COPPE
UFRJ

**Instituto Alberto Luiz Coimbra de
Pós-Graduação e Pesquisa de Engenharia**

EVALUATION OF TONE-MAPPING ALGORITHMS FOR FOCAL-PLANE IMPLEMENTATION

Gustavo Martins da Silva Nunes

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia Elétrica, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia Elétrica.

Orientador: José Gabriel Rodríguez Carneiro Gomes

Rio de Janeiro
Fevereiro de 2018

EVALUATION OF TONE-MAPPING ALGORITHMS FOR FOCAL-PLANE
IMPLEMENTATION

Gustavo Martins da Silva Nunes

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO
ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE
ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE
JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A
OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA
ELÉTRICA.

Examinada por:

_____

Prof. José Gabriel Rodríguez Carneiro Gomes, Ph.D.

_____

Prof. Antonio Petraglia, Ph.D.

_____

Prof. Mylène Christine Queiroz de Farias, Ph.D.

RIO DE JANEIRO, RJ – BRASIL
FEVEREIRO DE 2018

A todos que estão sempre ao
meu lado e me inspiram a ser
uma pessoa melhor.

Agradecimentos

Temo que palavras não sejam suficientes para expressar exatamente toda a minha gratidão e o meu apreço pelas pessoas que citarei nesta seção. Acredito que a melhor forma de demonstrar isso seja no tratamento que busco dar a elas pessoalmente e através de minhas atitudes e ações para com elas, porque, para mim, ações são muito mais importantes do que palavras. Porém, mesmo assim, tentarei usar o breve espaço a mim concedido nessa seção para verbalizar um pouco esse meu sentimento.

Primeiramente, gostaria de agradecer à minha família por todo o suporte e carinho que me deram ao longo de toda a minha vida. Agradeço à minha mãe, Ana Lúcia, e ao meu pai, Nelson, que nunca, jamais, deixaram faltar nada para mim e que sempre estão lá ao meu lado, dando incondicionalmente todo o apoio e carinho durante todos os momentos da minha vida. Se hoje eu sou quem eu sou, se tenho o caráter que tenho, é, com toda a certeza, graças aos meus pais, que servem, e sempre servirão, de exemplo para mim. Agradeço também à minha irmã, Luciana, que, assim como meus pais, sempre está comigo e também serve de grande inspiração para mim, pois é uma pessoa que sempre corre atrás de seus objetivos e seus sonhos, sem depender de ninguém.

Em seguida, gostaria de agradecer ao meu orientador, José Gabriel Gomes, de quem tive o prazer de ser aluno e que também já havia orientado o meu trabalho de conclusão de curso na graduação e aceitou me orientar, novamente, no curso de mestrado. Apesar de suas infinitas reuniões, ele sempre arranjou tempo, entre uma e outra, para me auxiliar e orientar o meu trabalho, levantando discussões interessantes e pontos que eu provavelmente não pensaria em abordar. Sua participação na realização deste trabalho foi decisiva, sempre procurando contribuir de forma positiva, para o enriquecimento deste trabalho. Valorizo muito os seus conselhos e sei que ele sempre buscou ajudar da melhor forma possível.

Agradeço especialmente também à minha coorientadora, Fernanda Duarte, que por questão de alguns meses não está com o seu nome na capa deste trabalho, mas que faço questão de ressaltar aqui o quanto ela foi parte integrante e importante na concepção deste trabalho. Ela me ajudou imensamente no transcorrer deste trabalho (mais até do que ela realmente acha), não só por ter estado sempre disposta a tirar minhas dúvidas, mas também por ter sugerido soluções a problemas que

em direção ao sol nascente), Lucas Tavares (que faz o melhor Mate deste Brasil e, mais importante, sempre o compartilha com os amigos), Michel Morais (que está a um passo de me convencer de que o dia tem mais de 24 horas, dada a quantidade de atividades que consegue realizar simultaneamente), Pedro Bandeira (que me fez reincorporar a gíria "bicho" ao meu vocabulário) e Pedro Fonini (que é capaz de resolver problemas matemáticos antes que eu sequer consiga entendê-los). Obrigado a todos vocês pelos momentos de descontração e de seriedade que tivemos e saibam que vocês me motivam a cada dia a buscar fazer sempre o meu melhor.

Um agradecimento final também a você, leitor, que por um ou outro motivo se interessou por este trabalho e teve paciência de lê-lo, superficialmente ou detalhadamente (ou que, mesmo que não tenha tido paciência nenhuma de lê-lo, ao menos tenha olhado algumas das imagens exibidas neste trabalho e achado pelo menos algumas delas bonitas).

ESTUDO DE ALGORITMOS DE TONE MAPPING PARA IMPLEMENTAÇÃO NO PLANO FOCAL

Gustavo Martins da Silva Nunes

Fevereiro/2018

Cenas no mundo real podem conter uma ampla faixa de valores de diferentes intensidades luminosas. Mostrar a cena original em um aparelho de exibição convencional, tal como um monitor de computador, leva a uma (possivelmente grande) perda de detalhes na cena exibida, uma vez que esses aparelhos são capazes de representar somente uma quantidade limitada de diferentes intensidades luminosas, as quais ocupam uma faixa de valores menor. Para diminuir a perda de detalhes, antes de ser exibida em tais aparelhos, a cena deve ser processada por um algoritmo de tone mapping, o qual mapeia os valores originais de intensidade luminosa em valores que são representáveis pelo aparelho de exibição, acomodando, com isso, a alta faixa dinâmica dos valores de entrada em uma faixa de valores menor. Neste trabalho, uma comparação entre diferentes algoritmos de tone-mapping é apresentada. Mais especificamente, são comparados entre si os desempenhos (referentes a tempos de execução e qualidade geral da imagem processada) da versão digital do operador de tone mapping originalmente proposto por Fernández-Berni *et al.* [11] que é implementado no plano focal da câmera e de diferentes operadores de tone mapping que são originalmente implementados em software. Além disso, uma segunda versão digital do operador no plano focal, a qual simula uma versão modificada da implementação original em hardware, é considerada e seu desempenho é analisado. Essa versão modificada requer um hardware que é menos complexo e ocupa menos espaço que o hardware da implementação original, além de, subjetivamente, manter a qualidade geral da imagem próxima daquela alcançada por operadores digitais. Questões referentes às cores das imagens processadas também são tratadas, especialmente os processamentos que são requeridos pelo operador do plano focal após o tone mapping, de modo a gerar imagens sem distorções de cor.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

EVALUATION OF TONE-MAPPING ALGORITHMS FOR FOCAL-PLANE IMPLEMENTATION

Gustavo Martins da Silva Nunes

February/2018

Advisor: José Gabriel Rodríguez Carneiro Gomes

Department: Electrical Engineering

Scenes in the real world may simultaneously contain very bright and very dark regions, caused by different illumination conditions. These scenes contain a wide range of different light intensity values. Attempting to exhibit a picture of such scene on a conventional display device, such as a computer monitor, leads to (a possibly large) loss of details in the displayed scene, since conventional display devices can only represent a limited amount of different light intensity values, which span a smaller range. To mitigate the loss of details, before it is shown on the display device, the picture of the scene must be processed by a tone-mapping algorithm, which maps the original light intensities into the light intensities representable by the display, thereby accommodating the input high dynamic range of values into a smaller range. In this work, a comparison between different tone-mapping algorithms is presented. More specifically, the performances (regarding processing time and overall quality of the processed image) from a digital version of the tone-mapping operator originally proposed by Fernández-Berni *et al.* [11] that is implemented in the focal plane of the camera and from different tone-mapping operators that are originally implemented in software are compared. Furthermore, a second digital version of the focal-plane operator, which simulates a modified version of the original hardware implementation, is considered and its performance is analyzed. The modified hardware implementation is less complex and requires less space than the original implementation and, subjectively, keeps the overall image quality approximately equal to that achieved by digital operators. Issues regarding colors of the tone-mapped images are also addressed, especially the required processing that must be performed by the focal-plane operator after the tone mapping, in order to yield images without color distortions.

Contents

List of Figures

List of Tables

List of Abbreviations

ADC   Analog-to-Digital Converter

APS   Active Pixel Sensor

CCD   Charge-Coupled Device

CDF   Cumulative Distribution Function

CDS   Correlated Double Sampling

CFA   Color Filter Array

CIE   Comission Internationale d'Éclairage

CMOS   Complementary Metal-Oxide-Semiconductor

CPU   Central Processing Unit

CSF   Contrast Sensitivity Function

DP   Digital Processor

DTMO   Digital Tone-Mapping Operator

FF   Fill Factor

FFT   Fast Fourier Transform

FPTMO   Focal-Plane Tone-Mapping Operator

HDR   High Dynamic Range

HDRI   High Dynamic Range Imaging

JND   Just-Noticeable Difference

LDR   Low Dynamic Range

LED   Light-emitting Diode

MOSFET   Metal-Oxide-Semiconductor Field-Effect Transistor

MSE   Mean Squared Error

NSS   Natural Scene Statistics

NTSC   National Television System Committee

PPD   Pinned Photodiode

PPS   Passive Pixel Sensor

RAWB   Robust Auto White Balance

RGB   Red-Green-Blue

ROI   Region of Interest

SAR   Successive Approximation Register

SI   Système International (d'unités)

SSIM   Structural Similarity

TMQI   Tone-Mapped image Quality Index

TVI   Threshold-versus-Intensity

VDP   Visual Difference Predictor

WB   White Balance

Chapter 1

Introduction

Real-world scenes usually contain many different illumination conditions, provided by multiple light sources. The luminance (amount of light per unit area) emitted by a source can be roughly as low as 10^{-3} cd/m² (starlight) or as high as 10^5 cd/m² (sunlight) [1]. Thus, the luminance values presented in a scene may span several orders of magnitude. Through advanced biological mechanisms, the human visual system adapts itself to the current illumination conditions, accommodating the high dynamic range (HDR) of luminance values of the scene in such a way that details in both darker and brighter regions are observed (the dynamic range is defined as the ratio between the highest and the lowest luminance values of the scene).

Conventional display mediums, such as a regular computer monitors, have a limited amount of luminance levels that can be displayed, which is far less than the amount needed to represent all the luminance intensities of the real scene. These display mediums are capable of reproducing luminance values in an 8-bit range (i.e. up to 256 different levels). Therefore, the luminance values of conventional display devices have a low dynamic range (LDR) and, as a consequence, attempting to exhibit the real scene without any pre-processing in such devices does not maintain the same level of detail observed in the original scene. There exists displays in which the original HDR scenes can be directly displayed without any pre-processing and loss of detail, because they use more than eight bits to represent the luminance values. Such displays are called HDR displays. These, however, are still not widely commercialized, thereby justifying the need of finding solutions that allow the exhibition of these scenes in conventional LDR displays.

High Dynamic Range Imaging (HDRI) is the field of study that addresses different problems associated with high dynamic range scenes, which include capturing, displaying, and storing such scenes. The display problem is also referred to as *tone-mapping* (or *tone-reproduction*) *problem* and consists in finding a function that non-linearly transforms the luminance values of the original scene, in order to fit them into a low dynamic range of display luminance values. These functions are

| (a) Without tone mapping | (b) With tone mapping |

Figure 1.1: Exhibition of an HDR scene with and without the previous application of a tone-mapping operator.

called *tone-mapping operators*. A large number of different operators have been proposed in the literature, some of which are inspired by photographic techniques [2], while others are based on the human eye perception of brightness [3] and others become even more sophisticated as they try to incorporate some aspects of human vision, such as visual acuity, color sensitivity and glare [4]. Figure 1.1 shows the result of displaying an HDR scene in a conventional 8-bit monitor with and without previously applying some tone-mapping operator on it.

Although HDR displays are not commonly used, their invention caused the creation of the *inverse tone-mapping problem*, which consists in mapping LDR scenes into HDR displays, in order to produce scenes with richer detail content, by taking advantage of the increased number of luminance levels that can be represented in such displays. Examples of inverse tone-mapping operators can be found in the literature ([5], [6], [7]).

Most tone-mapping operators are implemented in software, executed in digital hardware and their performances are compared to each other. In this work, these are referred to as *digital* tone-mapping operators. However, there also exist non-digital tone-mapping operators (although not many) that are implemented in hardware and are applied directly to the luminance values registered in the sensors of a camera. Some examples of non-digital tone-mapping operators can be found in the literature ([8], [9] and [10]).

In this work, a detailed discussion of one particular tone-mapping operator implemented in hardware, namely the one originally proposed by Fernández-Berni *et al.* [11], is made. This tone-mapping operator is implemented inside the *focal plane* (i.e. the internal sensing circuit) of the camera, before analog-to-digital conversion, and immediately maps the HDR image to the 8-bit range, as soon as it is acquired. The (grayscale) output image is then readily available for display, without requiring further processing or multiple exposures of the scene. In this work, the tone-mapping operator is further extended to include color information, thereby leading to mo-

difications in the original hardware. The addition of color also requires additional processing steps to be performed, in order to address new problems in the operator that are associated with color, such as color distortions in the resulting images. The hardware modifications and color-related problems (as well as the proposed solutions to them) are also analyzed and discussed.

The goal of this work is divided into two intermediate goals. The first intermediate goal is to prove that this particular focal-plane tone-mapping operator is capable of producing tone-mapped images whose overall quality is comparable to the overall quality of those generated by other digital tone-mapping operators. The resulting images of the focal-plane operator that are used for comparison are obtained from a digital implementation of this operator, which simulates the processing conducted in the focal plane and the constraints imposed by it. Furthermore, digital implementations allow for assessing the performance of different versions of the focal-plane operator, each with a particular hardware configuration and operating principle, without the need of fabricating the hardware of each version. The second intermediate goal is to show that the total processing time required by the focal-plane operator is less than the one demanded by its digital competitors.

1.1   Text Structure

This text is organized as follows. In Chapter 2, some basic concepts are introduced, such as different pixel architectures used in CMOS image sensors and the basic functioning of the human visual system. This chapter also presents the digital tone-mapping operators that were implemented in this work and whose results are later compared to the ones obtained from the focal-plane tone-mapping operator.

In Chapter 3, the focal-plane tone-mapping operator is thoroughly discussed. First, the circuit that determines its operating principle is analyzed and the parameters that affect attributes of the resulting image, such as contrast and brightness, are defined. Then, different implementations of the operator, which include color information, are explored and the problems associated with these parameters are discussed. Lastly, the complexity analysis adopted in this work is explained, which is the guideline used to determine the total processing time required by the focal-plane and the digital tone-mapping operators.

In Chapter 4, the performances of the proposed focal-plane tone-mapping operator and the implemented digital tone-mapping operators are compared by analyzing the quality of the resulting images, with respect to both detail level and colors, and their total processing times, which are obtained from the complexity analysis, are evaluated. Chapter 5 presents the conclusions of this work and future work possibilities.

Chapter 2

Theory

In this chapter, some basic concepts and definitions are introduced. First, the CMOS image sensor history is briefly discussed, presenting the evolution of some of its pixel architectures and culminating with the 4T-APS pixel, which is commonly used in CMOS digital cameras. The concept of quantum efficiency, which is associated with solid-state image sensors, is also presented. Then, a brief introduction of the human visual system is made, discussing some of its particularities and presenting some concepts as luminance, brightness and contrast. Lastly, the tone-mapping problem is defined and some aspects related to tone-mapping operators (more particularly, digital tone-mapping operators) are considered. The chapter ends with a more detailed explanation of the operating principle of each digital tone-mapping operator implemented in this work, whose results are later compared to those obtained from a non-digital tone-mapping operator, which is treated in more detail in Chapter 3. The last subsections, in which each digital tone-mapping operator is explained in more detail, may be skipped, without compromising the understanding of the main discussion topics of this work by the reader.

2.1   4T-APS Pixel

The first image sensors based on semiconductor technology (also called solid-state imagers) date from the early 1960's [12]. These imagers perform light detection through the photoelectric effect, in which photons with sufficient energy striking the semiconductor material generate an electron-hole pair, thus yielding photo-generated free charges. In order to determine the intensity of the incoming light, the photo-generated charges are accumulated in the sensors. The sensors are scanned and their corresponding values are read in an operation called readout, which involves charge transference, charge amplification, and the conversion from charge to voltage values.

The development of different types of solid-state imagers motivated the fabrication of different digital cameras, whose first commercial models emerged in the early

1980's. However, the image sensors using the charge-coupled device (CCD) technology, invented in 1970 by Boyle and Smith [13] yielded a superior image quality than its competitors, thereby popularizing the manufacture of digital cameras with solid-state imagers that use this technology. For many years, CCD image sensors dominated the market of digital cameras, until the 1990's, when solid-state imagers using the complementary metal-oxide-semiconductor (CMOS) technology with in-pixel amplification were developed.

In CMOS imagers, a pixel consists in a light sensor, which is a photodetector that accumulates photo-generated charges, and additional components that perform some processing with the sensor values. The ratio between the photodetector and pixel areas is defined as the fill factor (FF) of a pixel. Several types of photodetectors exist, such as photogates and phototransistors ([12], [14]), but one of the most typically used in CMOS imagers is the pn-photodiode. The pn-photodiode consists in a n-well implant inside a p-substrate, forming a pn-junction and a depletion region around it. Photocarriers (i.e. electron-hole pairs) generated in the depletion region are swept by the electric field in it towards the n-well (for holes) and the p-substrate (for electrons). The holes accumulate in the n-well, thereby changing its electric potential. The amount of photo-generated charges that can be accumulated before the electric potential of the n-well and p-substrate are equalized defines the photodiode full-well capacity and, consequently, the maximum amount of light that can be measured before the sensor saturates. To reduce dark current noise (i.e. the free charge flow in the semiconductor material when no light is striking it) and improve the sensor sensibility, a thin layer of p^+ doped semiconductor material is formed atop the n-well and p-substrate. This version of the pn-photodiode is called *pinned photodiode* (PPD).

As stated in [12], the process of generating and accumulating photocarriers is interpreted as the discharge of a charged capacitor by the generated photocurrent. In circuit schematics, the pn-photodiodes are commonly represented as a current source (the generated photocurrent) in parallel with a capacitor (the pn-junction capacitance), as shown in Figure 2.1. Multiple pixels are arranged in matrix format (i.e. organized in rows and columns), constituting the pixel (or sensor) array of the camera. Additional circuitry is required to access each pixel in the array and to perform the readout operation ([15], [12]). Figure 2.2 shows a simplified schematic of the pixel arrangement in the array and the additional circuitry.

Before the introduction of in-pixel amplification, CMOS image sensors used a simple pixel architecture, which is shown in Figure 2.3 and consisted of a photodiode and a switching transistor M_{RS}. When a given row of the sensor array is selected, the transistors M_{RS} of each pixel of the row are turned on, connecting the photodiodes to the corresponding output column lines, in order to read out their
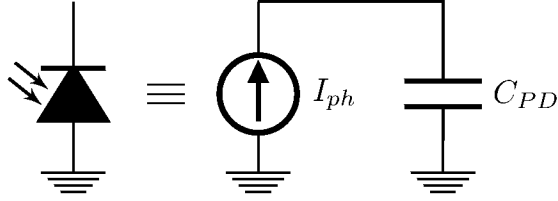
Figure 2.1: Photodiode circuit model.
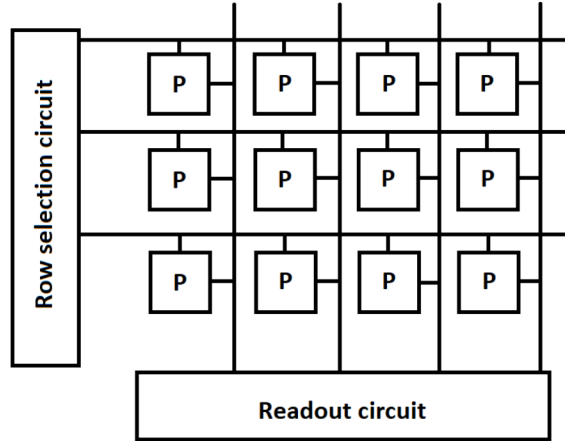
Figure 2.2: Sensor array model.

values. To differentiate from other pixel architectures in which signal amplification is performed, this architecture is called *passive pixel sensor* (PPS). The PPS has a poor performance, especially when compared to CCD image sensors, because it is greatly affected by several sources of noise, such as the one induced by the column output line during the charge transference from the photodiode to the readout circuit [12].

