations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6541–6549, 2017.

[20] B. Bayer. Color imaging array, July 1976. US Patent 971 065.

[21] X. Bi and J. Xing. Multi-scale weighted fusion attentive generative adversarial network for single image de-raining. *IEEE Access*, 8:69838–69848, 2020.

[22] S. Bianco and C. Cusano. Quasi-unsupervised color constancy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12212–12221, 2019.

[23] S. Bianco, G. Ciocca, C. Cusano, and R. Schettini. Improving color constancy using indoor–outdoor image classification. *IEEE Transactions on image processing*, 17(12):2381–2392, 2008.

[24] S. Bianco, F. Gasparini, and R. Schettini. Consensus-based framework for illuminant chromaticity estimation. *Journal of Electronic Imaging*, 17(2):023013, 2008.

[25] S. Bianco, G. Ciocca, C. Cusano, and R. Schettini. Automatic color constancy algorithm selection and combination. *Pattern recognition*, 43 (3):695–705, 2010.

[26] S. Bianco, C. Cusano, and R. Schettini. Color constancy using cnns. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 81–89, 2015.

[27] S. Bianco, L. Celona, and R. Schettini. Robust smile detection using convolutional neural networks. *Journal of Electronic Imaging*, 25(6): 063002, 2016.

[28] S. Bianco, G. Ciocca, and R. Schettini. Combination of video change detection algorithms by genetic programming. *IEEE Transactions on Evolutionary Computation*, 21(6):914–928, 2017.

[29] S. Bianco, C. Cusano, and R. Schettini. Single and multiple illuminant estimation using convolutional neural networks. *IEEE Transactions on Image Processing*, 26(9):4347–4362, 2017.

[30] S. Bianco, M. Buzzelli, and R. Schettini. A unifying representation for pixel-precise distance estimation. *Multimedia Tools and Applications*, 78 (10):13767–13786, 2019.

[31] S. Bianco, M. Buzzelli, G. Ciocca, and R. Schettini. Neural architecture search for image saliency fusion. *Information Fusion*, 57:89–101, 2020.

[32] S. Bianco, M. Buzzelli, G. Ciocca, R. Schettini, M. Tchobanou, and S. Zini. Analysis of biases in automatic white balance datasets. In *Proceedings of the International Colour Association (AIC) Conference 2021. Milan, Italy. AIC*, pages 233–238, 2021.

[33] N. Burningham, Z. Pizlo, and J. Allebach. Encyclopedia of imaging science and technology, chapter image quality metrics, 2002.

[34] P. Burt and E. Adelson. The laplacian pyramid as a compact image code. *IEEE Transactions on communications*, 31(4):532–540, 1983.

[35] M. Buzzelli, J. van de Weijer, and R. Schettini. Learning illuminant estimation from object recognition. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 3234–3238. IEEE, 2018.

[36] M. Buzzelli, S. Bianco, and R. Schettini. Arc: Angle-retaining chromaticity diagram for color constancy error analysis. *JOSA A*, 37(11): 1721–1730, 2020.

[37] M. Buzzelli, S. Zini, S. Bianco, G. Ciocca, R. Schettini, and M. Tchobanou. Analysis of biases in automatic white balance datasets and methods. *Submitted at Color Research and Application*, 2022.

[38] V. Bychkovsky, S. Paris, E. Chan, and F. Durand. Learning photographic global tonal adjustment with a database of input / output image pairs. In *The Twenty-Fourth IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

[39] G. F. C. Campos, S. M. Mastelini, G. J. Aguiar, R. G. Mantovani, L. F. de Melo, and S. Barbon. Machine learning hyperparameter selection for contrast limited adaptive histogram equalization. *EURASIP Journal on Image and Video Processing*, 2019(1):1–18, 2019.

[40] V. C. Cardei and B. Funt. Committee-based color constancy. In *Color and Imaging Conference*, volume 1999, pages 311–313. Society for Imaging Science and Technology, 1999.

[41] L. Cavigelli, P. Hager, and L. Benini. Cas-cnn: A deep convolutional neural network for image compression artifact suppression. In *Neural Networks (IJCNN), 2017 International Joint Conference on*, pages 752–759. IEEE, 2017.

[42] A. Chakrabarti. Color constancy by learning to predict chromaticity from luminance. *arXiv preprint arXiv:1506.02167*, 2015.

[43] A. Chakrabarti, K. Hirakawa, and T. Zickler. Color constancy with spatio-spectral statistics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(8):1509–1519, 2011.

[44] H. Chen, X. He, C. An, and T. Q. Nguyen. Deep wide-activated residual network based joint blocking and color bleeding artifacts reduction for 4: 2: 0 jpeg-compressed images. *IEEE Signal Processing Letters*, 26(1): 79–83, 2018.

[45] D. Cheng, D. K. Prasad, and M. S. Brown. Illuminant estimation for color constancy: why spatial-domain methods work and the role of the color distribution. *JOSA A*, 31(5):1049–1058, 2014.

[46] D. Cheng, B. Price, S. Cohen, and M. S. Brown. Effective learning-based illuminant estimation using simple features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1000–1008, 2015.

[47] C. CIE. Commission internationale de l'eclairage proceedings, 1931. *Cambridge University, Cambridge*, 1932.

[48] D. R. Cok. Signal processing method and apparatus for producing interpolated chrominance values in a sampled color image signal, Feb. 10 1987. US Patent 4,642,678.

[49] S. Corchs and F. Gasparini. A multidistortion database for image quality. In *International Workshop on Computational Color Imaging*, pages 95–104. Springer, 2017.

[50] S. Corchs, F. Gasparini, and R. Schettini. No reference image quality classification for jpeg-distorted images. *Digital Signal Processing*, 30: 86–100, 2014. ISSN 1051-2004. doi: 10.1016/j.dsp.2014.04.003.

[51] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3213–3223, 2016.

[52] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[53] S. Dodge and L. Karam. Understanding how image quality affects deep neural networks. In *2016 eighth international conference on quality of multimedia experience (QoMEX)*, pages 1–6. IEEE, 2016.

[54] C. Dong, Y. Deng, C. Change Loy, and X. Tang. Compression artifacts reduction by a deep convolutional network. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 576–584, 2015.

[55] C. Dong, C. C. Loy, K. He, and X. Tang. Image super-resolution using deep convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, 38(2):295–307, 2016.

[56] H. Dong, J. Pan, L. Xiang, Z. Hu, X. Zhang, F. Wang, and M.-H. Yang. Multi-scale boosted dehazing network with dense feature fusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2157–2167, 2020.

[57] D. Eigen, D. Krishnan, and R. Fergus. Restoring an image taken through a window covered with dirt or rain. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 633–640, 2013.

[58] M. D. Fairchild. *Color appearance models*. John Wiley & Sons, 2013.

[59] R. D. Fiete. *Modeling the Imaging Chain of Digital Cameras*. SPIE, Nov. 2010. doi: 10.1117/3.868276. URL `https://doi.org/10.1117/3.868276`.

[60] G. D. Finlayson, S. D. Hordley, and P. Morovic. Colour constancy using the chromagenic constraint. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 1079–1086. IEEE, 2005.

[61] A. Foi, V. Katkovnik, and K. Egiazarian. Pointwise shape-adaptive dct for high-quality deblocking of compressed color images. In *Signal Processing Conference, 2006 14th European*, pages 1–5. IEEE, 2006.

[62] R. Franzen. Kodak lossless true color image suite. [Online]. Available: `http://r0k.us/graphics/kodak/`, 1999.

[63] C. Fredembach and G. Finlayson. Bright chromagenic algorithm for illuminant estimation. *Journal of Imaging Science and Technology*, 52 (4):40906–1, 2008.

[64] X. Fu, J. Huang, X. Ding, Y. Liao, and J. Paisley. Clearing the skies: A deep network architecture for single-image rain removal. *IEEE Transactions on Image Processing*, 26(6):2944–2956, 2017.

[65] X. Fu, J. Huang, D. Zeng, Y. Huang, X. Ding, and J. Paisley. Removing rain from single images via a deep detail network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3855–3863, 2017.

[66] X. Fu, B. Liang, Y. Huang, X. Ding, and J. Paisley. Lightweight pyramid networks for image deraining. *IEEE Transactions on Neural Networks and Learning Systems*, 2019.