The first pixel architecture of CMOS imagers that employed signal amplification is called *3T active pixel sensor* (3T-APS), which is shown in Figure 2.4. This architecture uses three transistors, namely a reset transistor M_{RST}, a source-follower amplifying transistor M_{SF} and the same row selection transistor M_{RS} from the PPS architecture. The reset transistor is used to assure that no residual photo-generated charges from previous light measurements stay in the photodiode for the next measurement. Thus, before any measurement, the reset transistor is turned on for some time, removing the remaining photocarriers generated from previous measurements. After the reset transistor is turned off, the photodiode starts measuring the current light intensity to which it is exposed. The accumulated photo-generated charges in the photodiode correspond to a voltage value at the gate of the source-follower transistor. The source follower acts as a voltage buffer, isolating the photo-generated charges from any load in the column output line or in the readout circuit, when the row selection transistor is turned on. The voltage gain is slightly less than one, but the input charge is multiplied in the readout circuit, because its overall capacitance
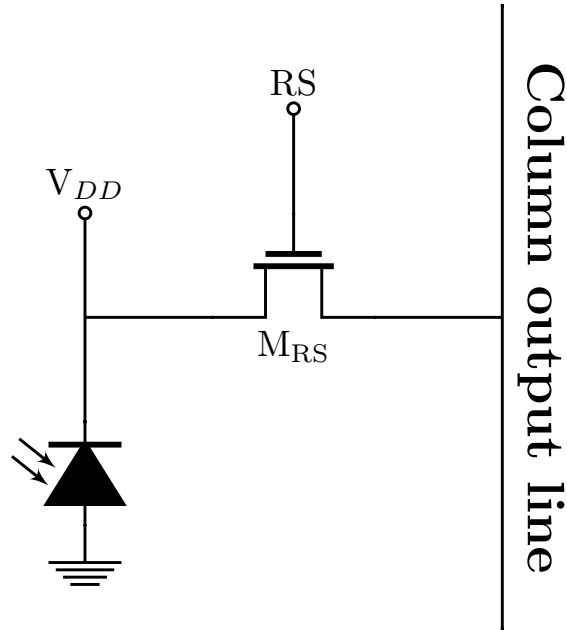
Figure 2.3: Passive pixel sensor (PPS) architecture. RS denotes the row selection signal that switches the M_{RS} transistor on and off.
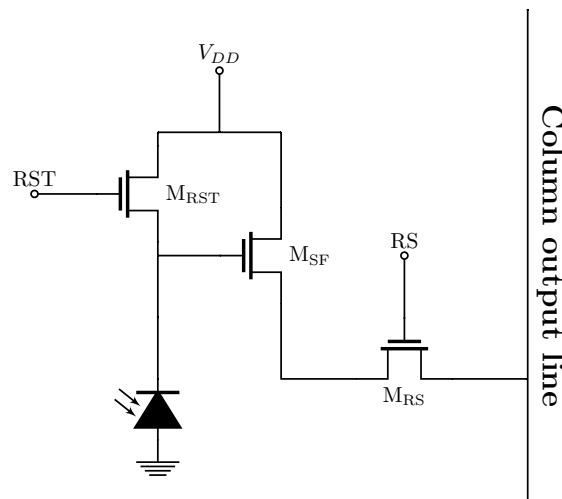
Figure 2.4: 3T active pixel sensor (3T-APS) architecture. RST denotes the reset signal that switches the M_{RST} transistor on and off.

(mainly given by its sample-and-hold capacitor) is higher than the photodiode pn-junction capacitance, thus yielding a charge gain higher than one and amplifying the input signal ([15], [12]). The 3T-APS has an improved signal-to-noise ratio (SNR) when compared to the earlier PPS architecture, but suffers from thermal noise and image lag ([14], [12]), which is caused by an incomplete reset operation (which leaves residual photocarriers from one image to another).

Another active pixel sensor architecture that has improved performance when compared to the 3T-APS is the 4T-APS. This architecture adds one more transistor to the 3T-APS, namely a transfer gate transistor M_{TG}, which isolates the photo-
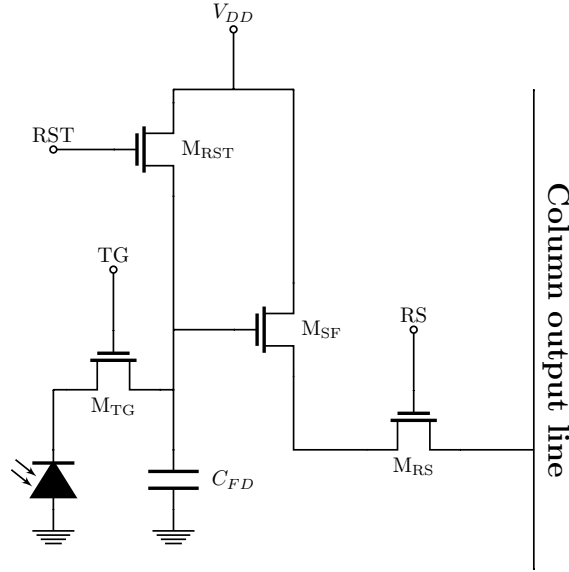
Figure 2.5: 4T active pixel sensor (4T-APS) architecture. ϕ_{TG} denotes the transfer gate signal that switches the M_{TG} transistor on and off.

detection from photointegration (i.e. the conversion from accumulated charges to voltage) operations. The node in which photointegration takes place is called the *floating diffusion node*. After the photodiode measures the incoming light, the reset transistor is turned on, to remove the residual photocarriers from the floating diffusion capacitor. Then, the reset transistor is turned off and the transfer gate is turned on for some time, in order to assure that all photo-generated charges are transferred to the floating diffusion capacitance. The rest of the process is similar to the one of the 3T-APS. In comparison with the 3T-APS architecture, the 4T-APS has less image lag and better SNR, because noise reduction techniques, such as correlated double sampling (CDS), can be used in this architecture [12], but has worse fill factor, since it requires one more transistor.

The 4T-APS architecture produces images whose quality is comparable to those generated by CCD image sensors. Because CMOS image sensors offer some advantages in relation to CCD image sensors, namely lower power consumption, simpler fabrication processes and the possibility of incorporating other on-chip functionalities and processing (leading to the so-called smart CMOS sensors [15], [12]), the 4T-APS architecture allowed CMOS imagers to compete with CCD imagers, making them both share the market of digital cameras nowadays. The 4T-APS architecture serves as basis for an image sensor, originally proposed by Fernández-Berni *et al.* [11], which incorporates the tone mapping operation and is discussed in Chapter 3 of this work.

2.2 Quantum Efficiency

Light is not uniformly absorbed by the semiconductor material. Depending on the material, it is more responsive to some light wavelengths than others. As a consequence, each wavelength of light penetrates differently inside the given semiconductor material, thus generating photocarriers at different depths. The amount of generated photocarriers that contribute as signal charges N_{sig} (i.e. that are actually measured) for a given amount of incident photons N_{ph} per pixel is called *quantum efficiency* η_Q and is defined as [15]:

$$\eta_Q = \frac{N_{sig}}{N_{ph}}. \tag{2.1}$$

The semiconductor material is more sensitive to light wavelengths that generate photocarriers inside the depletion region of the photodetector, because the photo-generated carriers are swept by the electric field in it and accumulate in the potential well. However, photocarriers that are generated too near the surface (by short wavelengths of light) or too deep in the substrate (by long wavelengths of light) may not accumulate in the potential well. In the latter case, the electric field in the substrate is weak and, consequently, generated photocarriers may only reach the depletion region through diffusion, which consists in random movements of the photocarriers. During this process, some of them recombine with the majority carriers of the substrate and thus do not contribute as signal charges. In the former case, photocarriers are trapped and recombined inside surface states [12], thereby also not contributing as signal charges (pinned photodiodes improve the quantum efficiency for shorter wavelengths of light, by avoiding the photocarriers generated by such wavelengths from being trapped in such levels and suffering recombination).

The quantum efficiency curves define the sensitivity of the semiconductor material to every wavelength of light. These curves play an important role in color applications, since they have an impact on the color of the produced images. These curves must be remembered when color distortions are observed in the generated images and solutions are proposed to correct them (the color distortion problem is later discussed in more detail in Chapters 3 and 4).

2.3 Human Visual System

The human vision is a complex mechanism that involves both sensory (i.e. the transduction of light stimuli registered by the eye into nerve electrical impulses) and cognitive (i.e. the processing of such impulses by the brain) aspects. This mechanism motivated a large number of different psychophysical studies, but because of its high

complexity, it is still not fully understood. Therefore, it is difficult to find accurate models of the human visual system that simulate all of its particularities. In this section, some features of the human visual system, as well as visual attributes used to characterize observed scenes, are briefly discussed.

Light is electromagnetic radiation, which is the amount of energy (also referred to as radiant energy) carried by electromagnetic waves, over a range of wavelengths. The radiant energy carried by all possible wavelengths define the *electromagnetic spectrum*. The *visible light spectrum* corresponds to the total radiant energy from the wavelengths between $\lambda \approx 380$ nm (blue light) and $\lambda \approx 740$ nm (red light). Light is a physical quantity measured in energy SI unit Joule (J). At the back of the globe of the eye, there is region called the *retina*, in which a large number of photoreceptors that sense the incoming visible light exists. These photoreceptors are not equally sensitive to all wavelengths of the visible spectrum, which means that light intensities of some wavelengths are better perceived than others. The curve that models the human eye response over the visible light spectrum is called *luminous efficiency curve* $V(\lambda)$ (a plot of this curve can be found in [16]). The quantity of light intensity perceived by the human eye is then the total radiant energy obtained after weighing the incoming light spectrum by the luminous efficiency curve. This physical quantity, called *luminance L*, is defined as:

$$L = \int_{380}^{740} V(\lambda) \cdot R(\lambda) d\lambda, \tag{2.2}$$

where $R(\lambda)$ is the radiant energy spectral distribution of the incoming light per unit area. The human eye photoreceptors are classified in two different types: *rods* and *cones*. The cones are highly concentrated in a small region of the retina, called *fovea*, while the rods are more dispersed through the retina. Each photoreceptor type has its own light sensitivity curve and, as a consequence, each type contributes in determining the luminous efficiency curve of the human eye. The rods are more sensitive to low light levels than the cones, thereby having, in such conditions, more influence on the luminous efficiency curve than the cones. Low illumination conditions are referred to as *scotopic* conditions and, similarly, the vision under these circumstances is called *scotopic* vision. On the contrary, for high light levels, the cones are more sensitive than the rods, thus dictating the human eye response to the incoming light. Vision under high illumination conditions is called *photopic* vision.

The cones are further divided into three subtypes, each one responding differently to each wavelength of light. The subtypes of cones are classified according to the color (or the region) in the visible light spectrum that they are most sensitive to, namely the *blue* (or *S-*) cones, *green* (or *M-*) cones and *red* (or *L-*) cones (*S*, *M* and

L denote short, medium and large wavelengths, respectively). Because the cones measure the overall light intensities in different regions of the visible spectrum, they are responsible for the color vision of the human eye. Colors are visible under photopic conditions, when the cone responses largely contribute to the human eye light sensitivity, and become less distinguishable as light levels decrease, since the cones are less responsive to the light under scotopic conditions.

When the surface of an object is illuminated by a light source, some wavelengths of the incident light are absorbed, others are diffracted and others are reflected. The spectral distribution of the reflected light determines the color of the object. Depending on the reflectance characteristics of the object surface and on the spectral distribution of the incident light, the color of an object should vary under different illuminants. The human eye, however, perceives the same color for an object, regardless of the light source illuminating it, because the human visual system automatically adapts itself to the current illuminant color, thus correcting any color differences caused by the illuminant. This self-adaptation mechanism is called *color constancy*.

While luminance corresponds to an objectively measurable quantity, *brightness* refers to the subjective perception of luminance and is one of the attributes commonly used to describe the overall impression of a scene. As verified by psychophysical studies, such as the one conducted by Stevens and Stevens [17], the brightness impression is not linearly related to the luminance values (e.g. doubling the luminance intensities of a scene does not correspond to the sensation that the overall brightness of the scene has also doubled). Brightness impression of a region also does not depend only on the luminance values of the region, but also on the luminance values of surrounding regions. Figure 2.6 illustrates this statement. The center square has the same intensity in the two images; however it looks brighter when the surrounding square is darker (Figure 2.6a) and it looks darker when the surrounding square is brighter (Figure 2.6b). Finding an accurate model that relates brightness and luminance is thus a difficult task.

Another attribute usually used to characterize the overall impression of a scene is the *contrast*. Like brightness, it is difficult to define the contrast of a scene purely based on its luminance values, because different factors interfere with the perception of contrast. Some of these factors, as well as different definitions of contrast found in the literature, are presented next. However, unless otherwise noted, contrast assessment of images in this work does not refer to any particular definition of contrast, but rather to a more subjective and general understanding of this attribute.

One of the first attempts to model the human eye contrast perception is based on the *Weber-Fachner law*, a relation derived from the results of a series of experi-

(a)	(b)
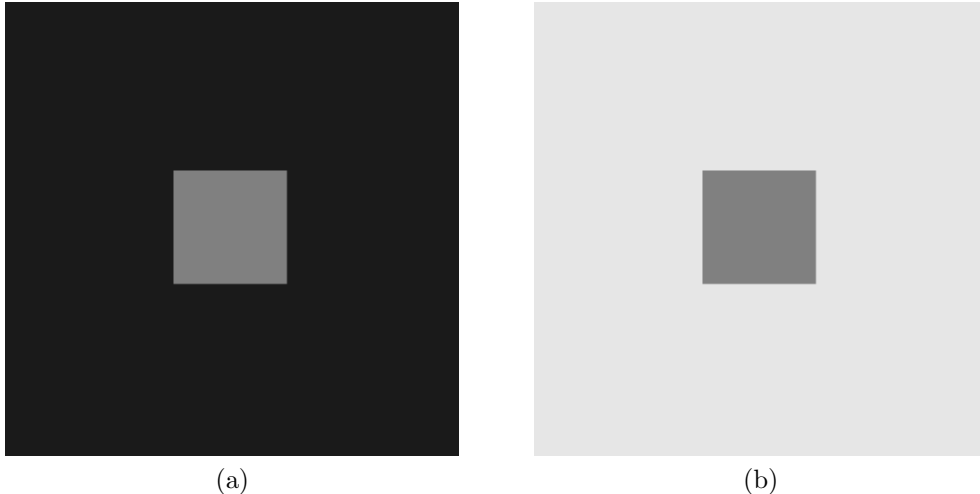
Figure 2.6: Examples of how the luminances of surrounding regions affect the brightness impression of a center region. In (a), all pixels of the surroundings (i.e. the larger square) have the same intensity of 0.1, while in (b), all surrounding pixels have the same intensity of 0.9. In both images, all pixels of the center square have the same intensity of 0.5. The center square looks darker when its surroundings are brighter.

ments conducted by both researchers, in order to measure the human response to physical stimulus of different types. In the context of light perception, this experiment can be reproduced as follows: the subject observes for some time a uniformly illuminated background with luminance L_a. After the observer is adapted to the background luminance, a dot with a certain luminance value, different from the background luminance, is briefly flashed on the background. The dot luminance intensities are varied, until the viewer distinguishes the dot from the background. For each adaptation luminance L_a there is a minimum luminance threshold ΔL, above which the observer barely notices luminance differences. This threshold is commonly referred to as *just-noticeable difference* (JND) threshold. The curves that describe the relation between the adaptation luminances and the just-noticeable luminance differences are called *threshold-versus-intensity* functions (or TVI curves). The Weber-Fechner law, in this context, is defined as:

$$C = \frac{|\Delta L|}{L_a} = k, \tag{2.3}$$

where $\Delta L = L - L_a$ denotes the luminance JND. For a given adaptation luminance, two intensities are perceived as being different if the relative difference between the new and the reference (adaptation) luminance intensities is at least equal to a number k (which changes according to the adaptation luminance intensity considered).

Other definition of contrast is given by Michelson [18], which is based on images of periodic sinusoidal patterns (an example is shown in Figure 2.7). This definition
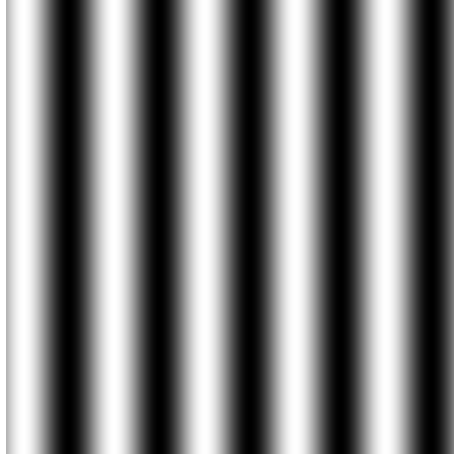
Figure 2.7: An example of a sinusoidal pattern image.

is also referred to as *Michelson contrast*. Its formula is expressed as:

$$C = \frac{L_{max} - L_{min}}{L_{max} + L_{min}}, \tag{2.4}$$

where L_{max} and L_{min} denote the maximum and minimum luminance intensities of the sinusoidal pattern. The two contrast perception models presented assume images of simple patterns. Such images, however, are not representative cases of real-world images, which usually contain more complex structures and patterns. Therefore, for real-world images, these models may not accurately correspond to the overall impression of contrast of the scene. Later studies showed that contrast perception is not only influenced by the luminance intensities, but also by other aspects of the image. One of these aspects, for example, is the spatial variation of the luminance intensities (i.e. spatial frequencies) of the scene, as indicated from the studies of Campbell and Robson [19]. Figure 2.8 illustrates the dependence of contrast perception and spatial frequencies of the image. The perceived height of the illuminated patches changes according to a modulating function, which depends on the spatial frequency, given in cycles per degree (i.e. the number of cycles subtended at the eye in one degree). This dependence motivated the development of contrast perception models that consider how luminance intensities spatially vary in an image. These models are called *Contrast Perception Functions* (CSFs). Some examples of contrast perception models that incorporate this aspect (by calculating local contrasts at different spatial frequency bands in the image) are the ones proposed by Peli [20] and Matkovič *et al.* [21].

Besides spatial variations, contrast perception in a region of the image also depends on the luminance intensities in surrounding regions. An example of this aspect is illustrated in the image called Mach bands, shown in Figure 2.9. All pixels contained in a patch of this image have the same intensity. However, pixels near the

Figure 2.8: Chart illustrating the dependence between contrast perception and spatial frequencies of the image. The frequency of the pattern increases horizontally, from left to right. All illuminated patches, between the dark patches, have their intensities vertically decreasing at the same rate, from the highest intensity (bottom row) to the lowest intensity (top row). The perceived height of the illuminated patches changes according to the spatial frequency (expressed in cycles/degree).
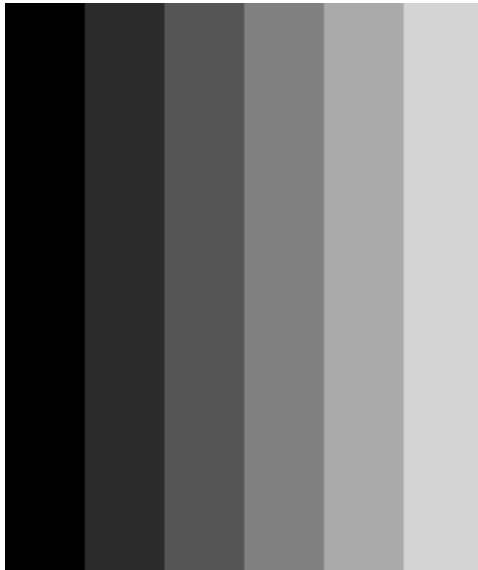
Figure 2.9: Image of the Mach bands, which is used to illustrate how contrast perception near edges depends on the pixel intensities of its adjacent regions. Although all pixels contained within a patch have the same intensities, pixels near the left and right sides of each edge are perceived as darker and brighter, respectively, than pixels more distant from the edges, at each patch.

edges of each patch are perceived as darker and brighter (i.e. with lower and higher intensities, respectively) than other pixels in the corresponding patches, hence, leading to a slightly exaggerated contrast perception in such regions.
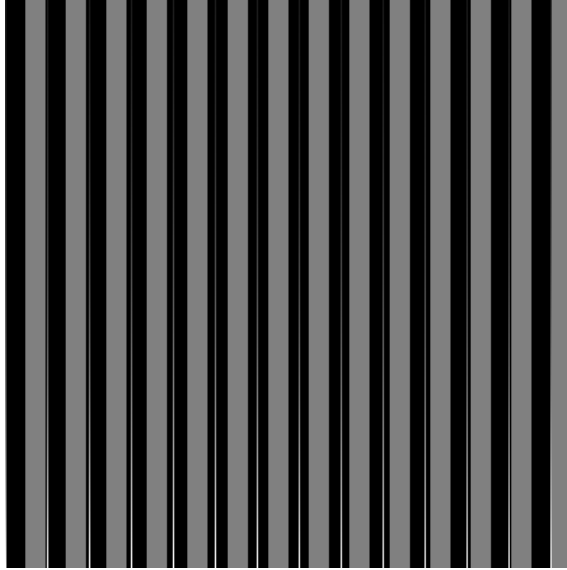
Figure 2.10: Image illustrating the influence of the intensities of surrounding regions in the contrast perception of one region. Pixels in the smaller patches, contained within the black patches, have intensities varying vertically from the highest (bottom row) to the lowest (top row). The height of the smaller patches is perceived as higher when they are located in the center of the black patch, thereby indicating that contrast variations are better perceived in such regions.

Another example of how contrast perception within a region is influenced by the intensities of its surroundings is in the image shown in Figure 2.10. In this image, black and gray patches are disposed in an alternate fashion. All pixels contained within the same patch (i.e. black or gray) have the same intensities. A small patch, along which the pixel intensities vary vertically, is differently positioned inside each black patch. Contrast variations in the small patch are more discriminated when these patches are located in the center of the black patches than when they are located near the edges between a gray and black patch.

Spatially-varying intensities in surrounding regions may also affect the contrast perception of one region. Figure 2.11 illustrates an example of this effect. No contrast variations should be perceived in the inner rectangle, since all of its pixels have the same intensities. However, because the pixel intensities of the surrounding rectangle vary along the horizontal direction, the same perception is induced in the pixels of the inner rectangle: their intensities seem to vary horizontally, but in the opposite direction as the intensity variation in the surrounding rectangle (that is, decreasing intensities from left to right in the inner rectangle, if, in the same direction, the intensities of the surroundings are increasing, and vice-versa). These examples demonstrate the difficulty of finding a complete accurate model of the human eye contrast perception, since it is a complex mechanism that depends on different factors.
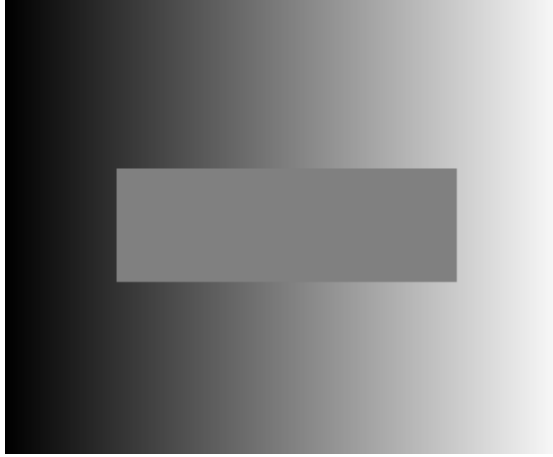
Figure 2.11: Image illustrating contrast perception in one region, when intensities of surrounding regions are spatially-varying. Pixels in the inner rectangle have the same intensity of 0.5. Pixels in the surrounding rectangle vary horizontally, from the minimum intensity of 0.0 (left) to the maximum intensity of 1.0 (right). The varying intensities in the surroundings produce a false impression of contrast variations in the inner rectangle.

2.4   Digital Tone-Mapping Operators

The tone-mapping problem consists in finding a function that maps HDR intensity values of a real scene to LDR values that can be represented on a display device, attending certain conditions. In order to yield mapped values that span a lower dynamic range, the original dynamic range must be somehow compressed. This process of dynamic range compression inevitably leads to some loss of information about the scene, since such information is represented by the original intensity values. The amount and nature of lost information are what define the conditions that guide the development of the tone-mapping functions.

Each tone-mapping operator has its own set of conditions to satisfy. For example, some may focus in preserving details in dark regions, at the cost of losing details in brighter regions (and vice-versa), others may try to simulate imperfections of the human visual system, in order to produce images that more closely resemble the real scene and others may even be concerned with generating visually appealing images. This is in fact discussed in the work of *Čadík et al.* [22], in which the authors categorize the purpose of the tone-mapping reproduction in three approaches: *perceptual*, which aims to simulate the human vision process, *cognitive*, whose objective is to preserve the maximum amount of detail in the scene, and *aesthetical*, which focuses on producing visually pleasant images. The authors argue that any tone-mapping operator realizes a mixture of these approaches. Depending on the established conditions, the mappings vary from linear mappings with saturation zones, defined above or below a threshold value, to (fully) non-linear models that

simulate complex aspects of the human visual system.

The tone-mapping operators can also be divided into two classes: *global* and *local* operators. Global operators are defined as functions that use only the input pixel value and possibly some global information from the scene (like its average pixel value), in order to determine the output pixel values. For these functions, the same input pixel value always results in the same output pixel value. On the other hand, local operators use some *context* information of the input pixels to determine the corresponding output pixel values. Context information refers to any local information that typically changes according to the pixel position in the image, such as the average pixel value of a limited-size neighborhood around the given pixel. For local operators, the same input pixel value may not map to the same output pixel value, since the mapping depends on the surroundings of each pixel. Each approach has its own advantages and limitations, as shown later in this section.

The operators discussed in this section are all purely digital. They are implemented as a *software* which runs on digital hardware, receives as input a digitized HDR image and returns a tone-mapped digital image. As reported in Chapter 1, there also exist tone-mapping operators that are not digital, but rather analog. These are implemented in *hardware* and operate directly on the analog values registered by photosensors, such as photodiodes, of a camera. One tone-mapping operator of this class is later discussed in detail in Chapter 3. Some operators are defined for grayscale images, i.e. they are applied on the luminance channel of the color input image. For a given color (RGB) image, its corresponding luminance channel Y is calculated through the luminance formula applied by the NTSC system (this was a standardized system used for video transmission in North America and part of South America):

$$Y(m,n) = 0.299 \cdot R(m,n) + 0.587 \cdot G(m,n) + 0.114 \cdot B(m,n), \qquad (2.5)$$

where R, G and B denote the red, green and blue color channels of the image, respectively, and (m,n) denotes the pixel position in the image. Other slightly different formulations for the luminance also exist, but, throughout this work, luminance is assumed to be defined by this formula. Some digital operators further require that the pixels of the input image luminance channel, which are in dimensionless normalized units, represent absolute luminance values that are measured in the real scene. In such cases, the pixels are pre-multiplied by a constant factor that converts the input grayscale dimensionless values to absolute luminance values.

For the digital operators that are defined for grayscale images, some extra processing must be performed to incorporate color information in the tone-mapped images. One possibility is to convert the original color image, which is in the RGB

17

color space, to another color space that separates the image luminance information Y from the color information (also referred to as *chrominance* components), perform the tone-mapping transformation on the luminance component Y and apply the corresponding conversion from the luminance-chrominance color space to the RGB color space using the original chrominance components and the tone-mapped luminance component Y_{TM}, thus resulting in a tone-mapped image in the RGB color space. Another possibility is the one defined in [23]. The author argues that minimum color shifts occur if the transformed image preserves the color channel ratios from the original image. By denoting the original and the tone-mapped color channel pixel values as (R, G, B) and (R_{TM}, G_{TM}, B_{TM}), respectively:

$$\frac{R_{TM}}{R} = \frac{G_{TM}}{G} = \frac{B_{TM}}{B}. \tag{2.6}$$

In order to obtain the red tone-mapped color channel value of the pixel at position (m, n) in the image, the corresponding tone-mapped luminance value $Y_{TM}(m, n)$, given by Equation (2.5), is initially divided by $R_{TM}(m, n)$. By starting to solve for $R_{TM}(m, n)$, one finds:

$$R_{TM}(m, n) = \frac{Y_{TM}(m, n)}{0.299 + 0.587 \cdot \frac{G_{TM}(m,n)}{R_{TM}(m,n)} + 0.114 \cdot \frac{B_{TM}(m,n)}{R_{TM}(m,n)}}. \tag{2.7}$$

From Equation (2.6), $\frac{G_{TM}(m,n)}{R_{TM}(m,n)} = \frac{G(m,n)}{R(m,n)}$ and $\frac{B_{TM}(m,n)}{R_{TM}(m,n)} = \frac{B(m,n)}{R(m,n)}$. By making this substitution in Equation (2.7):

$$\begin{aligned} R_{TM}(m, n) &= \frac{Y_{TM}(m, n)}{0.299 + 0.587 \cdot \frac{G(m,n)}{R(m,n)} + 0.114 \cdot \frac{B(m,n)}{R(m,n)}} \\ &= \frac{Y_{TM}(m, n)}{0.299 \cdot R(m, n) + 0.587 \cdot G(m, n) + 0.114 \cdot B(m, n)} \cdot R(m, n). \end{aligned} \tag{2.8}$$

The denominator from Equation (2.8) corresponds to the original luminance value $Y(m, n)$. Therefore, the red tone-mapped color channel is obtained by dividing the tone-mapped luminance channel Y_{TM} by the original luminance channel Y and then multiplying the original red color channel by the resulting division (this operation is performed at each pixel of the image):

$$R_{TM}(m, n) = \frac{Y_{TM}(m, n)}{Y(m, n)} \cdot R(m, n). \tag{2.9}$$

The same procedure applies for obtaining the green and blue tone-mapped color channels. Unless otherwise stated, this is the approach used to recover color information for operators that are only defined for grayscale images. Next, the digital tone-mapping operators implemented in this work are discussed in more detail.

18

2.4.1 Ward Contrast-Based

A tone-mapping operator proposed by Ward [24] consists in a linear relation between the real scene luminance values and the display luminance values, which incorporates some characteristics of the human visual response. Previous attempts at displaying real scenes on a monitor included simple linear mappings, such as mapping the average luminance of the scene to half the maximum display brightness. These linear mappings, however, did not take any advantage of how the human eye response changes according to different illumination conditions. One of the first non-linear tone-mapping operators to do so was the one proposed by Miller *et al.* [16], which aimed at preserving the brightness ratios between parts of the real scene on the displayed scene, based on the human eye perception of brightness. Claiming that the human eye is more sensitive to contrast than brightness, Ward developed its operator based on the assumption that contrast visibility of the real scene should be roughly preserved on the displayed scene, rather than the brightness ratios (that is, discernible contrasts in the real scene should also be discernible in the displayed scene). The real scene luminance value $L_w(m,n)$ and the display luminance value $L_d(m,n)$ are related as follows:

$$L_d(m,n) = f \cdot L_w(m,n). \tag{2.10}$$

The constant factor f is calibrated based on the psychophysical data obtained from experiments to measure the human eye capability of discerning contrast under different illumination conditions. Because the viewing conditions of the real scene and the displayed scene are usually different, the adaptation luminance for the observer eye in both conditions are also different. Based on such measured psychophysical data, the constant factor f can be expressed as a function of the just-noticeable luminance differences associated with the display and the real scene adaptation luminances:

$$f = \frac{\Delta L(L_{wa})}{\Delta L(L_{da})} = \left[\frac{1.219 + L_{da}^{0.4}}{1.219 + L_{wa}^{0.4}} \right]^{2.5}, \tag{2.11}$$

where L_{da} and L_{wa} denote the display and the real scene adaptation luminance, respectively. The real scene adaptation luminance is estimated as the exponential of the average log of the real scene luminance values:

$$L_{wa} = \exp \left(\frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} \log \left[L_w(m,n) + \epsilon \right] \right), \tag{2.12}$$

where $\exp(.)$ denotes the natural exponential function and ϵ is a small constant to avoid indetermination in the log function for $L_w(m,n) = 0$. The display adaptation

(a) Courtyard. (b) Fire Extinguisher.

Figure 2.12: Images resulting from the Ward contrast-based scale factor operator. Details in dark and/or bright regions are lost, such as the inner corridors and roof tiles in (a) and the grass outdoors in (b).

luminance is estimated at half the maximum luminance of the display device, which is denoted as L_{dmax} and typically ranges between 30 cd/m^2 and 100 cd/m^2 [16]. So far, all the computations for this operator are performed on absolute luminance values, which are expressed in the SI unit candela per square meter (cd/m^2). In order to show the real scene on a display, the display luminance values L_d must be converted to display input values p, which are dimensionless (and usually expected to be in the [0,1] range). This is achieved through the normalization of the constant factor f by the maximum display luminance L_{dmax}. Applying this normalization factor to Equation (2.11) and then substituting the resulting expression in Equation (2.10) leads to the final expression for the tone-mapping operator:

$$Y_{TM}(m,n) = \frac{1}{L_{dmax}} \cdot \left[\frac{1.219 + \left(\frac{L_{dmax}}{2}\right)^{0.4}}{1.219 + L_{wa}^{0.4}} \right]^{2.5} \cdot L_w(m,n). \qquad (2.13)$$

Since the operation is linear, the dynamic range of the real scene luminance values is not changed. This operator, like other linear operators, fails to show details at either very bright or very dark regions of HDR scenes on a conventional display device. This is because display devices can only represent a limited number of luminance values, typically in the 8-bit range. The dynamic range of the display luminance values is usually inferior to the dynamic range of the luminance values in such scenes. Pixel values that lie outside the displayable range are clamped to the extreme values of the display dynamic range and, therefore, details in very dark or very bright regions of the scene are lost. This is illustrated in Figure 2.12, which shows the images resulting from this operator.

2.4.2 Ferwerda

A tone-mapping operator similar to the previously discussed Ward operator was proposed by Ferwerda *et al.* [25]. This operator attends to the fact that the rod and cone system of the human eye respond differently to the light stimulus and they are both considered in determining the human eye luminous efficiency curve. Therefore, instead of modelling the human eye luminance sensitivity with one single function and calculating one global scale factor f, Ferwerda operator uses two functions to model such sensitivity and each one corresponds to the cone and rod responses to the input light stimulus. In this operator, two scale factors are then calculated: one associated with the cone system f_c and other, with the rod system f_r. Since the cone system is responsible for the color sensitivity of the human eye, the cone scale factor f_c is applied directly on the color channels of the input image. The rod system is insensitive to the color variations and, as such, the rod scale factor f_r is applied on the luminance values of the input image, which is achromatic. The operator assumes that the input luminance values are calibrated in the SI unit cd/m². Because of this, the input image must be multiplied by a constant calibration factor L_{max} that converts the dimensionless display input values to luminance values before the tone-mapping operation is applied. The tone-mapping expression for this operator is given by [16]:

$$
\begin{aligned}
R_{TM}(m,n) &= \frac{L_{wmax}}{L_{dmax}} \cdot [f_c \cdot R(m,n) + f_r \cdot L_w(m,n)] \\
G_{TM}(m,n) &= \frac{L_{wmax}}{L_{dmax}} \cdot [f_c \cdot G(m,n) + f_r \cdot L_w(m,n)] \\
B_{TM}(m,n) &= \frac{L_{wmax}}{L_{dmax}} \cdot [f_c \cdot B(m,n) + f_r \cdot L_w(m,n)] ,
\end{aligned}
\tag{2.14}
$$

where R, G and B represent the red, green and blue color channels of the input image, R_{TM}, G_{TM} and B_{TM} represent the red, green and blue color channels of the resulting tone-mapped image, L_w is original image luminance channel and L_{dmax} is the maximum luminance of the display. Like in the Ward contrast operator, both scale factors f_c and f_r are determined by the ratio between the TVI functions of the rod and cone systems, evaluated at the adaptation luminance of the real scene (L_{wa}) and at the adaptation luminance of the display (L_{da}):

$$
\begin{aligned}
f_c &= \frac{\Delta L(L_{wa})}{\Delta L(L_{da})}\bigg|_{photopic} \\
f_r &= \frac{\Delta L(L_{wa})}{\Delta L(L_{da})}\bigg|_{scotopic}
\end{aligned}
\tag{2.15}
$$

The adaptation luminance of the real scene L_{wa} and of the display L_{da} are set to half the maximum luminance of the scene L_{wmax} and of the display L_{dmax}, re-
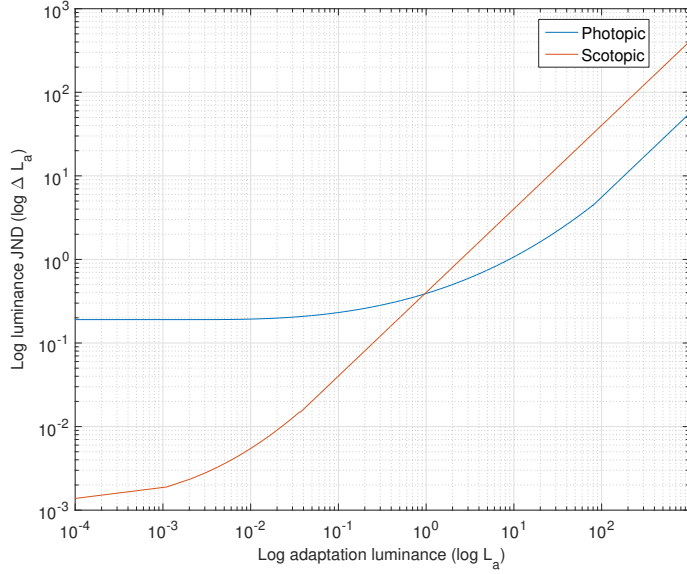
Figure 2.13: TVI functions for the cone and rod systems.

spectively. Different TVI functions are used in the calculation of each scale factor, since these functions represent the different responses of the rod and cone systems. Figure 2.13 shows the TVI functions $\Delta L(L_a)|_{photopic}$ and $\Delta L(L_a)|_{scotopic}$ used for the cone and rod system, respectively. The corresponding expressions for each function are defined in Equations (2.16) and (2.17):

$$
\log_{10} \Delta L(L_a)|_{photopic} = \begin{cases} -0.72, & \text{if } \log_{10} L_a \leq -2.16 \\ \log_{10} L_a - 1.255, & \text{if } \log_{10} L_a \geq 1.9 \\ (0.249 \cdot \log_{10} L_a + 0.65)^{2.7} - 0.72, & \text{otherwise.} \end{cases} \tag{2.16}
$$

$$
\log_{10} \Delta L(L_a)|_{scotopic} = \begin{cases} -2.86, & \text{if } \log_{10} L_a \leq -3.94 \\ \log_{10} L_a - 0.395, & \text{if } \log_{10} L_a \geq 1.44 \\ (0.405 \cdot \log_{10} L_a + 1.6)^{2.18} - 2.86, & \text{otherwise.} \end{cases} \tag{2.17}
$$

Other than the different responses of the rod and cone systems, the operator also includes more complex aspects of the human visual system, such as visual acuity (i.e. the capability of the visual system to resolve spatial details) and light and dark adaptation of the visual system over time. The implementation of this tone-mapping operator in this work does not include these additional features of the human visual system. Figure 2.14 shows the images resulting from the Ferwerda operator, with different pre-calibration factors L_{max} previously applied to the input
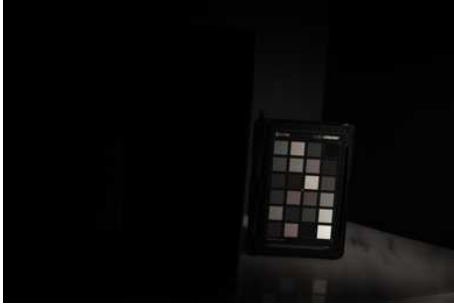
(a) $L_{max} = 1$.   (b) $L_{max} = 100$.