[67] B. Funt, K. Barnard, and L. Martin. Is machine colour constancy good enough? In *European Conference on Computer Vision*, pages 445–459. Springer, 1998.

[68] M. J. Gaboury. Illuminant discriminator with improved boundary conditions, Aug. 6 1991. US Patent 5,037,198.

[69] A. C. Gallagher and E. B. Gindele. Method for adjusting the tone scale of a digital image, Aug. 2001. URL `https://patents.google.com/patent/US6275605B1/en`.

[70] L. Galteri, L. Seidenari, M. Bertini, and A. Del Bimbo. Deep universal generative adversarial compression artifact removal. *IEEE Transactions on Multimedia*, 2019.

[71] K. Garg and S. K. Nayar. Vision and rain. *International Journal of Computer Vision*, 75(1):3–27, 2007.

[72] P. V. Gehler, C. Rother, A. Blake, T. Minka, and T. Sharp. Bayesian color constancy revisited. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008.

[73] A. Gijsenij and T. Gevers. Color constancy using natural image statistics and scene semantics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(4):687–698, 2010.

[74] A. Gijsenij, T. Gevers, and M. P. Lucassen. Perceptual analysis of distance measures for color constancy algorithms. *JOSA A*, 26(10):2243–2256, 2009.

[75] R. C. Gonzales and R. E. Woods. Digital image processing, 2002.

[76] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014. URL `http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf`.

[77] I. Goodfellow, Y. Bengio, and A. Courville. *Deep learning*. MIT press, 2016.

[78] R. M. Goodwin and A. Gallagher. Method and apparatus for area selective exposure adjustment, Oct. 6 1998. US Patent 5,818,975.

[79] S. S. Halder, J.-F. Lalonde, and R. d. Charette. Physics-based rendering for improving robustness to rain. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 10203–10212, 2019.

[80] Z. Hao, S. You, Y. Li, K. Li, and F. Lu. Learning from synthetic photorealistic raindrop for single image raindrop removal. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 0–0, 2019.

[81] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[82] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[83] R. H. Hibbard, K. A. Parulski, and L. J. D'luna. Detail processing method and apparatus providing uniform processing of horizontal and vertical detail components, Oct. 9 1990. US Patent 4,962,419.

[84] K. Hirakawa and T. W. Parks. Adaptive homogeneity-directed demosaicing algorithm. *Ieee transactions on image processing*, 14(3):360–369, 2005.

[85] C.-H. Hsieh, B.-C. Chen, C.-M. Lin, and Q. Zhao. Detail aware contrast enhancement with linear image fusion. In *2010 2nd International Symposium on Aware Computing*, pages 1–5. IEEE, 2010.

[86] Y. Hu, B. Wang, and S. Lin. Fc4: Fully convolutional color constancy with confidence-weighted pooling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4085–4094, 2017.

[87] R. W. G. Hunt. *The reproduction of colour*, volume 4. Wiley Online Library, 1995.

[88] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.

[89] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. *CVPR*, 2017.

[90] J. Jancsary, S. Nowozin, and C. Rother. Loss-specific training of non-parametric image restoration models: A new state of the art. In *European Conference on Computer Vision*, pages 112–125. Springer, 2012.

[91] R. Jaroensri, S. Paris, A. Hertzmann, V. Bychkovsky, and F. Durand. Predicting range of acceptable photographic tonal adjustments. In *2015 IEEE International Conference on Computational Photography (ICCP)*, pages 1–9. IEEE, 2015.

[92] D.-W. Jaw, S.-C. Huang, and S.-Y. Kuo. Desnowgan: An efficient single image snow removal framework using cross-resolution lateral connection and gans. *IEEE Transactions on Circuits and Systems for Video Technology*, 31(4):1342–1350, 2020.

[93] K. Jiang, Z. Wang, P. Yi, C. Chen, Z. Han, T. Lu, B. Huang, and J. Jiang. Decomposition makes better rain removal: An improved attention-guided deraining network. *IEEE Transactions on Circuits and Systems for Video Technology*, 2020.

[94] J. Johnson, A. Alahi, and L. Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European conference on computer vision*, pages 694–711. Springer, 2016.