Figure 2.14: Images resulting from the Ferwerda operator, with the input image pre-calibrated with different constant factors L_{max}, in order to simulate different illumination conditions. For low luminance levels, the colors are less evident, because the human visual system response is mostly determined by the rod system, which is insensitive to color variations. For high luminance levels, the cones dominate the human vision response to light and, because of this, colors are distinguishable.

image, in order to simulate the same scene under low and high illumination. Under photopic conditions, $f_c > f_r$ and colors are more distinguishable in the resulting images. Under scotopic conditions, $f_c < f_r$ and colors of the image are less visible (for very low illumination conditions, $f_r >> f_c$ and the image is mostly achromatic, as shown in Figure 2.14).

The Ferwerda operator is also a linear operator and, therefore, the same observations made for the Ward contrast operator are valid for this operator. Tone-mapped HDR scenes from this operator displayed on a conventional monitor do not exhibit details at either very bright or very dark regions of the image, as illustrated in Figure 2.14.

2.4.3 Logarithm & Exponential

Tone-mapping operators can also be simple mathematical functions. Non-linear functions are usually more suitable for the tone-mapping task than generic linear functions, because they change the original relations between the pixel values of the input image without necessarily clamping a large range of pixel values to one value. It is then possible to effectively reduce the original HDR luminance values of the captured scene to LDR luminance values that are supported by the display device, while preserving details in dark and bright regions. A non-linear function that can be used as a tone-mapping operator is, for example, the logarithm function. The Logarithm tone-mapping operator implemented in this work is expressed as follows:

$$L_d(m, n) = \log_2 L(m, n), \tag{2.18}$$

where L denotes the luminance channel of the input image. In order to assure that

(a) Courtyard.

(b) Fire Extinguisher.

Figure 2.15: Images resulting from the Logarithm operator. Details are observable in dark and bright regions in both images.

the resulting tone-mapped luminance values L_d are also within the $[0,1]$ displayable range, a linear transformation is performed, thereby yielding the final tone-mapped luminance channel Y_{TM}:

$$Y_{TM}(m,n) = \frac{L_d(m,n) - L_{dmin}}{L_{dmax} - L_{dmin}}, \tag{2.19}$$

where L_{dmin} and L_{dmax} respectively denote the minimum and maximum pixel values obtained after applying the Logarithm operator to the input image luminance channel. Figure 2.15 shows some images resulting from the Logarithm operator. The tone-mapped scene contains visible details in bright and dark regions, which was not accomplished by the linear tone-mapping operators.

Another non-linear function that can be used as a tone-mapping operator is the exponential function e^x. The Exponential tone-mapping operator of this work is based on the implementation from [16] and is expressed as follows:

$$Y_{TM}(m,n) = 1 - e^{L(m,n)/L_{norm}}, \tag{2.20}$$

where L represents the luminance channel of the input image and L_{norm} is an arbitrary luminance normalization value. Two possible values considered for L_{norm} are the maximum luminance value L_{max} and the average luminance value L_{avg} of the input image. For $L_{norm} = L_{avg}$, the tone-mapped scene tends to be brighter, because the average luminance value of the input scene is mapped to a slightly higher value ($1 - e^{-1} \approx 0.63$) than the middle of the displayable values dynamic range (0.5). For $L_{norm} = L_{max}$, the maximum luminance value of the input image is not directly mapped to the highest displayable value and, thus, the tone-mapped images obtained from this choice of mapping look darker. Figure 2.16 illustrates the images resulting from the different choices of L_{norm}.

(a) $L_{norm} = L_{max}$.

(b) $L_{norm} = L_{avg}$.

(c) $L_{norm} = L_{max}$.

(d) $L_{norm} = L_{avg}$.

Figure 2.16: Images resulting from the Exponential operator, using different normalization factors L_{norm}.

2.4.4 Drago

Drago *et al.* proposed a tone-mapping operator which also consists in a logarithmic mapping [26]. However, in this operator, the base of the logarithmic function used to perform such mapping changes according to the pixel luminance value, automatically adapting itself to each pixel. The base varies between the values of 2 and 10, as established by the authors through empirical observations. The relation that defines the tone-mapping operator is given by:

$$L_d(m, n) = \frac{\log_{base}(1 + L_w(m, n))}{\log_{10}(1 + L_{wmax})}, \tag{2.21}$$

where L_d denotes the display luminances, L_w represents the real scene (original) luminances and L_{wmax} is the maximum luminance of the real scene (this operator assumes that the luminances are specified in cd/m² unit). In order to assure a smooth interpolation between the different bases and avoid undesired artifacts, the so-called *bias function*, developed by Perlin and Hoffert, is used to calculate the base. This function is defined as $bias_p(t) = t^{log(p)/log(0.5)}$, where p is a user-defined parameter, whose value is between 0 and 1. Using the bias function with $t = \frac{L_w(m,n)}{L_{wmax}}$ and an arbitrary value for the p parameter (within the [0,1] range), the base of the logarithmic function is determined according to the following relation:

$$base = 2 + 8 \cdot \text{bias}_p(t) = 2 + 8 \cdot \left(\frac{L_w(m,n)}{L_{wmax}} \right)^{\log_{10}(p)/\log_{10}(0.5)}. \qquad (2.22)$$

In Equation (2.22), the constants 2 and 8 bound the calculated base to be within the [2,10] value range. Since $\log_{base}(x) = \log_{10}(x)/\log_{10}(base)$, the final version of this operator can be expressed as:

$$
\begin{aligned}
L_d(m,n) &= \frac{L_{dmax}/100}{\log_{10}(1+L_{wmax})} \cdot \frac{\log_{10}(1+L_w(m,n))}{\log_{10} base} \\
&= \frac{L_{dmax}/100}{\log_{10}(1+L_{wmax})} \cdot \frac{\log_{10}(1+L_w(m,n))}{\log_{10}\left[2 + 8 \cdot \left(\frac{L_w(m,n)}{L_{wmax}}\right)^{\log_{10}(p)/\log_{10}(0.5)}\right]},
\end{aligned}
\qquad (2.23)
$$

where L_{dmax} is the maximum display luminance. The operator tends to compress more the brighter areas of the image than the darker ones, because pixels with higher values yield higher values for the base. On the other hand, darker areas are less compressed, which tends to make details more visible in such areas. How much compression occurs in the bright areas and how much detail is visible in the dark areas of the image is determined by the parameter p. This parameter also affects the overall contrast of the tone-mapped scene. Figure 2.17 shows the tone-mapping operator curves for different values of p. Typically, values slightly higher than 0.7 are chosen for this parameter, since they correspond to monotonically increasing curves that yield pixel values within the displayable range (and, therefore, no clamping is required). As reported by the authors, $p = 0.85$ produces well-balanced images. Figure 2.18 shows the effect of varying the parameter p in the resulting images.

2.4.5 Tumblin-Rushmeier

Tumblin and Rushmeier were among the first to address the tone reproduction problem in the context of computer graphics. This problem was already known in other areas, such as photography, printing and television. In their work [3], the authors present the solution used by television and film systems, in order to deal with the tone reproduction problem. Then, taking inspiration on aspects of the human visual system luminance perception, they proposed a new solution for this problem in the computer graphics area, namely a tone-mapping operator. Their operator is based on the assumption that the brightness values of the real-world scene Q_w and the display Q_d should be equal. The brightness function used in the operator is defined as [16]:

$$Q(m,n) = C_0 \cdot \left(\frac{L(m,n)}{L_a} \right)^{\gamma(L_a)}, \qquad (2.24)$$
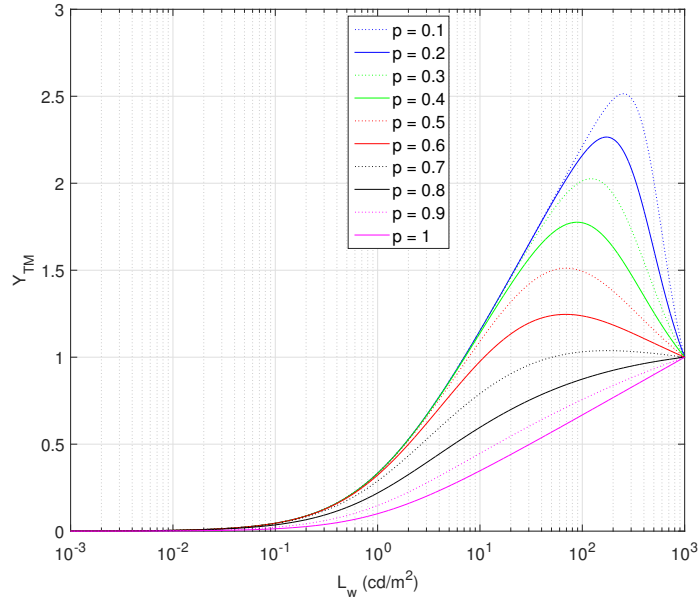
Figure 2.17: Tone-mapping curves of the Drago operator from Equation (2.23) for different values of the parameter p (assuming $L_{dmax} = 100$ cd/m^2).

(a) $p = 0.75$. (b) $p = 0.85$. (c) $p = 0.95$.

(d) $p = 0.75$. (e) $p = 0.85$. (f) $p = 0.95$.
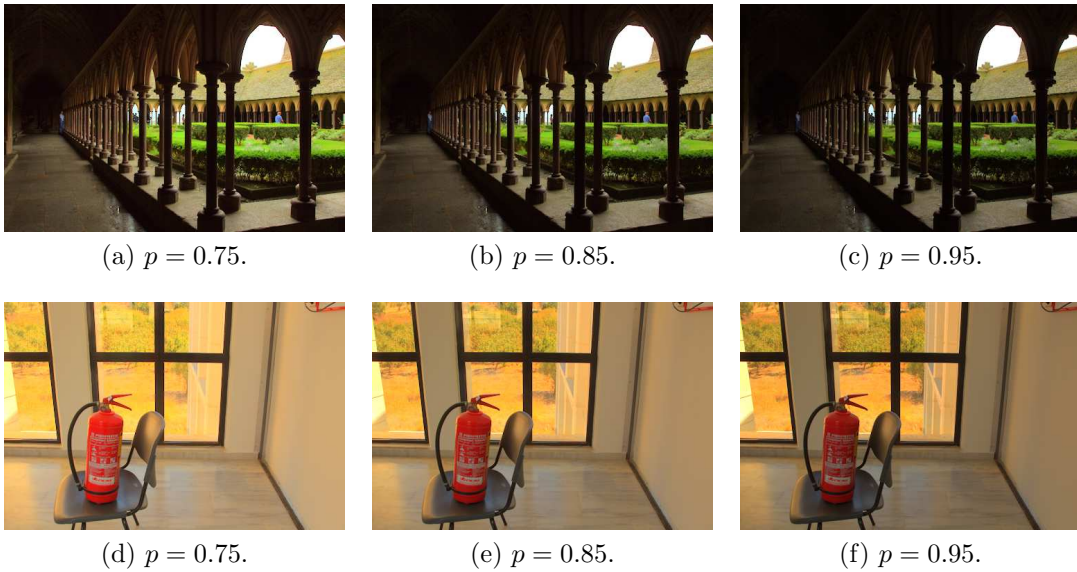
Figure 2.18: Images resulting from the Drago tone-mapping operator with different values of the parameter p. Lower values of p compress more bright areas and make details in dark areas more visible, such as the hallways in the Courtyard image and the white letters of the fire extinguisher.

where $Q(m, n)$ denotes the brightness value in brils, L is the input luminance in cd/m^2 and L_a is the adaptation luminance. The parameters $C_0 = 0.3698$ and the function $\gamma(L_a)$ of the brightness function are derived from the psychophysical data measured by Stevens and Stevens [17]. The gamma function $\gamma(L_a)$ is given by [16]:

$$\gamma(L_a) = \begin{cases} 2.655, & \text{if } L_a > 100 \text{ cd/m}^2 \\ 1.855 + 0.4 \cdot \log_{10}\left(L_a + 2.3 \cdot 10^{-5}\right), & \text{otherwise.} \end{cases} \qquad (2.25)$$

Substituting L and L_a in Equation (2.24) by the corresponding luminances and adaptation luminance of the real scene and the display and then equating both resulting brightness maps yields the tone-mapping operator expression:

$$L_d(m, n) = L_{da} \cdot \left(\frac{L_w(m, n)}{L_{wa}}\right)^{\gamma(L_{wa})/\gamma(L_{da})}, \qquad (2.26)$$

where L_d and L_w denote the display and real scene luminances, respectively and L_{da} and L_{wa} denote the corresponding display and real scene adaptation luminances. The real scene adaptation luminance is estimated according to the average log formula from Equation (2.12). An additional scale factor $f(L_{da})$ may be introduced in Equation (2.26), in order to slightly enhance contrast in tone-mapped scenes, which were obtained from real scenes that present low contrast, such as dim scenes. The factor $f(L_{da})$ is given by:

$$f(L_{wa}) = \left(\sqrt{C_{max}}\right)^{\frac{\gamma(L_{wa})}{1.855+0.4\cdot\log_{10}(L_{da})}-1}, \qquad (2.27)$$

where C_{max} is the maximum contrast that can be achieved by the display device. The final expression for the tone-mapping operator is expressed below:

$$L_d(m, n) = f(L_{da}) \cdot L_{da} \cdot \left(\frac{L_w(m, n)}{L_{wa}}\right)^{\gamma(L_{wa})/\gamma(L_{da})}. \qquad (2.28)$$

Figure 2.19 shows the effect of different values of C_{max} when considering the same scene under low and high illumination conditions, in order to simulate different contrast levels. In real scenes with more contrast, variation of this parameter produces almost no difference in the resulting images. On the other hand, for real scenes with low illumination, in which the contrast is low, this parameter can be used to enhance the contrast of the resulting tone-mapped image. Figure 2.20 shows the images resulting from this operator obtained from original scenes under different illumination conditions (that is, pre-calibrated by different factors L_{max}).

2.4.6 Reinhard Global

Reinhard and Devlin proposed a tone-mapping operator which is inspired by the human visual system photoreceptor response to input light [16]. The authors argue that logarithmic models may not accurately represent the human visual system response over a wide range of luminance values, since such models may yield nega-

(a) $C_{max} = 30$.   (b) $C_{max} = 70$.   (c) $C_{max} = 110$.

(d) $C_{max} = 30$.   (e) $C_{max} = 70$.   (f) $C_{max} = 110$.
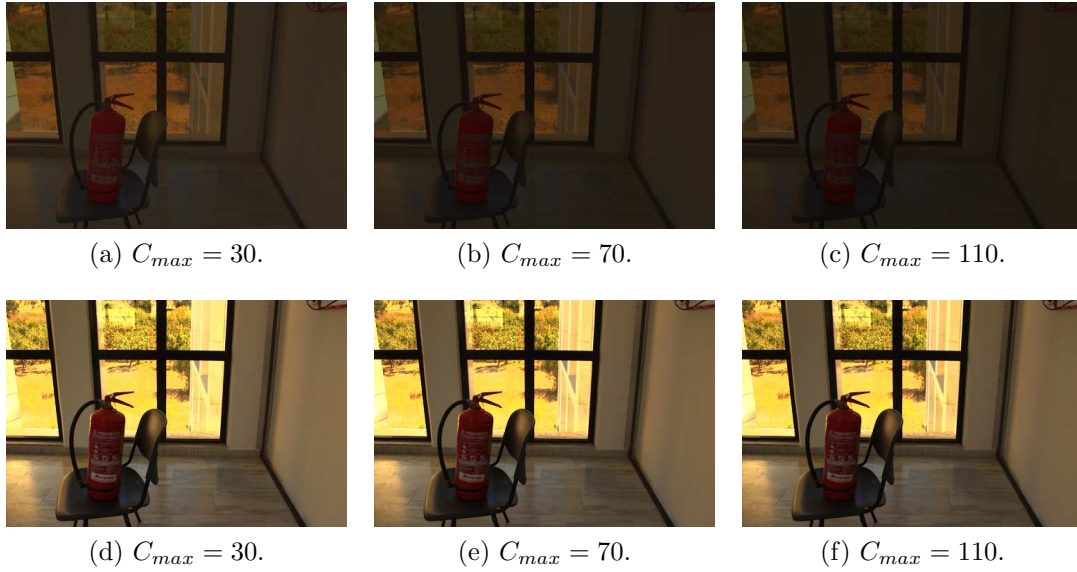
Figure 2.19: Images resulting from the Tumblin-Rushmeier tone-mapping operator with different values of the parameter C_{max}. For dim (low contrast) original scenes, this parameter can be used to slightly enhance the contrast of the tone-mapped image. For original scenes that present more contrast, this parameter barely affects the resulting tone-mapped images. In the top row, the input images were pre-calibrated by a factor $L_{max} = 0.1$, while in the bottom row, the pre-calibration factor used is $L_{max} = 100$.

(a) $L_{max} = 1$.   (b) $L_{max} = 100$.   (c) $L_{max} = 10000$.

(d) $L_{max} = 1$.   (e) $L_{max} = 100$.   (f) $L_{max} = 10000$.
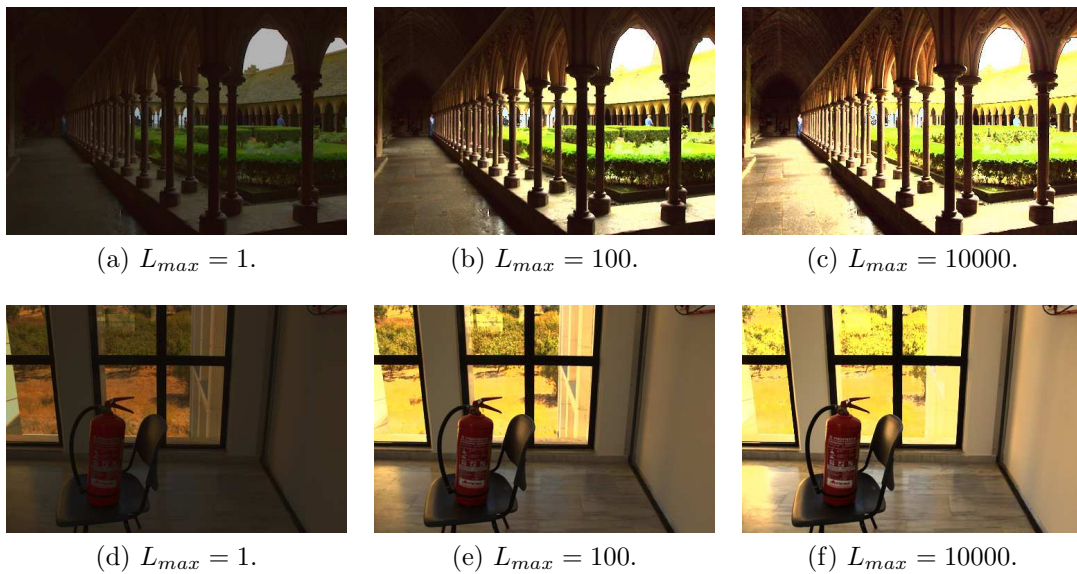
Figure 2.20: Images resulting from the Tumblin-Rushmeier tone-mapping operator with input images pre-calibrated with different factors, in order to simulate different illumination conditions.

tive values (which have no physical interpretation) and indefinitely high values for ever-increasing input luminance values. Electrophysiological studies conducted to measure the human eye photoreceptors response suggest that this response follows

(a) $f = 100$, $\gamma = 0.2$. (b) $f = 100$, $\gamma = 0.5$. (c) $f = 100$, $\gamma = 0.8$.