[95] H. R. V. Joze and M. S. Drew. Exemplar-based color constancy and multiple illumination. *IEEE transactions on pattern analysis and machine intelligence*, 36(5):860–873, 2013.

[96] H. R. V. Joze, M. S. Drew, G. D. Finlayson, and P. A. T. Rey. The role of bright pixels in illumination estimation. In *Color and Imaging Conference*, volume 2012, pages 41–46. Society for Imaging Science and Technology, 2012.

[97] J. E. A. Jr and J. F. H. Jr. Adaptive color plan interpolation in single sensor color electronic camera, Apr. 1996. URL `https://patents.google.com/patent/US5506619A/en?q=Adaptive+color+plan+interpolation+single+sensor+color+electronic+camera&oq=+Adaptive+color+plan+interpolation+in+single+sensor+color+electronic+camera`.

[98] J. F. H. Jr and J. E. A. Jr. Adaptive color plan interpolation in single sensor color electronic camera, May 1997. URL `https://patents.google.com/patent/US5629734A/en`.

[99] L.-W. Kang, C.-W. Lin, and Y.-H. Fu. Automatic single-image-based rain streaks removal via image decomposition. *IEEE Transactions on Image Processing*, 21(4):1742–1755, 2011.

[100] S. B. Kang, A. Kapoor, and D. Lischinski. Personalization of image enhancement. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1799–1806. IEEE, 2010.

[101] S. Khan, B. Phan, R. Salay, and K. Czarnecki. Procsy: Procedural synthetic dataset generation towards influence factor studies of semantic segmentation networks. In *CVPR Workshops*, pages 88–96, 2019.

[102] M. Kim and M. G. Chung. Recursively separated and weighted histogram equalization for brightness preservation and contrast enhancement. *IEEE Transactions on Consumer Electronics*, 54(3):1389–1397, 2008.

[103] Y.-T. Kim. Contrast enhancement using brightness preserving bi-histogram equalization. *IEEE transactions on Consumer Electronics*, 43(1):1–8, 1997.

[104] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[105] T.-H. Le, P.-H. Lin, and S.-C. Huang. Ld-net: An efficient lightweight denoising model based on convolutional neural network. *IEEE Open Journal of the Computer Society*, 1:173–181, 2020.

[106] C.-H. Lee, L.-H. Chen, and W.-K. Wang. Image contrast enhancement using classified virtual exposure image fusion. *IEEE Transactions on Consumer Electronics*, 58(4):1253–1261, 2012.

[107] J.-S. Lee. Digital image smoothing and the sigma filter. *Computer vision, graphics, and image processing*, 24(2):255–269, 1983.

[108] B. Li, W. Xiong, D. Xu, and H. Bao. A supervised combination strategy for illumination chromaticity estimation. *ACM Transactions on Applied Perception (TAP)*, 8(1):1–17, 2010.

[109] B. Li, W. Xiong, W. Hu, and B. Funt. Evaluating combinational illumination estimation methods on real-world images. *IEEE Transactions on Image Processing*, 23(3):1194–1209, 2013.

[110] B. Li, W. Xiong, W. Hu, B. Funt, and J. Xing. Multi-cue illumination estimation via a tree-structured group joint sparse representation. *International Journal of Computer Vision*, 117(1):21–47, 2016.

[111] S. Li, I. B. Araujo, W. Ren, Z. Wang, E. K. Tokuda, R. H. Junior, R. Cesar-Junior, J. Zhang, X. Guo, and X. Cao. Single image deraining: A comprehensive benchmark analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3838–3847, 2019.

[112] B. Lim, S. Son, H. Kim, S. Nah, and K. Mu Lee. Enhanced deep residual networks for single image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 136–144, 2017.

[113] B. R. Lim, R.-H. Park, and S. Kim. High dynamic range for contrast enhancement. *IEEE Transactions on Consumer Electronics*, 52(4):1454–1462, 2006.

[114] P. List, A. Joch, J. Lainema, G. Bjontegaard, and M. Karczewicz. Adaptive deblocking filter. *IEEE transactions on circuits and systems for video technology*, 13(7):614–619, 2003.