(d) $f = 1$, $\gamma = 0.8$. (e) $f = 100$, $\gamma = 0.8$. (f) $f = 10000$, $\gamma = 0.8$.
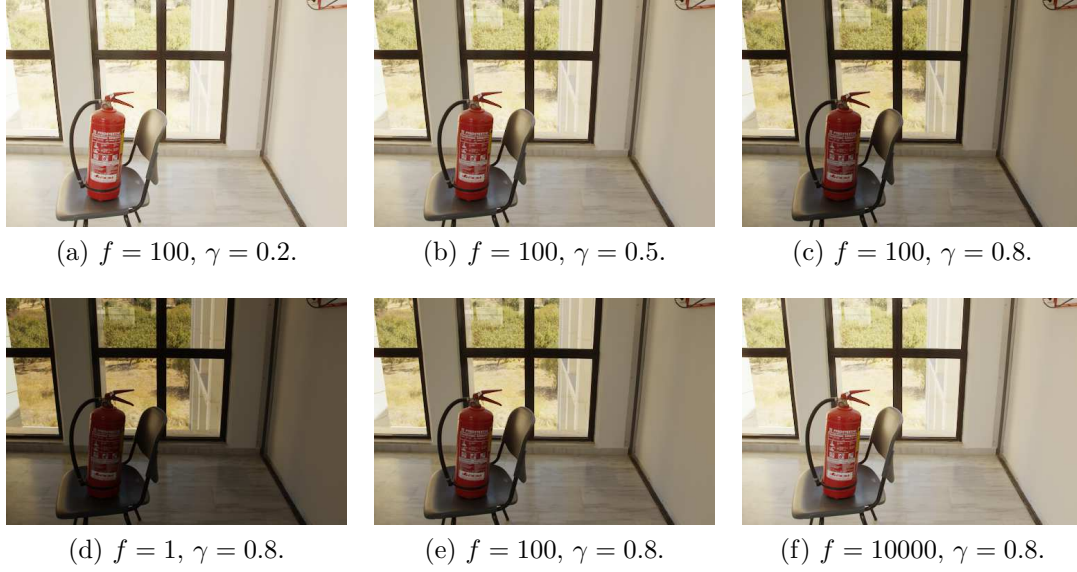
Figure 2.21: Images resulting from the Reinhard global tone-mapping operator with varying values of the γ parameter, which controls the contrast of the image (top row), and different values for the calibration factor f, which influences the overall impression of brightness of the image (bottom row).

an S-shaped curve. Functions with such shapes are also known as *sigmoids*. Based on these studies, the authors use a sigmoidal function to calculate the display input values. The tone-mapping operator is expressed as:

$$I_{TM}(m,n) = \frac{f \cdot I(m,n)}{f \cdot I(m,n) + \sigma(I_a(m,n))}, \tag{2.29}$$

where I_{TM} and I denote the tone-mapped and input images, f is a calibration factor to convert the input image pixel values to absolute luminance values and $\sigma(I_a)$ is a factor, which changes according to the photoreceptor adaptation level $I_a(m,n)$. The factor $\sigma(I_a)$ controls the shape of the sigmoid function and is defined below:

$$\sigma(I_a(m,n)) = (f \cdot I_a(m,n))^\gamma. \tag{2.30}$$

In Equation (2.30), γ is a user-defined parameter that affects the overall contrast impression. Although an empirical formulation to estimate the value of this parameter is offered by the authors, they report that this parameter value usually lies between 0.2 and 0.9, based on electrophysiological studies. Therefore, in this work, it is considered that the parameter γ varies within this range. Figure 2.21 shows the effects of different values for the parameter γ and calibration factor f in the resulting images.

Instead of using a single luminance adaptation value for the real scene and the displayed scene, this operator assumes that, because of the eye saccade movement through the observed scene, different adaptation values may occur, depending on the

30

point in the image fixed by the eye short time interval. Based on the assumption that the cone photoreceptors in the human eye operate independently, the tone-mapping operator is separately applied to each color channel of the input image. The calculation of the adaptation level I_a takes into account two aspects of the human vision, namely chromatic and light adaptation, which are, for clarity, first considered individually.

Chromatic adaptation refers to the human eye capability of perceiving the same colors in objects, regardless of the light source color illuminating them (that is, the eye adapts itself to the current illuminant). Image sensors, on the other hand, do not have this adaptive behavior and are therefore sensitive to the color of the light source. As a consequence, the image registered by the sensors may have an overall color shift towards the color of the illuminant used. This effect is called *color cast*. In order to simulate this feature of the human visual system and remove color casts from images, this operator calculates the adaptation level for each pixel by considering its corresponding red, green and blue color channels values, as well as its luminance value. The weight associated with the luminance and color information of every pixel, which defines the relative importance of each information in the calculation of the adaptation level, is controlled by an user-specified parameter c. Considering only the chromatic adaptation, the adaptation level I_a is calculated as follows:

$$I_a(m, n) = c \cdot I_{r|g|b}(m, n) + (1 - c) \cdot L(m, n), \tag{2.31}$$

where $I_{r|g|b}(m, n)$ denotes the red, green and blue component values of the pixel and $L(m, n)$ denotes the luminance of the pixel. Figure 2.22 shows the images that result from different values of the parameter c.

Light adaptation refers to the human eye adaptation, over time, to new illumination conditions. It temporally changes contrast perception and, consequently, scene visibility. Because of the eye rapid movements through the scene, illumination levels from different parts of the scene contribute to the determination of the luminance level to which the eye adapts. Considering only this effect, the calculation of the adaptation level $I_a(m, n)$ uses both the local pixel value and the global average pixel value and is expressed as:

$$I_a(m, n) = a \cdot I_{r|g|b}(m, n) + (1 - a) \cdot I_{r|g|b}^{avg}, \tag{2.32}$$

where $I_{r|g|b}$ denotes the red, green and blue component values of the pixel, $I_{r|g|b}^{avg}$ denotes the red, green and blue color channels average value and a is an user-specified parameter that controls the contributions of local and global pixel values to the calculation of the adaptation level. Figure 2.23 shows the images that result from different values of the parameter a.

(a) $c = 0$.

(b) $c = 0.33$.

(c) $c = 0.67$.

(d) $c = 1$.

Figure 2.22: Images resulting from the Reinhard global tone-mapping operator with different values of the parameter c, simulating the chromatic adaptation effect of the human eye.

(a) $a = 0$.

(b) $a = 0.33$.

(c) $a = 0.67$.

(d) $a = 1$.

Figure 2.23: Images resulting from the Reinhard global tone-mapping operator with different values of the parameter a, simulating the light adaptation effect of the human eye, which influences the contrast perception in the scene.

Both adaptation effects are combined in Equation (2.33), thereby yielding the final formula for the adaptation level I_a:

$$I_a^{local}(m,n) = c \cdot I_{r|g|b}(m,n) + (1-c) \cdot L(m,n)$$
$$I_a^{global} = c \cdot I_{r|g|b}^{avg} + (1-c) \cdot L^{avg} \qquad (2.33)$$
$$I_a(m,n) = a \cdot I_a^{local}(m,n) + (1-a) \cdot I_a^{global},$$

where L^{avg} denotes the average luminance value of the scene (in cd/m^2 unit). Using the calculated adaptation levels from Equation (2.33) in Equation (2.30) yields the corresponding factor value σ for each pixel in the image. The final tone-mapped image is then obtained by substituting these values in Equation (2.29).

Because this operator considers only the very pixel value to perform the computation of the tone-mapped value, it is categorized as a global tone-mapping operator. In this work, the term "global" is explicitly used to denote this operator, in order to distinguish it from the other tone-mapping operator proposed by the same author, in which the tone-mapped pixel value depends not only on the corresponding original pixel value, but also on its surroundings. This other tone-mapping operator, denoted as Reinhard Local, is discussed in Section 2.4.11.

2.4.7   Schlick

Schlick proposed a tone-mapping operator which is also a rational function and contains only a few user-defined parameters. In [23], the author briefly analyzes commonly used (and fairly simple) tone-mapping functions, such as logarithmic and exponential mappings, and argues that these functions contain parameters that must be calibrated for every image, in order to yield the best results. The calibration is not performed in an automatic fashion, but rather in a trial-and-error procedure by the user, which is a time-consuming and possibly difficult task, depending on the number of parameters considered. The author then proposes a scheme in which only a few parameters require calibration by the user, and their calibration is a fast and straightforward process. The tone-mapping function consists in the ratio between two polynomials and is expressed below:

$$L_d(m,n) = \frac{p \cdot L_w(m,n)}{(p-1) \cdot L_w(m,n) + L_{max}}, \qquad (2.34)$$

where L_d denotes the display luminance values (normalized to the [0,1] range), L_w denotes the real scene luminance values, L_{max} denotes the maximum luminance value of the real scene and p is a model parameter. The parameter p is automatically estimated using the following formula:

$$p = \frac{\delta L_0}{N} \cdot \frac{L_{max}}{L_{min}}, \tag{2.35}$$

where N is the maximum number of different gray levels that can be represented by the display (which, for conventional displays, is equal to 256), L_{min} is the minimum display luminance of the real scene, and δL_0 is the quantized display luminance value associated with the smallest non-zero gray level observed by the user on the display device. The parameter δL_0 can be determined by displaying different gray patches, each one representing a gray level of the display device, on a black background and then asking the user to pick the first gray patch that is distinguishable from black.

An alternative procedure for calculating the parameter p is also proposed by the author. This procedure takes into account the subjective brightness perception, in which a pixel may be perceived as brighter or darker depending on the overall brightness of its surroundings. If the luminance value representing the surroundings of a pixel $L_{surr}(m,n)$ is low, then the pixel should appear brighter. On the contrary, the pixel should appear darker if the luminance of its surroundings is high. In order to determine whether the luminance of the surroundings $L_{surr}(m,n)$ corresponds to a dark or bright region, its value is divided by the geometrical mean of the real scene dynamic range $\sqrt{L_{min} \cdot L_{max}}$. The choice for the geometrical mean is because it equally divides the dynamic range of the scene into two ranges. If the ratio $L_{surr}(m,n)/\sqrt{L_{min} \cdot L_{max}}$ is greater than one, than the pixel is considered to be within a bright region; otherwise, it is considered to be within a dark region.

Several ways of calculating the surrounding luminance of a pixel $L_{surr}(m,n)$ were tested by the author, such as low-pass filtering the image or segmenting the image into zones of similar luminance values. However, as reported by the author, the best results were obtained when the size of the considered surrounding zone reduced to the size of one pixel, that is, the luminance of the surroundings is equal to the own pixel luminance ($L_{surr}(m,n) = L_w(m,n)$). The value of the parameter p, incorporating the subjective brightness perception, is then given by:

$$p = \frac{\delta L_0}{N} \cdot \frac{L_{max}}{L_{min}} \cdot \left(1 - k + k \cdot \frac{L_w(m,n)}{\sqrt{L_{min} \cdot L_{max}}}\right), \tag{2.36}$$

where k is a parameter that weighs the relative importance of the subjective brightness perception feature on the tone-mapping operation. Figure 2.24 shows the resulting images for different values of the parameters δL_0 and k.

2.4.8 Ward Histogram Adjustment

Ward *et al.* later proposed another tone-mapping operator which incorporates more aspects of the human visual system [4]. The operator aims to reproduce a scene on

(a) $\delta L_0 = 10^{-2}$, $k = 0$.   (b) $\delta L_0 = 10^{-2}$, $k = 0.4$.   (c) $\delta L_0 = 10^{-2}$, $k = 1$.

(d) $\delta L_0 = 10^{-1}$, $k = 1$.   (e) $\delta L_0 = 10^{-3}$, $k = 1$.   (f) $\delta L_0 = 10^{-5}$, $k = 1$.
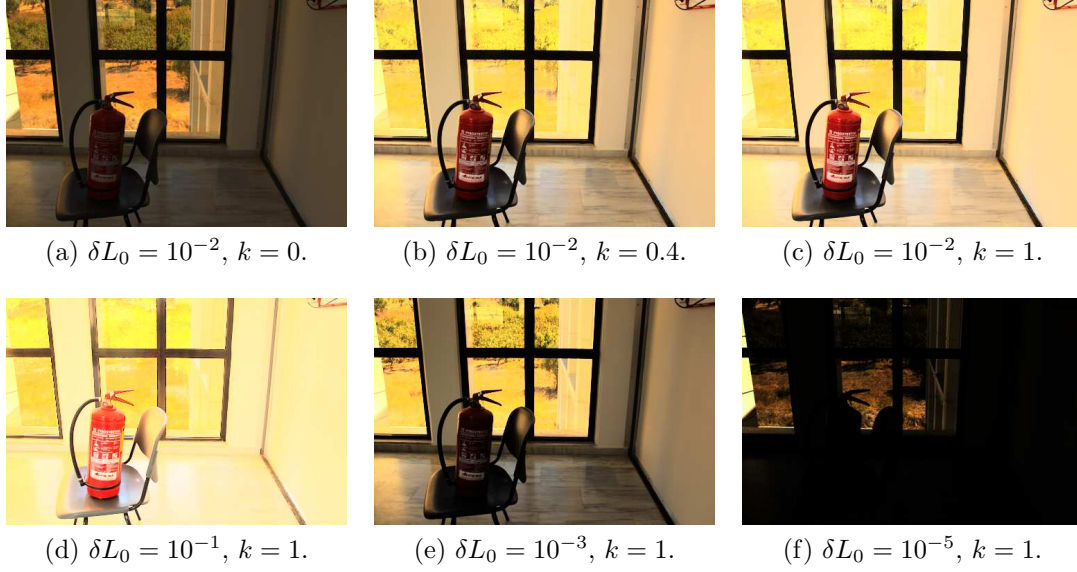
Figure 2.24: Images resulting from the Schlick tone-mapping operator applied to the Fire Extinguisher image, with varying values of the k parameter (top row) and the δL_0 parameter (bottom row).

the display as closely as possible to how an observer views the same scene in the real world, thus recreating the same subjective experience of viewing the real and the displayed scene. The operating principle of the proposed tone-mapping function is based on the idea of altering the histogram distribution of the image, which is a technique commonly used in the context of image enhancement mainly to adjust image contrast and visibility. Since the purpose of the operator is to simulate exactly the same attributes of the real scene, such as brightness and contrast, on the display and not enhance them, the modifications on the image histogram distribution take into account perceptual models of the human visual system.

The authors argue that instead of being uniformly distributed along the dynamic range, the luminance levels of an HDR scene are sparsely distributed, occurring in clusters. Since the human eye is sensitive to relative rather than absolute luminance variations, contrast visibility of the scene is preserved, as long as brighter regions of the scene are associated with higher display luminance values than darker regions. Because of the sparsity of the histogram, more gray levels should be assigned to highly populated areas of the histogram distribution than to lowly populated areas, in order to preserve visibility.

Before the tone-mapping operation takes place, the image is resampled to a resolution, in which each pixel roughly corresponds to 1º of the visual field. This transformation is based on the fact that the eye constantly changes its fixation point in the scene, focusing in different regions. Each focused region lies in the fovea region of the eye, which approximately corresponds to 1º of the visual field. The human eye

then constantly adapts itself to the luminance level of the foveal fixation point. Pixel values of the resampled image roughly represent the adaptation luminance values in each fixation point of the image. Details on the calculation of this resolution are in [4] (in this work, it is considered that the adaptation luminance values are approximated by the luminance value of each pixel in the original image and thus this resizing step is not performed).

After resizing the image, its brightness histogram distribution is calculated. The brightness values of the real scene $B_w(m, n)$ are estimated by taking the logarithm of the real scene luminance values $L_w(m, n)$: $B_w(m, n) = \log(L_w(m, n))$. The tone-mapping operation, consisting of the histogram equalization of the brightness distribution, can then be applied, in order to yield the display luminance values $L_d(m, n)$:

$$\log(L_d(m, n)) = \log(L_{dmin}) + [\log(L_{dmax}) + \log(L_{dmin})] \cdot P(\log(L_w(m, n))), \quad (2.37)$$

where $P(\log(L_w(m, n))) = P(B_w(m, n))$ denotes the *cumulative distribution function* (CDF) of the brightness histogram distribution, which is defined as:

$$P(b) = \frac{1}{T} \cdot \sum_{b_i < b} b_i, \quad (2.38)$$

where b_i represents the quantity of pixels located in the i-th bin of the histogram and T is the total number of pixels. Directly applying this basic version of the histogram equalization may generate exaggerated contrast in some regions of the image, which does not correspond to the contrast perception in the real scene [4]. In order to avoid such effect, the authors establish that the resulting contrast in any region of the image, obtained from the application of this histogram equalization, should not exceed the resulting contrast obtained after applying a linear transformation to the real scene, which preserves the real contrast of the scene (for a given limited dynamic range). A linear ceiling is imposed on the tone-mapping function by stating that the derivative of the function given in Equation (2.37) is less than or equal to the derivative of a linear function, that is:

$$\frac{\mathrm{d}L_d}{\mathrm{d}L_w} \leq \frac{L_d}{L_w}. \quad (2.39)$$

The derivative of the cumulative distribution function corresponds to the histogram distribution with a normalization factor:

$$\frac{\mathrm{d}P(b)}{\mathrm{d}b} = \frac{f(b)}{T\Delta b}, \quad (2.40)$$

where $f(b)$ corresponds to the number of pixels within bin b and $\Delta b =$

$\frac{[\log(L_{wmax}) - \log(L_{wmin})]}{N_{bins}}$ is the size of each bin b (N_{bins} is the number of bins in the histogram). Deriving Equation (2.37) and substituting the resulting expression in Equation (2.39) yields the following expression for the number of pixels at each bin of the brightness histogram $f(b)$:

$$f(b) \le \frac{T\Delta b}{\log\left(L_{dmax}\right) - \log\left(L_{dmin}\right)}. \tag{2.41}$$

Equation (2.41) implies that the original brightness histogram should be altered, in order to assure that each bin does not contain more than a maximum number of pixels (that is, the number of pixels per bin is less than a ceiling value). The authors handled the exceeding pixels at each bin in two different ways: redistributing them to other bins of the histogram and discarding them. They found the latter approach to be the simplest and most reliable one. Since discarding pixels affects the total count of pixels in the image T, the calculation of the ceilings is an iterative process. The authors arbitrarily define that the iteration must last until fewer than 2.5% of the original total number of samples exceed the ceiling. After determining the modified brightness histogram, its cumulative distribution function is used in the tone-mapping function in Equation (2.37), in order to yield the corresponding display luminance values.

Instead of limiting contrast by a generic linear function, this limitation can be determined from the TVI functions that define the human eye perception of contrast under different illumination conditions. The tone-mapped scenes produced using such functions for contrast limitation are expected to correlate more closely the same scene as perceived by an observer in the real world. In this case, the relation in Equation (2.39) is slightly changed:

$$\frac{\mathrm{d}L_d}{\mathrm{d}L_w} \le \frac{\Delta L(L_d)}{\Delta L(L_w)}, \tag{2.42}$$

where $\Delta L(.)$ denotes the TVI function of the human eye. The expression for the number of pixels at each bin of the brightness histogram $f(b)$ is obtained by differentiating Equation (2.37) with respect to L_w and substituting the resulting expression in Equation (2.42):

$$f(\log\left(L_d(m,n)\right)) \le \frac{\Delta L(L_d(m,n))}{\Delta L(L_w(m,n))} \cdot \frac{T\Delta b \cdot L_w(m,n)}{[\log\left(L_{dmax}\right) - \log\left(L_{dmin}\right)] \cdot L_d(m,n)}, \tag{2.43}$$

where $L_d(m,n)$ is the display luminance value, which is obtained from Equation (2.37). In this approach, each histogram bin has its own ceiling value, which depends on the real scene luminance value $L_w(m,n)$, thus contrasting with the previous approach, in which the ceilings are constant for all histogram bins. This is expected,

(a) Linear Ceiling

(b) TVI Ceiling

(c) Linear Ceiling

(d) TVI Ceiling
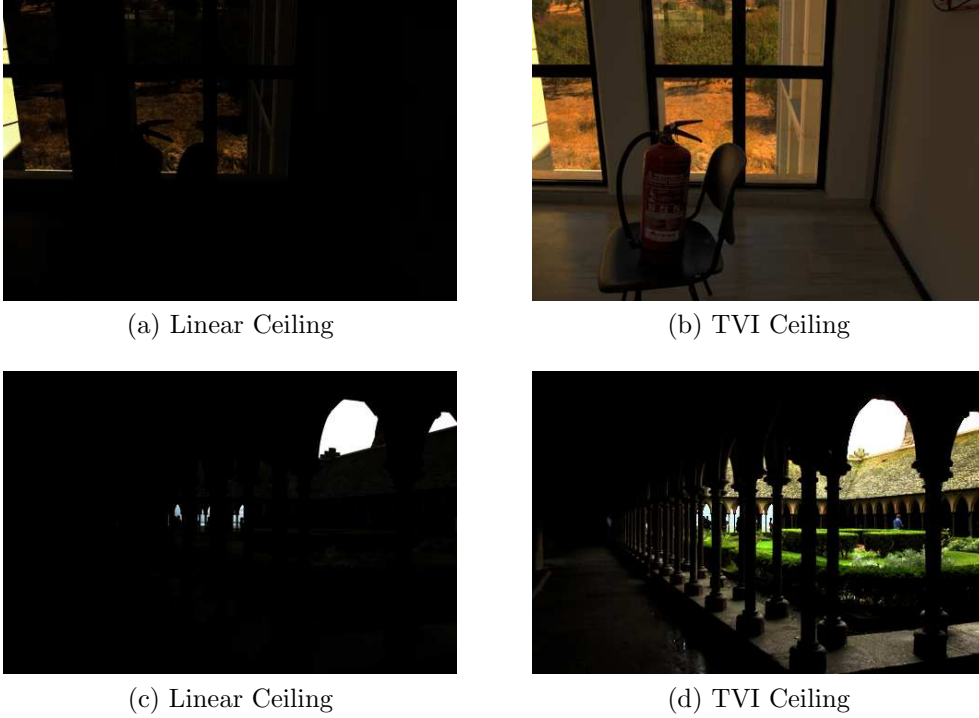
Figure 2.25: Images resulting from the Ward histogram adjustment tone-mapping operator, using different procedures to avoid exaggerated contrast in the tone-mapped images.

since human contrast perception changes according to the illumination level. The maximum number of pixels per bin is calculated through the same iterative procedure as before. The TVI functions used by the operator are expressed below:

$$\log_{10} \Delta L(L_a) = \begin{cases} -2.86, & \text{if } \log_{10}(L_a) < -3.94 \\ (0.405 \cdot \log_{10}(L_a) + 1.6)^{2.18} - 2.86, & \text{if } -3.94 \leq \log_{10}(L_a) < -1.44 \\ \log_{10}(L_a) - 0.395, & \text{if } -1.44 \leq \log_{10}(L_a) < -0.0184 \\ (0.249 \cdot \log_{10}(L_a) + 0.65)^{2.7} - 0.72, & \text{if } -0.0184 \leq \log_{10}(L_a) < 1.9 \\ \log_{10}(L_a) - 1.255, & \text{if } \log_{10}(L_a) \geq 1.9. \end{cases} \quad (2.44)$$

Figure 2.25 shows the images resulting from this operator, using the linear ceilings and the ceilings obtained from the TVI function. The complete version of this operator also models other complex aspects of the human visual system, such as visual acuity, glare (i.e. saturated regions observed around a light source caused by the scattering of the incident light inside the ocular globe) and color sensitivity. The implementation of this tone-mapping operator in this work does not include these additional features.

2.4.9 Chiu

The tone-mapping operator proposed by Chiu *et al.* is among the first ones to introduce the idea of using local information from a given pixel in the image in order

to determine its tone-mapped value [27]. The tone-mapping operation then takes into account not only the pixel value, but also its spatial location. In developing the tone-mapping operator, the authors claim that their intention is not to provide a more comprehensive model of the human visual system or a better alternative to other tone-mapping operators that already existed, but rather to perform an exploratory analysis on how using local information improves (or degrades) the quality of the tone-mapped images.

The authors briefly discuss the problems associated with global linear operators in the context of HDR scenes and state that global linear mappings, in which the output value depends solely on the input pixel value, inevitably lead to loss of details in some regions of the image. In order to discern visible features within dark and bright regions (whose pixels have overall low and high luminance values, respectively), instead of mapping all corresponding pixels to low or high levels (based on their *absolute* luminance values), such pixels should be mapped to different luminance levels, according to their *relative* luminance values, which are given in relation to the overall luminance of the region they are contained within. This idea was motivated by two observations: the fact that the human eye locally adapts to the luminance values around its fixation point (so that different adaptation values exist throughout the image, and not just one single value for the entire image) and that photographers use spatially varying techniques in order to represent the HDR luminance values with a smaller and limited number of gray levels (one of such techniques is discussed in the Subsection 2.4.10).

The tone-mapping operator proposed by the authors consists in applying a scaling function $S(m,n)$, which depends on the pixel position, to the original pixel value. This function is proportional to the inverse of a lowpass filtered version of the input image $I_{filt}(m,n)$: $S(m,n) = \frac{1}{k \cdot I_{filt}(m,n)}$. Lowpass image filtering is implemented by replacing each pixel of the original image by the weighted average of its surrounding pixel values (the size of the surroundings is determined by the size of the filter). The lowpass filter used in this operator is a Gaussian filter kernel given by:

$$G_{\alpha}(m,n) = \frac{1}{2 \cdot \pi \cdot \alpha^2} \cdot e^{\frac{m^2 + n^2}{2 \cdot \alpha^2}}, \tag{2.45}$$

where α is the standard deviation of the Gaussian, which may also be used for setting the filter size. The resulting filtered image is a *blurred* version of the original image and can be interpreted as a rough approximation of the adaptation luminance value at each point in the image. Therefore, dividing the original image by its filtered version yields a relative luminance value for each pixel. In order to assure that the resulting pixel values lie within the normalized range [0,1], a modified version of the

scaling function $S(m, n)$ is defined:

$$\hat{S}(m, n) = \begin{cases} S(i, j), & \text{if } S(m, n) < \frac{1}{I(m,n)} \\ \frac{1}{I(m,n)}, & \text{otherwise,} \end{cases} \qquad (2.46)$$

where I denotes the input image. By denoting $I_{filt}(m, n) = I_{blur}(m, n)$, the tone-mapping operator expression is obtained:

$$I_{TM}(m, n) = \hat{S}(m, n) \cdot I(m, n) = \begin{cases} \frac{I(m,n)}{k \cdot I_{blur}(m,n)}, & \text{if } I(m, n) < k \cdot I_{blur}(m, n) \\ 1, & \text{otherwise,} \end{cases} \qquad (2.47)$$

Figure 2.26 shows the resulting images using different values for the parameters k and σ. The resulting images show a common artifact that tends to arise in local tone-mapping operators, which is called *halo* (or *haloing artifact*). These artifacts correspond to contrast reversals that usually occur near border regions of the image. Because such regions contain abrupt changes in the pixel intensities, depending on the average intensity value against which the pixel intensities are compared, the resulting intensities from both sides of the border can be respectively lower and higher than what they should really be. This unrealistically enhances the contrast in these regions and then generates the halos. Halos tend to increase as the size α of the Gaussian filter increases, because the average value of larger neighborhoods takes into account different regions of the image that have different intensity profiles. Consequently, these average values may not correspond to the average intensity values over small regions of the image, thereby yielding inappropriate intensity values in these regions. The parameter k controls the overall impression of brightness in the images and the intensity of the halos, since it weighs the average luminance of the neighborhood of each pixel. Higher values of k produce darker images and darker halos.

2.4.10   Rahman

The tone-mapping operator developed by Rahman *et al.* [16] takes inspiration on a color vision theory called Retinex theory, proposed by Land [28], which explains certain aspects of human color vision, such as color constancy. In this operator, the input image is filtered several times by different Gaussian kernels. Each filtered version corresponds to the input image at different scales. Images at successively smaller scales contain less details and are generated by increasing Gaussian kernel sizes. The first (and largest) scale corresponds to a Gaussian kernel of size two. At each successive scale, the size of the Gaussian kernel is doubled. The pixel values

(a) $\alpha = 4$, $k = 2$. (b) $\alpha = 16$, $k = 2$. (c) $\alpha = 64$, $k = 2$.

(d) $\alpha = 16$, $k = 2$. (e) $\alpha = 16$, $k = 4$. (f) $\alpha = 16$, $k = 8$.
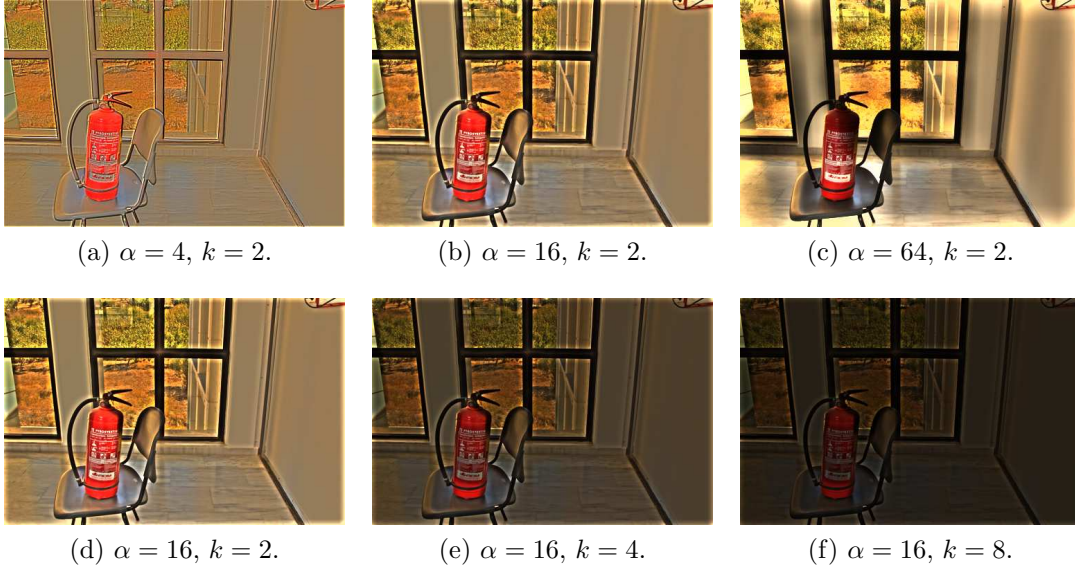
Figure 2.26: Images resulting from the Chiu tone-mapping operator for different values of the parameters α (top row) and k (bottom row). The parameter α controls the size of the halos, while parameter k affects the overall brightness impression in the image and the halo intensities.

of each filtered image represent the average luminance value taken over a pixel neighborhood, whose size is determined by the size of the Gaussian kernel used.

All remaining processing of the operator occurs in log domain, which means that, after the filtering, the logarithm is taken for all images (i.e. the input image and its filtered versions). The input image $\log(I(m,n))$ is then subtracted from each filtered version $\log\left(I_s^{blur}(m,n)\right)$, yielding several difference images $I_s^{dif}(m,n)$. Each difference image is weighted by a value w_s and the difference images are combined together, producing the tone-mapped image (in the log domain) $\log(I_{TM}(m,n))$. The exponential function e^x is applied to the tone-mapped image in the log domain, in order to return to the linear domain. The tone-mapping operation is summarized as follows:

$$
\begin{aligned}
\log(I_{TM}(m,n)) &= \sum_{s=1}^{S} w_s \cdot I_s^{dif}(m,n), \\
I_s^{dif}(m,n) &= \log(I(m,n)) - k \cdot \log\left(I_s^{blur}(m,n)\right),
\end{aligned}
\tag{2.48}
$$

where I_{TM} and I denote the tone-mapped and input image, respectively, $I_s^{blur}(m,n)$ is the input image at the s-th scale, S is the number of scales considered, w_s is the (normalized) weight given to each difference image $I_s^{dif}(m,n)$, and k is the relative weight given to the luminance information of the surroundings of a pixel. The normalized weight w_s is calculated as follows:
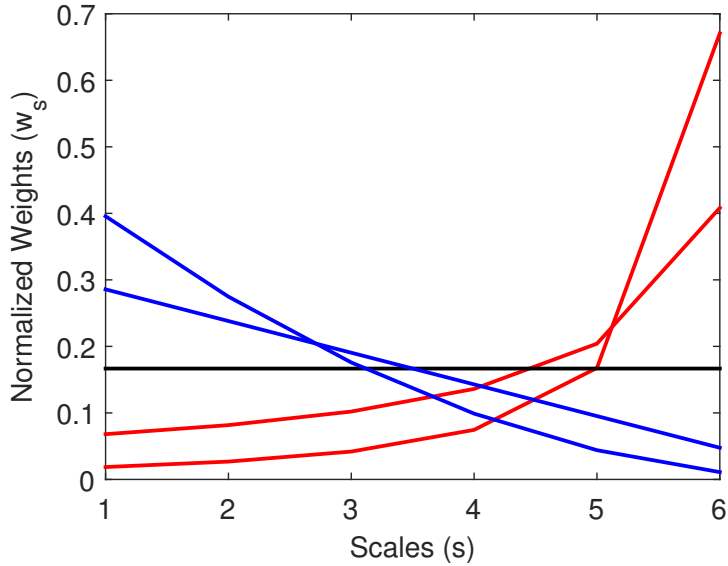
Figure 2.27: Normalized weights w_s, associated with each difference image $I_s^{dif}(m, n)$ at scale s, for different values of the parameter f (f is varied from -2 to 2, with step size 1). The largest and smallest scales correspond to $s = 1$ and $s = 6$, respectively. More weight is given to smaller scales when f is negative (red line). If f is positive, then more weight is given to larger scales (blue line). Equal weight is given to all scales when $f = 0$ (black line).

$$w_s = \frac{(S - s + 1)^f}{\sum_{s=1}^{S}(S - s + 2)^f}. \tag{2.49}$$

In Equation (2.49), f is a parameter that defines the relative weights of each scale. Figure 2.27 shows the corresponding normalized weight curves for different values of the parameter f. The operator is applied independently to each color channel of the input image.

Figure 2.28 shows the images that result from a different set of parameter values. The parameter k affects the overall brightness impression in the image. Lower values of the parameter f tend to generate more halos in the images, since more weight is given to the image differences associated with the smaller scales, which consider larger pixel neighborhoods, thereby inducing more contrast reversals in border regions. Similarly, as more scales are considered, more filtered versions are created and used in the calculation of the resulting image, which tends to produce more halos in the resulting images (this can be mitigated by an appropriate choice of the parameter f).

2.4.11 Reinhard Local

The second tone-mapping operator proposed by Reinhard *et al.* borrows some ideas used in photography and printing to address the tone reproduction problem in HDR

(a) $k = 0.4$, $f = 2$, $S = 6$.   (b) $k = 0.6$, $f = 2$, $S = 6$.   (c) $k = 0.8$, $f = 2$, $S = 6$.

(d) $k = 0.6$, $f = -2$, $S = 6$.   (e) $k = 0.6$, $f = 0$, $S = 6$.   (f) $k = 0.6$, $f = 2$, $S = 6$.

(g) $k = 0.6$, $f = -2$, $S = 2$.   (h) $k = 0.6$, $f = -2$, $S = 4$.   (i) $k = 0.6$, $f = -2$, $S = 6$.
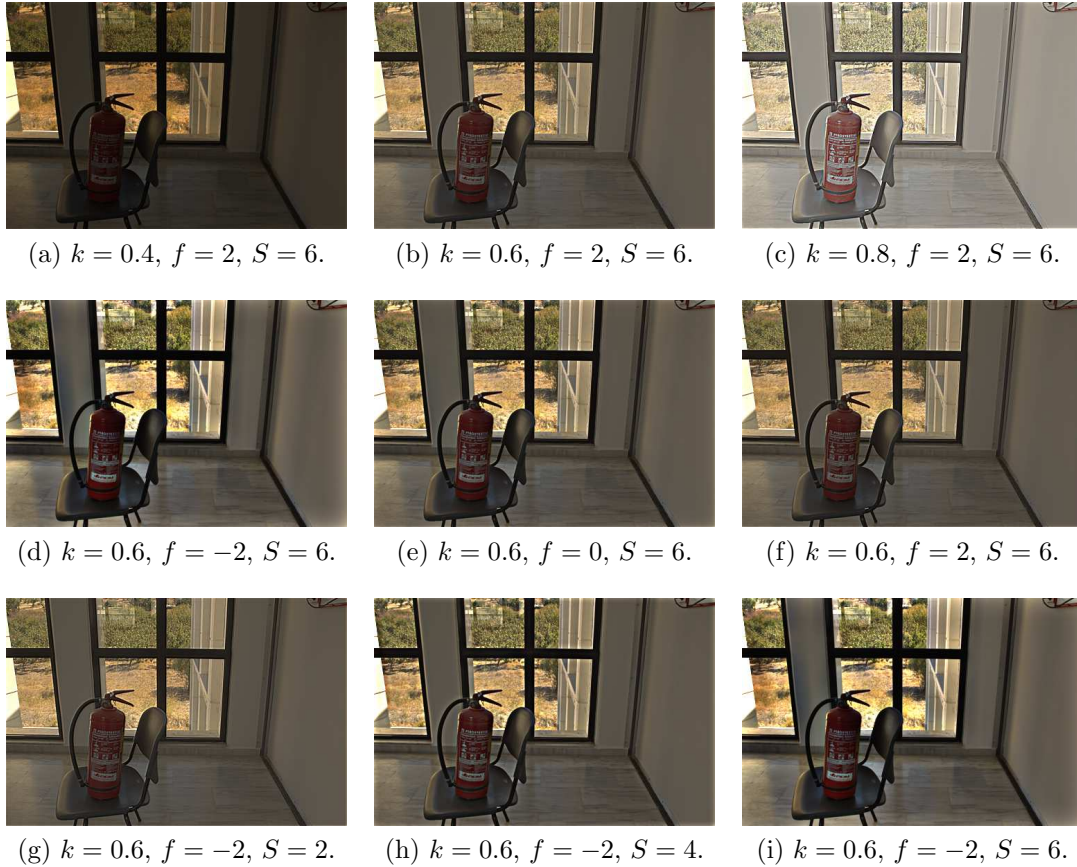
Figure 2.28: Images resulting from the Rahman tone-mapping operator for different values of parameters k (top row), f (middle row) and S (bottom row).

scenes [2]. In printing, the luminances from a scene are translated to printed reflectances. Like a conventional display, the printer can reproduce only a limited number of reflectances, which are organized in the so-called *zones*. There are typically eleven zones (0 - X, in Roman numeral), each one representing a particular reflectance intensity. Each successive zone has twice the intensity of the previous one. The darkest and brightest regions of the scene are mapped to zones 0 and X, respectively. Middle-gray regions which are subjectively perceived as having middle brightness in the scene are normally mapped to zone V (which traditionally corresponds to 18% reflectance in the print [16]), but this may change, depending on the overall impression of brightness (also called the *key*) of the scene. For high-key scenes (i.e. brighter scenes), the middle gray corresponds to brighter zones, while for low-key scenes, it corresponds to darker zones. This procedure, which uses the zone system, is also applicable in photography.