[115] P. Liu, H. Zhang, K. Zhang, L. Lin, and W. Zuo. Multi-level wavelet-cnn for image restoration. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2018.

[116] X. Liu, X. Wu, J. Zhou, and D. Zhao. Data-driven soft decoding of compressed images in dual transform-pixel domain. *IEEE Transactions on Image Processing*, 25(4):1649–1659, 2016.

[117] X. Liu, M. Suganuma, Z. Sun, and T. Okatani. Dual residual networks leveraging the potential of paired operations for image restoration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7007–7016, 2019.

[118] Y.-F. Liu, D.-W. Jaw, S.-C. Huang, and J.-N. Hwang. Desnownet: Context-aware deep network for snow removal. *IEEE Transactions on Image Processing*, 27(6):3064–3073, 2018.

[119] R. Lu, A. Gijsenij, T. Gevers, V. Nedović, D. Xu, and J.-M. Geusebroek. Color constancy using 3d scene geometry. In *2009 IEEE 12th International Conference on Computer Vision*, pages 1749–1756. IEEE, 2009.

[120] R. Lukac. *Single-sensor imaging: methods and applications for digital cameras*. CRC Press, 2018.

[121] Y. Luo, Y. Xu, and H. Ji. Removing rain from a single image via discriminative sparse coding. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3397–3405, 2015.

[122] D. Mazzini and R. Schettini. Spatial sampling network for fast scene understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019.

[123] D. Mazzini, M. Buzzelli, D. P. Pauy, and R. Schettini. A cnn architecture for efficient semantic segmentation of street scenes. In *2018 IEEE 8th International Conference on Consumer Electronics-Berlin (ICCE-Berlin)*, pages 1–6. IEEE, 2018.

[124] Y. Mi, S. Yuan, X. Li, and J. Zhou. Dense residual generative adversarial network for rapid rain removal. *IEEE Access*, 9:24848–24858, 2021.

[125] M. Mirza and S. Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[126] A. Mittal, R. Soundararajan, and A. C. Bovik. Making a "completely blind" image quality analyzer. *IEEE Signal processing letters*, 20(3): 209–212, 2012.

[127] A. Mokrane. A new image contrast enhancement technique based on a contrast discrimination model. *CVGIP: Graphical Models and Image Processing*, 54(2):171–180, 1992.

[128] G. Montavon, W. Samek, and K.-R. Müller. Methods for interpreting and understanding deep neural networks. *Digital Signal Processing*, 73: 1–15, 2018.

[129] S. Nah, T. Hyun Kim, and K. Mu Lee. Deep multi-scale convolutional neural network for dynamic scene deblurring. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3883–3891, 2017.

[130] S. W. Oh and S. J. Kim. Approaching the computational color constancy as a classification problem through deep learning. *Pattern Recognition*, 61:405–416, 2017.

[131] J. Peng, Y. Xu, T. Chen, and Y. Huang. Single-image raindrop removal using concurrent channel-spatial attention and long-short skip connections. *Pattern Recognition Letters*, 131:121–127, 2020.

[132] S. M. Pizer, E. P. Amburn, J. D. Austin, R. Cromartie, A. Geselowitz, T. Greer, B. ter Haar Romeny, J. B. Zimmerman, and K. Zuiderveld. Adaptive histogram equalization and its variations. *Computer vision, graphics, and image processing*, 39(3):355–368, 1987.

[133] N. Ponomarenko, L. Jin, O. Ieremeiev, V. Lukin, K. Egiazarian, J. Astola, B. Vozel, K. Chehdi, M. Carli, F. Battisti, et al. Image database tid2013: Peculiarities, results and perspectives. *Signal processing: Image communication*, 30:57–77, 2015.

[134] H. Porav, T. Bruls, and P. Newman. I can see clearly now: Image restoration via de-raining. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 7087–7093. IEEE, 2019.

[135] V. Prinet, D. Lischinski, and M. Werman. Illuminant chromaticity from image sequences. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3320–3327, 2013.

[136] R. Qian, R. T. Tan, W. Yang, J. Su, and J. Liu. Attentive generative adversarial network for raindrop removal from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2482–2491, 2018.