For LDR scenes, the eleven zones are sufficient to reproduce the different luminance intensities, without loss of detail, and the mapping is direct. However, for HDR scenes, details are lost in very bright and very dark regions, since they are mapped to the same zone (X or 0, respectively). In photography, when loss of detail

(a) $a = 0.09$ (b) $a = 0.18$

(c) $a = 0.36$ (d) $a = 0.72$
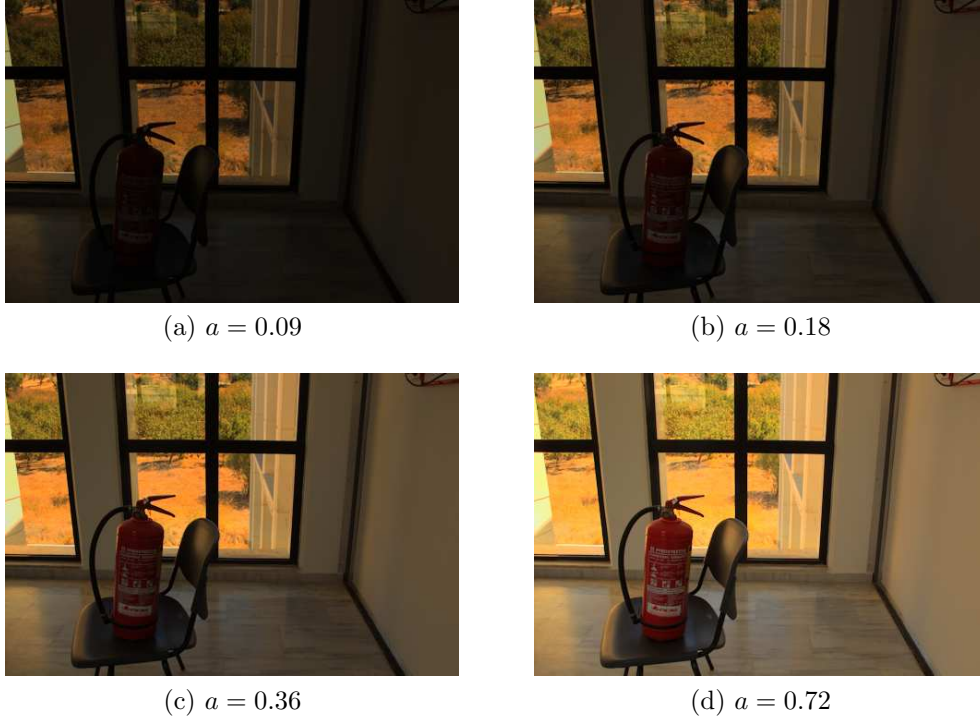
Figure 2.29: Effects of varying the parameter a of the Reinhard local tone-mapping operator in the resulting images. This parameter represents the key of the scene and, consequently, influences the overall impression of brightness.

in these regions is undesirable, a well-known technique, called *Dodging-and-Burning*, is used, which consists in under- or overexposing parts of these regions to light in a controllable fashion. By using this technique, instead of mapping entire regions to the corresponding extreme zones, portions of the dark region are mapped to different dark zones, and, similarly, portions of the bright region are mapped to different bright zones, thereby preserving details in these regions.

The authors propose a tone-mapping operator that simulates an automatic version of the Dodging-and-Burning technique. First, the luminance values L_w of the input image are normalized according to the following relation:

$$L(m, n) = \frac{a}{\overline{L_w}} \cdot L_w(m, n), \tag{2.50}$$

where $\overline{L_w}$ denotes the average log luminance of the scene (calculated using Equation (2.12)) and a is a user-defined parameter that corresponds to the key of the scene. This parameter affects the overall brightness of the scene, as shown in Figure 2.29.

The idea of the Dodging-and-Burning technique is to associate darker and less dark pixels in dark regions to darker and less dark gray levels (that is, these pixels are *globally* dark, but *locally*, over some neighborhood, some are darker and some are brighter than others). A similar procedure applies for the bright regions. In order to determine whether a pixel in some region is brighter or darker than its surroundings,

it is necessary to define the size (i.e. the boundaries) of the surroundings that compose such region. Once the surrounding region is determined, its average pixel value is calculated and then pixels are considered darker or brighter with respect to this value.

The image can be divided into regions with different brightness profiles, to which the technique is applied. Each region has a particular average pixel value. Two regions are considered to have different brightness profiles if the difference between their average pixel values is large. These differences can be used to establish the boundaries of each different brightness region in the image. Therefore, in order to find such boundaries, the normalized input image L is filtered by two Gaussian kernels, given by:

$$
\begin{aligned}
R_s(m,n) &= \frac{1}{\pi \alpha_s^2} \cdot e^{-\frac{m^2+n^2}{\alpha_s^2}}, \\
R_{s+1}(m,n) &= \frac{1}{\pi \alpha_{s+1}^2} \cdot e^{-\frac{m^2+n^2}{\alpha_{s+1}^2}},
\end{aligned}
\tag{2.51}
$$

where α_s and α_{s+1} denote the standard deviation of the Gaussian kernel at the s-th and $(s+1)$-th scale, respectively. Each pixel of the filtered image corresponds to the average value of its surroundings, whose size varies according to the size of the Gaussian filter used (defined by its standard deviation). The two filtered images $V_i(m,n,s)$ are then subtracted from each other and the result is normalized by the filtered image of the pair that considers the smallest surroundings size, yielding the normalized surrounding difference image $V(m,n,s)$ [16]:

$$
V_s(m,n) = \frac{|L_s^{blur}(m,n) - L_{s+1}^{blur}(m,n,s)|}{2^\phi \cdot a/s^2 + L_s^{blur}(m,n)}.
\tag{2.52}
$$

The term $2^\phi \cdot a/s^2$ avoids indetermination in the division when $L_s^{blur}(m,n)$ is zero. The parameter ϕ defines the sharpness of the edges of the resulting image, as shown later. The difference between the filtered images corresponds to the difference between the average pixel values of two different neighborhoods. If $V_s(m,n)$ is higher than a user-defined threshold ϵ, then the maximum neighborhood size to be considered for the corresponding pixel is defined by the Gaussian size that generated $L_s^{blur}(m,n)$, which is the smallest Gaussian size in the pair of kernels. The absolute value of this difference is normalized, in order to make the threshold ϵ independent from absolute luminance values and, thus, easier to adjust.

Since the neighborhood size of each pixel may vary, this process is repeated for different pairs of Gaussians with increasing sizes (which represent larger neighborhoods), until the maximum scale s_{max} for which the condition $V_{s_{max}}(m,n) < \epsilon$ holds for all pixels. For every Gaussian pair, the ratio between the Gaussian sizes is fixed at 1.6, that is to say $\alpha_{s+1} = 1.6 \cdot \alpha_s$. According to the author, this value is chosen

because the difference of Gaussians then resembles a Laplacian of Gaussian filter [2]. In the present work, the smallest Gaussian size, at scale $s = 1$, has standard deviation $\alpha_1 = \frac{1}{2\sqrt{2}}$, which is also the same value used in [2]. After calculating the average luminance value of each pixel taking into account its neighborhood $L^{blur}_{s_{max}}$, the tone-mapping operation is applied:

$$L_d(m, n) = \frac{L(m, n)}{1 + L^{blur}_{s_{max}}(m, n)}. \tag{2.53}$$

Dark pixels in bright surroundings are assigned lower values by the operator, because, compared to their surroundings, they are very dark (in this case, $L(m, n) < L^{blur}_{s_{max}}(m, n)$). In dark surroundings, dark pixels are assigned higher values, because, in this case, $L(m, n) \approx L^{blur}_{s_{max}}(m, n)$, that is, compared to their surroundings, they are not very dark. A similar analysis is valid for bright pixels in bright and dark regions. The operator then simulates the Dodging-and-Burning technique.

Figure 2.30 shows the effect of varying the parameter ϵ in the images resulting from this operator. As ϵ increases more halos become visible. This is because larger neighborhood sizes are considered for each pixel, since the threshold value that defines different brightness regions in the image is higher. These neighborhoods include a wide variety of pixel intensities, and it is likely that the average luminance values of such neighborhoods do not correspond to the average luminance values of pixels in particular small regions of the image. This leads the operator to incorrectly increase or decrease the pixel value in some regions, such as borders, thus creating these halos. A similar explanation is valid for the decrease in the halos size and number in the image, as the value of the ϵ parameter decreases. The effects of varying the parameter ϕ are shown in Figure 2.31. Higher values of the parameter ϕ lead to sharper edges, but also induce more haloing artifacts.

2.4.12 Ashikhmin

Ashikhmin proposed a tone-mapping operator that has the same principle of linear operators, which is to preserve local contrasts in the image [29]. In order to achieve this, it considers the human visual system contrast perception, which depends on the adaptation luminance level and is modelled by the TVI functions. One of the key points of this operator is that, instead of considering one single adaptation luminance value for the HDR scene, the operator calculates different adaptation values for each point in the image. In this case, each point in the image has a small range of discernible contrasts, to which different display luminance levels can be assigned, thus preserving details throughout the image. The author defines local contrast as:

(a) $\epsilon = 0$ (b) $\epsilon = 0.05$

(c) $\epsilon = 0.1$ (d) $\epsilon = 0.5$
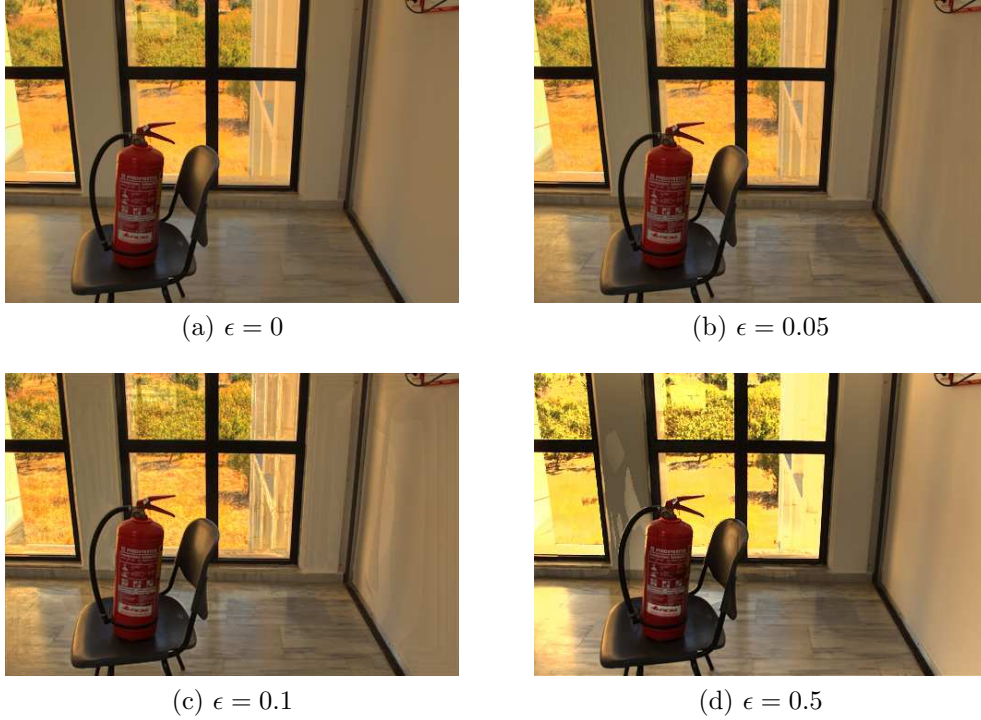
Figure 2.30: Effects of varying the parameter ϵ of the Reinhard local tone-mapping operator in the resulting images. More halos are produced as ϵ increases, because larger neighborhood sizes are considered for each pixel in the image. When $\epsilon = 0$, the operator becomes purely global, since no information from the surroundings of a pixel is used to determine its value.

(a) $\phi = 1$ (b) $\phi = 10$ (c) $\phi = 15$
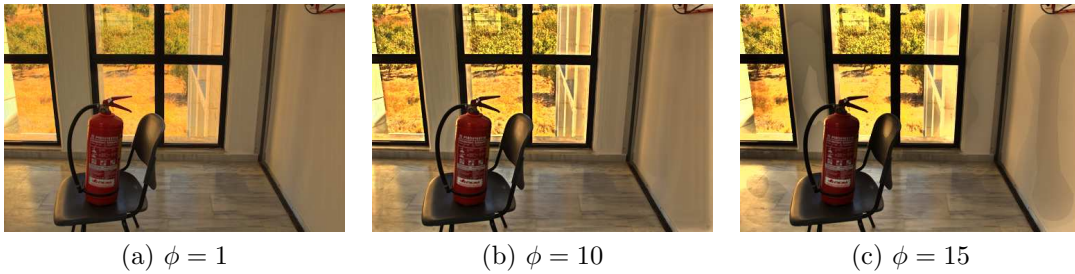
Figure 2.31: Effects of varying the parameter ϕ of the Reinhard local tone-mapping operator in the resulting images. Edges become sharper as ϕ increases, but more halos also arise.

$$c(m, n) = \frac{L(m, n)}{L_a(m, n)} - 1, \tag{2.54}$$

where L and L_a denote the absolute luminance and adaptation luminance values, respectively. Local contrast in the display and in the real scene should be preserved, i.e. $c_d(m, n) = c_w(m, n)$, which leads to the following relation:

$$L_d(m, n) = \frac{L_w(m, n) \cdot L_{da}(m, n)}{L_{wa}(m, n)}, \tag{2.55}$$

where L_{wa} and L_{da} denote the real scene and the display adaptation luminance values. The display adaptation luminance values L_{da} correspond to the values determined by the application of a compressive function $F(L_w)$ to the real scene adaptation luminance values L_{wa}. The tone-mapping operator is fully defined by specifying the function $F(L_w)$ and a procedure for calculating $L_{wa}(m, n)$.

In order to determine the real scene adaptation luminance values, a procedure similar to the one used in the Reinhard Local operator to find the size of each pixel neighborhood is adopted. The adaptation luminance corresponds to the average luminance in a pixel neighborhood with arbitrary size. As stated by the author, the adaptation luminance should vary smoothly, which means that the size of the neighborhood to be considered should be the largest size that does not cause abrupt changes in the average luminance value of the neighborhood. A quantity lc that measures the changes in the average luminance values, when taking into account different neighborhood sizes, is defined as:

$$lc_\alpha(m, n) = \frac{|L_\alpha^{blur}(m, n) - L_{2\alpha}^{blur}(m, n)|}{L_\alpha^{blur}(m, n)}, \qquad (2.56)$$

where $L_\alpha^{blur}(m, n)$ and $L_{2\alpha}^{blur}(m, n)$ denote filtered versions of the input image, which are obtained using Gaussian kernels with sizes α and 2α (α is expressed in pixel units). A tolerance ϵ on the variation of the average luminance is defined. If $lc_\alpha(m, n) < \epsilon$, then α is incremented by one and another pair of Gaussian kernels with larger sizes $\alpha + 1$ and $2(\alpha + 1)$ are used to filter the input image again. The maximum neighborhood size of each pixel α_{max} is the last size for which the condition $lc_{\alpha_{max}}(m, n)$ is still satisfied (thereby assuring that regions which contain highly contrasting pixels are not considered in the calculation of the average luminance of the pixel neighborhood). The process starts with $\alpha = 1$, and it is repeated until either the maximum neighborhood size of each pixel has been found or α reaches the maximum value of 10 pixels, as suggested by the author [29]. The adaptation luminance value of each pixel $L_{wa}(m, n)$ corresponds to the average luminance value computed over the maximum neighborhood size of the given pixel $L_{\alpha_{max}}^{blur}(m, n)$.

After calculating the different luminance adaptation values for each pixel in the image, these values are transformed by the function $F(L_w)$, in order to yield the corresponding display adaptation values. Before determining the function $F(L_w)$, the author defines an auxiliary function $C(L)$, denoted as *perceptual capacity* function. For each adaptation luminance L_a, there exists a JND δL, above which the human eye perceives two luminance values as different. In the context of visibility, each JND is equally important, regardless of the adaptation luminance value they are associated to, because it defines the human eye capacity of distinguishing different luminance levels for the given adaptation luminance. The perceptual capacity

function C then models this ability of the human eye to discern luminance levels over a range of adaptation luminance values. It is defined as follows:

$$C(L) = \int_0^L \frac{dl}{\Delta L(l)},$$ (2.57)

where $\Delta L(.)$ denotes the JND value given by the TVI function for different luminance values l, which are in the $[0,L]$ range. The perceptual capacity within a luminance range $[L_1, L_2]$ is given by $C(L_2) - C(L_1)$. This expression is used for determining the perceptual capacity over the real scene luminance range, because there potentially exists a high number of different luminance adaptation values in an HDR scene. For the displayed scene, since the luminance values are in a low dynamic range, in order to facilitate the calculation of the perceptual capacity over the display luminance range, the adaptation luminance can be approximated by a constant for the entire scene. In this case, the perceptual capacity in the displayed scene C_d becomes:

$$C_d(L_d) = \frac{L_d}{\Delta L(L_{da})}$$ (2.58)

The author uses a simplified TVI function, approximated as four linear segments in the log-log space [29]. The real scene perceptual capacity curve $C(L)$ from Equation (2.57) is then approximated using such TVI function, yielding the following expression (the luminance values are in cd/m²):

$$C(L) = \begin{cases} \frac{L}{0.0014}, & \text{if } L < 0.0034 \\ 2.4483 + \frac{\log\left(\frac{L}{0.0034}\right)}{0.4027}, & \text{if } 0.0034 \leq L < 1 \\ 16.5630 + \frac{(L-1)}{0.4027}, & \text{if } 1 \leq L < 7.2444 \\ 32.0693 + \frac{\log\left(\frac{L}{7.2444}\right)}{0.0556}, & \text{otherwise.} \end{cases}$$ (2.59)

The function $F(L_w)$ is achieved by equating the perceptual capacities from both the real and displayed scenes, considering the luminance ranges from each case and then solving for the display luminance L_d:

$$\frac{C_d(L_d) - C_d(L_{dmin})}{C_d(L_{dmax}) - C_d(L_{dmin})} = \frac{C(L_w) - C_d(L_{wmin})}{C_d(L_{wmax}) - C_d(L_{wmin})},$$ (2.60)

where L_{wmax} and L_{wmin} denote the maximum and minimum luminance of the real scene, and L_{dmin} and L_{dmax} correspond to the minimum and maximum display luminance values. By setting $L_{dmin} = 0$ and replacing the left side of Equation (2.60) by the expression defined in Equation (2.58), an expression for the function $F(L_w)$ is achieved:

| (a) $\epsilon = 0$ | (b) $\epsilon = 0.05$ |

| (c) $\epsilon = 0.1$ | (d) $\epsilon = 0.5$ |
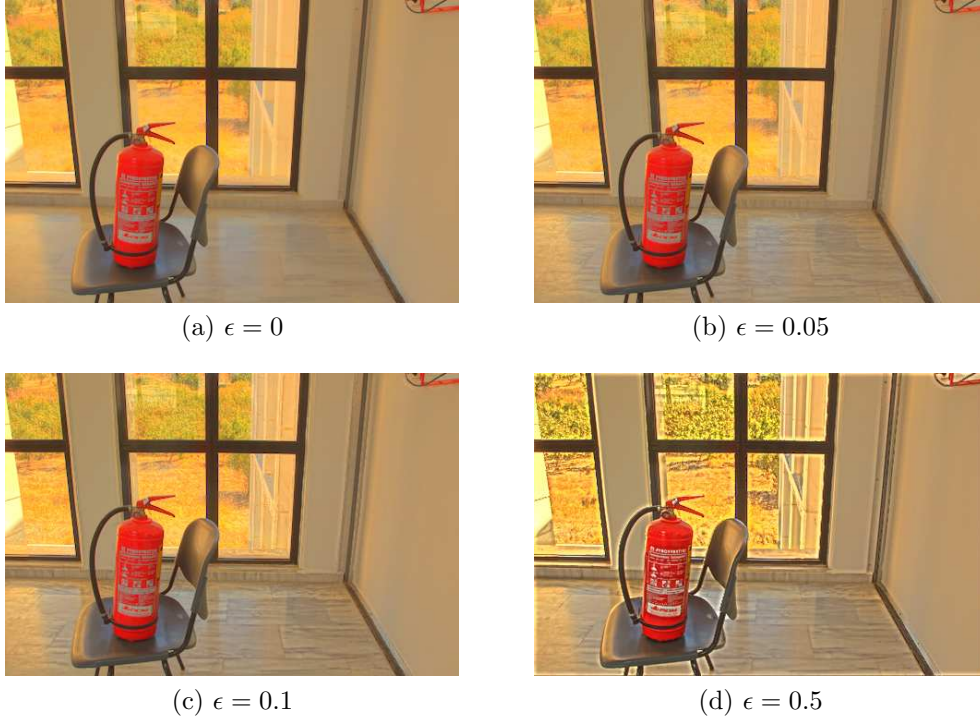
Figure 2.32: Effects of varying the parameter ϵ of the Ashikhmin tone-mapping operator in the resulting images. More halos are produced as ϵ increases, because larger neighborhood sizes are considered for each pixel in the image. Large neighborhoods most likely include highly contrasting pixels, which incurs in abrupt changes in the adaptation luminance values throughout the images.

$$L_d = F(L_w) = L_{dmax} \frac{C(L_w) - C_d(L_{wmin})}{C_d(L_{wmax}) - C_d(L_{wmin})}. \tag{2.61}$$

The display adaptation luminance values L_{da} can be calculated by applying this function to the estimated real scene adaptation luminance values L_{wa}. The resulting display adaptation luminance values are then used in Equation (2.55), in order to yield the final tone-mapped luminance pixel values. Figure 2.32 shows the effects of varying the parameter ϵ in the resulting images. Like in the Reinhard Local operator, this parameter affects the size of the neighborhood to be considered for each pixel, when calculating the real scene adaptation luminance values L_{wa}. If larger neighborhoods are used, then more halos are produced, because the average luminance value of the neighborhood is most likely calculated considering regions with highly contrasting pixels. Consequently, the calculated value does not correspond to the adaptation luminance values from different small regions of the image (in which the pixel values do not vary greatly), thus leading to haloing artifacts.