[137] Y. Qian, K. Chen, J. Nikkanen, J.-K. Kamarainen, and J. Matas. Recurrent color constancy. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5458–5466, 2017.

[138] Y. Qian, J.-K. Kamarainen, J. Nikkanen, and J. Matas. On finding gray pixels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8062–8070, 2019.

[139] Y. Qian, J. Käpylä, J.-K. Kämäräinen, S. Koskinen, and J. Matas. A benchmark for burst color constancy. In *European Conference on Computer Vision*, pages 359–375. Springer, 2020.

[140] Y. Quan, S. Deng, Y. Chen, and H. Ji. Deep learning for seeing through window with raindrops. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2463–2471, 2019.

[141] R. Ramanath, W. E. Snyder, Y. Yoo, and M. S. Drew. Color image processing pipeline. *IEEE Signal Processing Magazine*, 22(1):34–43, 2005.

[142] H. C. Reeve and J. S. Lim. Reduction of blocking effects in image coding. *Optical Engineering*, 23(1):230134, 1984.

[143] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[144] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3234–3243, 2016.

[145] M. Rosenblatt. Remarks on some nonparametric estimates of a density function. *The Annals of Mathematical Statistics*, pages 832–837, 1956.

[146] I. Safonov, M. Rychagov, K. Kang, and S. H. Kim. Automatic correction of exposure problems in photo printer. In *2006 IEEE International Symposium on Consumer Electronics*, pages 1–6. IEEE, 2006.

[147] G. Schaefer and M. Stich. Ucid: An uncompressed color image database. In *Storage and Retrieval Methods and Applications for Multimedia 2004*, volume 5307, pages 472–480. International Society for Optics and Photonics, 2003.

[148] R. Schettini, F. Gasparini, S. Corchs, F. Marini, A. Capra, and A. Castorina. Contrast image correction method. *Journal of Electronic Imaging*, 19(2):023005, 2010.

[149] N. Sengee, A. Sengee, and H.-K. Choi. Image contrast enhancement using bi-histogram equalization with neighborhood metrics. *IEEE Transactions on Consumer Electronics*, 56(4):2727–2734, 2010.

[150] M.-W. Shao, L. Li, D.-Y. Meng, and W.-M. Zuo. Uncertainty guided multi-scale attention network for raindrop removal from a single image. *IEEE Transactions on Image Processing*, 30:4828–4839, 2021.

[151] G. Sharma and H. J. Trussell. Digital color imaging. *IEEE transactions on image processing*, 6(7):901–932, 1997.

[152] H. R. Sheikh and A. C. Bovik. Image information and visual quality. *IEEE Transactions on image processing*, 15(2):430–444, 2006.

[153] L. Shi and B. Funt. Re-processed Version of the Gehler Color Constancy Dataset of 568 Images, 2012. `https://www2.cs.sfu.ca/$\sim$colour/data/shi\_gehler/` (Accessed on 30 July 2021).

[154] W. Shi, C. C. Loy, and X. Tang. Deep specialized network for illuminant estimation. In *European conference on computer vision*, pages 371–387. Springer, 2016.

[155] Y.-G. Shin, S. Park, Y.-J. Yeo, M.-J. Yoo, and S.-J. Ko. Unsupervised deep contrast enhancement with power constraint for oled displays. *IEEE Transactions on Image Processing*, 29:2834–2844, 2019.

[156] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint*, arXiv:1409.1556, 2014.

[157] M. Stokes. A standard default color space for the internet-srgb. *http://www. color. org/contrib/sRGB. html*, 1996.

[158] S. K. Subhashdas, Y.-H. Ha, and D.-H. Choi. Multi-class dynamic weight model for combinational color constancy. *Journal of Imaging Science and Technology*, 62(3):30502–1, 2018.

[159] S. K. Subhashdas, Y.-H. Ha, and D.-H. Choi. Hybrid direct combination color constancy algorithm using ensemble of classifier. *Expert Systems with Applications*, 116:410–429, 2019.

[160] F. Sudo and T. Asaida. Image defect correcting circuit for a solid state imager, Sept. 1 1992. US Patent 5,144,446.

[161] S.-H. Sun, S.-P. Fan, and Y.-C. F. Wang. Exploiting image structural similarity for single image rain removal. In *2014 IEEE International Conference on Image Processing (ICIP)*, pages 4482–4486. IEEE, 2014.

[162] Y. Tai, J. Yang, X. Liu, and C. Xu. Memnet: A persistent memory network for image restoration. In *Proceedings of the IEEE international conference on computer vision*, pages 4539–4547, 2017.

[163] Y. Takagi and T. Imaide. White balance adjusting system including a color temperature variation detector for a color image pickup apparatus, Dec. 8 1992. US Patent 5,170,247.

[164] J. Takayama and N. Takizawa. Electronic camera capable of detecting defective pixel, Jan. 27 2004. US Patent 6,683,643.

[165] R. Timofte, E. Agustsson, L. Van Gool, M.-H. Yang, L. Zhang, B. Lim, S. Son, H. Kim, S. Nah, K. M. Lee, et al. Ntire 2017 challenge on single image super-resolution: Methods and results. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2017 IEEE Conference on*, pages 1110–1121. IEEE, 2017.

[166] T. Toizumi, S. Zini, K. Sagi, E. Kaneko, M. Tsukada, and R. Schettini. Artifact-free thin cloud removal using gans. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 3596–3600. IEEE, 2019.

[167] P.-S. Tsai, T. Acharya, and A. K. Ray. Adaptive fuzzy color interpolation. *Journal of Electronic Imaging*, 11(3):293–305, 2002.

[168] A. Valada, J. Vertens, A. Dhall, and W. Burgard. Adapnet: Adaptive semantic segmentation in adverse environmental conditions. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4644–4651. IEEE, 2017.

[169] J. Van De Weijer, T. Gevers, and A. Gijsenij. Edge-based color constancy. *IEEE Transactions on image processing*, 16(9):2207–2214, 2007.

[170] J. Van De Weijer, C. Schmid, and J. Verbeek. Using high-level visual information for color constancy. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8. IEEE, 2007.

[171] G. Verhoeven. Basics of photography for cultural heritage imaging. In *3D recording, documentation and management of cultural heritage*, pages 127–251. Whittles Publishing, 2016.

[172] N. Wada, M. Kazui, and M. Haseyama. Extended joint bilateral filter for the reduction of color bleeding in compressed image and video. *ITE Transactions on Media Technology and Applications*, 3(1):95–106, 2015.

[173] C. Wang and Z. Ye. Brightness preserving histogram equalization with maximum entropy: a variational perspective. *IEEE Transactions on Consumer Electronics*, 51(4):1326–1334, 2005.

[174] C. Wang, J. Zhou, and S. Liu. Adaptive non-local means filter for image deblocking. *Signal Processing: Image Communication*, 28(5):522–530, 2013.

[175] H. Wang, Y. Wu, M. Li, Q. Zhao, and D. Meng. A survey on rain removal from video and single image. *arXiv preprint arXiv:1909.08326*, 2019.

[176] K. Wang. The street view text dataset. `http://www.iapr-tc11.org/mediawiki/index.php?title=The_Street_View_Text_Dataset`, 2011.

[177] N. Wang, B. Funt, C. Lang, and D. Xu. Video-based illumination estimation. In *International Workshop on Computational Color Imaging*, pages 188–198. Springer, 2011.

[178] X. Wang, K. Yu, S. Wu, J. Gu, Y. Liu, C. Dong, Y. Qiao, and C. Change Loy. Esrgan: Enhanced super-resolution generative adversarial networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 63–79, 2018.

[179] Y. Wang, Q. Chen, and B. Zhang. Image enhancement based on equal area dualistic sub-image histogram equalization method. *IEEE transactions on Consumer Electronics*, 45(1):68–75, 1999.

[180] Z. Wang and A. C. Bovik. Modern image quality assessment. *Synthesis Lectures on Image, Video, and Multimedia Processing*, 2(1):1–156, 2006.

[181] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[182] Z. Wang, D. Liu, S. Chang, Q. Ling, Y. Yang, and T. S. Huang. D3: Deep dual-domain based fast restoration of jpeg-compressed images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2764–2772, 2016.

[183] F. Wilcoxon. Individual comparisons by ranking methods. In *Breakthroughs in statistics*, pages 196–202. Springer, 1992.

[184] Q. Wu, W. Zhang, and B. V. Kumar. Raindrop detection and removal using salient visual features. In *2012 19th IEEE International Conference on Image Processing*, pages 941–944. IEEE, 2012.

[185] J. Xie, L. Xu, and E. Chen. Image denoising and inpainting with deep neural networks. In *Advances in neural information processing systems*, pages 341–349, 2012.

[186] K.-F. Yang, S.-B. Gao, and Y.-J. Li. Efficient illuminant estimation for color constancy using grey pixels. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2254–2263, 2015.

[187] Q. Yang, S. Wang, N. Ahuja, and R. Yang. A uniform framework for estimating illumination chromaticity, correspondence, and specular reflection. *IEEE Transactions on Image Processing*, 20(1):53–63, 2010.

[188] W. Yang, R. T. Tan, J. Feng, J. Liu, Z. Guo, and S. Yan. Deep joint rain detection and removal from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1357–1366, 2017.

[189] W. Yang, R. T. Tan, S. Wang, Y. Fang, and J. Liu. Single image deraining: From model-based to data-driven and beyond. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[190] C. Yim and A. C. Bovik. Quality assessment of deblocked images. *IEEE Transactions on Image Processing*, 20(1):88–98, 2011.

[191] J. Yoo, S.-h. Lee, and N. Kwak. Image restoration by estimating frequency distribution of local patches. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6684–6692, 2018.

[192] F. Yu and V. Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015.

[193] K. Zeng, J. Yu, R. Wang, C. Li, and D. Tao. Coupled deep autoencoder for single image super-resolution. *IEEE transactions on cybernetics*, 47 (1):27–37, 2017.

[194] H. Zhang and V. M. Patel. Density-aware single image de-raining using a multi-stream dense network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 695–704, 2018.

[195] H. Zhang, V. Sindagi, and V. M. Patel. Image de-raining using a conditional generative adversarial network. *IEEE transactions on circuits and systems for video technology*, 2019.

[196] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE Transactions on Image Processing*, 26(7):3142–3155, 2017.

[197] X. Zhang, W. Yang, Y. Hu, and J. Liu. Dmcnn: Dual-domain multi-scale convolutional neural network for compression artifacts removal. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 390–394. IEEE, 2018.

[198] B. Zheng, R. Sun, X. Tian, and Y. Chen. S-net: a scalable convolutional neural network for jpeg compression artifact reduction. *Journal of Electronic Imaging*, 27(4):043037, 2018.

[199] S. Zini and M. Buzzelli. On the impact of rain over semantic segmentation of street scenes. In *Workshop on Metrification and Optimization of Input Image Quality in Deep Networks, ICPR 2020*, pages 597–610. Springer, 2021.

[200] S. Zini and M. Buzzelli. Laplacian encoder-decoder network for raindrop removal. *Submitted at Pattern Recognition Letters, Special issue VSI:VETERAN*, 2022.

[201] S. Zini, S. Bianco, and R. Schettini. Cnn-based rain reduction in street view images. In *Proceedings of the 2020 London Imaging Meeting*, pages 78–81, 2020. doi: doi.org/10.2352/issn.2694-118X.2020.LIM-12.

[202] S. Zini, S. Bianco, and R. Schettini. Deep residual autoencoder for blind universal jpeg restoration. *IEEE Access*, 8:63283–63294, 2020.

[203] S. Zini, M. Buzzelli, S. Bianco, and R. Schettini. Cocoa: Combining color constancy algorithms for images and videos. *Submitted at IEEE Transactions on Computational Imaging*, 2022.

[204] S. Zini, M. Buzzelli, S. Bianco, and R. Schettini. A framework for contrast enhancement algorithms optimization. In *Submitted at International Conference on Image Processing*, 2022.

[205] S. Zini, M. Buzzelli, B. Twardowski, and J. van de Weijer. Planckian jitter: enhancing the color quality of self-supervised visual representations. In *Submitted at International Conference on Machine Learning*, 2022.

[206] J. Zwinkels. Light, electromagnetic spectrum. *Encyclopedia of Color Science and Technology*, 8071:1–8, 2015.