Table 2.1 summarizes some characteristics of the tone-mapping operators discussed in this chapter. Different images resulting from each operator are shown in Appendix C.

Operator	Global/Local	Linear/Non-linear	Comments
Ward Con. Pre.	Global	Linear	Fails to exhibit details in dark and bright regions simultaneously.
Ferwerda	Global	Linear	Applied on the color channels. Fails to exhibit details in dark and bright regions simultaneously.
Logarithm	Global	Non-Linear	Has no parameters to adjust.
Exponential	Global	Non-Linear	Has one parameter to adjust (L_{norm}).
Drago	Global	Non-Linear	Adjusts the base of its logarithm function according to pixel value.
Tumblin	Global	Non-Linear	Assumes that real-world scene and display brightness values are equal to perform tone mapping.
Reinhard Global	Global	Non-Linear	Applied on the color channels. Considers chromatic and light adaptations.
Schlick	Global	Non-Linear	Has few parameters to adjust.
Ward Hist. Adj.	Global	Non-Linear	Adjusts the image histogram without exaggerating contrast.
Chiu	Local	Non-Linear	Tends to produce halos.
Rahman	Local	Non-Linear	Applied on the color channels. Uses images at different scales to perform tone mapping.
Reinhard Local	Local	Non-Linear	Automatic "Dodging-and-Burning" technique.
Ashikhmin	Local	Non-Linear	Considers multiple adaptation luminance values to perform tone mapping.

Table 2.1: Classifications of each digital tone-mapping operator discussed in this chapter.

Chapter 3

The Focal-Plane Tone-Mapping Operator

The tone-mapping operators discussed so far are purely digital. They are implemented on a digital system that is, in general, independent from the capture device. The digital tone-mapping operators are executed offline, rather than online, since the operations are performed *after* the image is acquired and digitized (with more than 8 bits) and not *as* the image is acquired by the capture device (that is, the operations are not performed on the analog values registered by the sensors of the capture device). In this chapter, a non-digital tone-mapping operator is discussed. It is implemented inside the focal plane of the capture device. Hence, the tone-mapping operation takes place before analog-to-digital conversion, mapping the HDR image to the 8-bit range, as soon as it is acquired.

3.1   Internal Circuit

The Focal-Plane Tone-Mapping Operator (FPTMO) discussed in this chapter, originally proposed by Fernández-Berni *et al.* [11], adjusts the value of each pixel in the sensor array by using two pieces of information: the global luminance of the scene, given by its average value, and the local luminance, given by the amount of incident light on the particular pixel. Figure 3.1 shows the schematic diagram of a pixel in the circuit of the FPTMO from [11].

As can be seen from Figure 3.1, $V_{DD} = 3.3$ V is the circuit supply voltage and the pixel itself is composed by two photodiodes. Each photodiode stores the amount of incident light at the pixel for different purposes. The photocurrent $I_{ph_{m,n}}$ represents the value stored in the photodiode located at the bottom part of the circuit (which is similar to a 4-T pixel structure). This photocurrent value is used to actually determine the pixel final value. This photodiode is called *capture* photodiode and
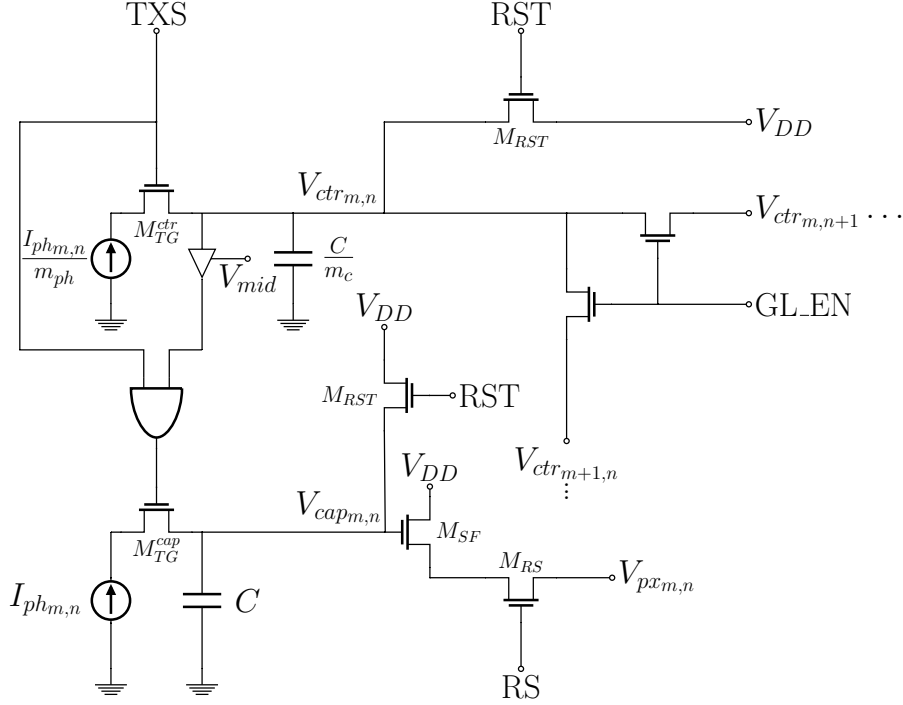
Figure 3.1: Schematic diagram of a pixel in the circuit of the FPTMO from [11].

its associated node, the *capture* node. The capture node voltage value is $V_{cap_{m,n}}$ and its associated capacitance is C.

The value stored in the photodiode located at the upper part of the circuit is required for globally calculating the average luminance of the scene. The voltage value $V_{ctr_{m,n}}$ of the node to which this photodiode is connected controls the duration of the photocurrent integration that takes place in the capture node (also called *integration time*). Hence, this photodiode and its node are called *control* photodiode and *control* node, respectively.

The capture and control photodiodes may not have the same size and, as a consequence, the same light intensity may lead to different photocurrent values in each node. The ratio between the capture node photocurrent and the control node photocurrent is defined as m_{ph}. Likewise, the capacitances associated with the capture and control nodes may also be different. The ratio between the capture node capacitance and the control node capacitance is defined as m_c.

The voltage value V_{mid} is a threshold value used in a comparator circuit. Once the control node voltage $V_{ctr_{m,n}}$ reaches this threshold value, the comparator outputs a low voltage value (which is interpreted as boolean value `false`) to one input of the "AND" logic gate. The output of this logic gate then turns off the capture node transfer gate transistor M_{TG}^{cap}, thereby ceasing the photocurrent integration in the capture node. This yields the final capture node voltage value $V_{px_{m,n}}$. The transistors M_{SF} and M_{RS} are the source follower and the row select transistor, respectively, from the 4-T pixel architecture discussed in Chapter 2.
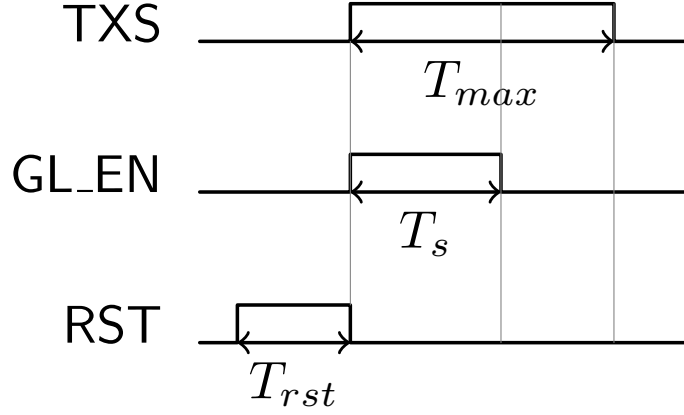
Figure 3.2: Timing diagrams of control signals present in the operator circuit.

The signals TXS, GL_EN and RST define how the circuit operates. The control signal RST is the reset signal (associated with the reset transistor M_{RST}), which removes remaining charges from previous light measurements in the capture and control photodiodes, before a new measurement occurs. The control signal GL_EN interconnects the control nodes from every pixel in the sensor array and, hence, makes the voltage value of every control node evolve equally. The control signal TXS defines the maximum duration for the photocurrent integration in the capture and control nodes, by controlling how long the respective transfer gate transistors M_{TG}^{cap} and M_{TG}^{ctr} remain turned on. Figure 3.2 shows the timing diagrams of these signals. The analysis of the circuit operation can then be divided in three time periods: $t \leq T_{rst}$, $T_{rst} < t \leq T_s$ and $T_s < t \leq T_{max}$. Each of these time periods is next discussed in more detail (other circuit parameters, such as m_{ph}, m_c and C are discussed in Section 3.2).

1) $t \leq T_{rst}$

In the first stage of operation, only the control signal RST is enabled. During the period while the control signal RST is active, the voltage values at the capture and control nodes are the same and fixed at $V_{rst} = V_{DD} - V_{th}$, where V_{th} is the threshold voltage associated with the MOSFET transistor.

$$V_{ctr_{m,n}} = V_{cap_{m,n}} = V_{rst}. \tag{3.1}$$

2) $T_{rst} < t \leq T_s$

In the second stage of operation, the control signal RST is disabled and the control signals GL_EN and TXS are enabled. While the control signal GL_EN is active, the control nodes of every pixel in the sensor array are connected together. The overall photocurrent and overall capacitance associated with the (common) control node are given, respectively, by the sum of every photocur-

54

rent and capacitance present in every control node of the sensor array, since such components are all connected together in parallel. Therefore, assuming the sensor array is composed by M rows and N columns, and m and n denote the pixel located at the m-th row and n-th column of the sensor array, the resulting voltage value of the control node can be calculated according to the following procedure. First, consider that:

$$\sum_{m,n} \frac{I_{ph_{m,n}}}{m_{ph}} = M \cdot N \cdot \frac{C}{m_c} \cdot \frac{\mathrm{d}V_{ctr_{m,n}}}{\mathrm{d}t}. \tag{3.2}$$

Isolating $\mathrm{d}V_{ctr_{m,n}}$ in Equation (3.2) and integrating both sides leads to:

$$\int_{V_{ctr_{m,n}}}^{V_{rst}} \mathrm{d}V_{ctr_{m,n}} = \frac{\sum_{m,n} \frac{I_{ph_{m,n}}}{m_{ph}}}{M \cdot N \cdot \frac{C}{m_c}} \cdot \int_0^t \mathrm{d}t. \tag{3.3}$$

Solving Equation (3.3) for $V_{ctr_{m,n}}(t)$ yields the following expression:

$$V_{ctr_{m,n}}(t) = V_{rst} - \frac{m_c}{m_{ph}} \cdot \frac{\sum_{m,n} I_{ph_{m,n}}}{M \cdot N \cdot C} \cdot t. \tag{3.4}$$

The final expression for the control node voltage $V_{ctr_{m,n}}$ is then obtained by making the substitution $\overline{I_{ph}} = \frac{\sum_{m,n} I_{ph_{m,n}}}{M \cdot N}$:

$$V_{ctr_{m,n}}(t) = V_{rst} - \frac{m_c}{m_{ph}} \cdot \frac{\overline{I_{ph}}}{C} \cdot t, \quad T_{rst} < t \le T_s. \tag{3.5}$$

In Equation (3.5), $\overline{I_{ph}}$ represents the average photocurrent of the sensor array which, in turn, is a representation of the average luminance of the scene. Assuming that the control node voltage $V_{ctr_{m,n}}$ does not reach the threshold voltage V_{mid} and because the control signal TXS is also active, the capture node voltage value $V_{cap_{m,n}}$ is determined by a capacitor discharge equation:

$$V_{cap_{m,n}}(t) = V_{rst} - \frac{I_{ph_{m,n}}}{C} \cdot t, \quad T_{rst} < t \le T_s. \tag{3.6}$$

- $T_s < t < T_{max}$

  In the third and last stage of operation, the control signal GL_EN is disabled and only the control signal TXS is left enabled. Because the control signal GL_-EN is disabled, the control nodes of every pixel in the sensor array no longer remain connected. Thus, the control node voltage of each pixel is determined by its corresponding local photocurrent and local capacitance, according to the procedure that is described next. First, consider that:

$$\frac{I_{ph_{m,n}}}{m_{ph}} = \frac{C}{m_c} \cdot \frac{dV_{ctr_{m,n}}}{dt}. \tag{3.7}$$

Isolating $dV_{ctr_{m,n}}$ in Equation (3.7) and integrating both sides leads to:

$$\int_{V_{ctr_{m,n}}}^{V_{rst} - \frac{m_c}{m_{ph}} \cdot \frac{\overline{I_{ph}}}{C} \cdot T_s} dV_{ctr_{m,n}} = \frac{m_c}{m_{ph}} \cdot \frac{I_{ph_{m,n}}}{C} \cdot \int_{T_s}^{t} dt. \tag{3.8}$$

In Equation (3.8), the upper limit of the left-hand side integral $V_{rst} - \frac{m_c}{m_{ph}} \cdot \frac{\overline{I_{ph}}}{C} \cdot T_s$ corresponds to the capture node voltage value at the instant when the control signal GL_EN is deactivated. Solving for $V_{ctr_{m,n}}$ yields the final expression for the control node voltage:

$$V_{ctr_{m,n}}(t) = V_{rst} - \frac{m_c}{C \cdot m_{ph}} \cdot \left[T_s \cdot \left(\overline{I_{ph}} - I_{ph_{m,n}} \right) + I_{ph_{m,n}} \cdot t \right], \quad t > T_s. \tag{3.9}$$

The voltage value at the capture node $V_{p_{i,j}}$ continues to be determined by Equation (3.6), until either the control node voltage $V_{a_{i,j}}$, given by Equation (3.9), reaches the threshold voltage value V_{mid} or $t = T_{max}$, whichever occurs first. In either case, the transfer gate transistor M_{TG}^{cap} is turned off (either because the output of the comparator located at the control node switches to low or because the control signal TXS is disabled). Consequently, the photocurrent integration at the capture node is ceased, making its voltage ready for readout. The two conditions yield different expressions for the final capture node voltage and are considered next.

i) $V_{ctr_{m,n}} = V_{mid} \ (t < T_{max})$

By denoting $T_{mid_{m,n}}$ as the time instant at which the control node voltage $V_{ctr_{m,n}}$ reaches V_{mid}, the expression for the voltage value at the capture node is given by:

$$V_{px_{m,n}} = V_{cap_{m,n}}(T_{mid_{m,n}}) = V_{rst} - \frac{I_{ph_{m,n}}}{C} \cdot T_{mid_{m,n}}. \tag{3.10}$$

An expression for $T_{mid_{m,n}}$ can be determined by making $V_{ctr_{m,n}} = V_{mid}$ and $t = T_{mid_{m,n}}$ in Equation (3.9), and then solving for $T_{mid_{m,n}}$. This leads to the following relation:

$$T_{mid_{m,n}} = \frac{C}{I_{ph_{m,n}}} \cdot \frac{m_{ph}}{m_c} \cdot (V_{rst} - V_{mid}) + T_s \cdot \left(1 - \frac{\overline{I_{ph}}}{I_{ph_{m,n}}} \right). \tag{3.11}$$

56

By substituting Equation (3.11) in (3.10), the capture node voltage is then obtained:

$$V_{px_{m,n}} = V_{cap_{m,n}}(T_{mid_{m,n}}) = V_{rst} - \frac{m_{ph}}{m_c} \cdot (V_{rst} - V_{mid}) - \frac{T_s}{C} \cdot \left(I_{ph_{m,n}} - \overline{I_{ph}}\right). \quad (3.12)$$

ii) $t \geq T_{max}$

If $t \geq T_{max}$, then the control node voltage has not yet reached the voltage threshold V_{mid}, but since the control signal TXS is disabled, the photocurrent integration at the capture node is ceased and its voltage does not vary anymore. Because the instant $T_{mid_{m,n}}$, at which the control node voltage reaches V_{mid}, occurs after the instant T_{max}, at which the control signal TXS is disabled, this condition can also be stated as $T_{mid_{m,n}} > T_{max}$. In this case, capture node voltage is determined by making $t = T_{max}$ in Equation (3.6):

$$V_{px_{m,n}} = V_{cap_{m,n}}(T_{max}) = V_{rst} - \frac{I_{ph_{m,n}}}{C} \cdot T_{max}. \quad (3.13)$$

Considering both cases, the final expression for the capture node voltage $V_{px_{m,n}}$ is given by:

$$V_{px_{m,n}} = \begin{cases} V_{rst} - \frac{m_{ph}}{m_c} \cdot (V_{rst} - V_{mid}) - \frac{T_s}{C} \cdot \left(I_{ph_{m,n}} - \overline{I_{ph}}\right), & t = T_{mid_{m,n}} \\ V_{rst} - \frac{I_{ph_{m,n}}}{C} \cdot T_{max}, & t = T_{max}. \end{cases} \quad (3.14)$$

These voltage values still need to be converted to tone-mapped pixel values. High light intensities generate higher photocurrent values in the photodiodes and, since the photocurrent is higher, the capture node discharges faster, yielding lower voltage values. Bright pixels are then associated with lower voltage values, while dark pixels are associated with higher voltage values. Since the highest possible voltage value for the capture (and control) node is V_{rst} and assuming the pixel values are given in the [0,1] range, the expression for the tone-mapped pixel value is determined by:

$$p_{TM} = \frac{V_{rst} - V_{cap_{m,n}}(T_{int_{m,n}})}{V_{rst}} = 1 - \frac{V_{cap_{m,n}}(T_{int_{m,n}})}{V_{rst}}, \quad (3.15)$$

where $T_{int_{m,n}}$ is the integration time of the pixel. Since $T_{int_{m,n}} = \min(T_{mid_{m,n}}, T_{max})$, the resulting tone-mapped pixel is expressed as follows:
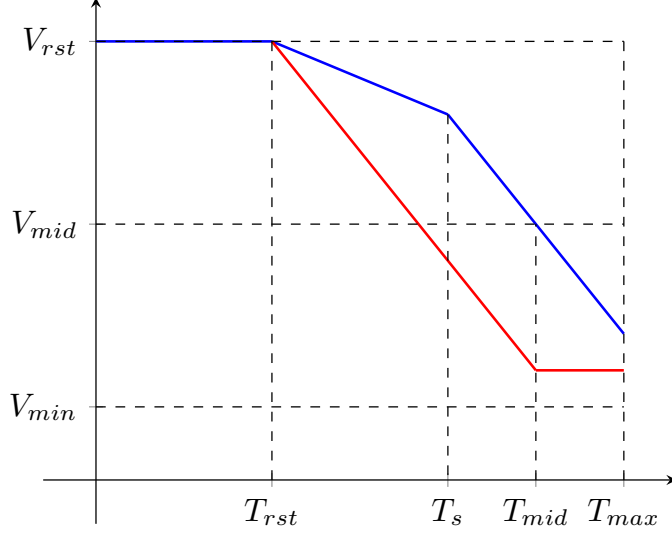
Figure 3.3: Voltage waveforms of the control and capture nodes of an arbitrary pixel of the circuit. Blue: control node voltage. Red: capture node voltage.

$$p_{TM} = \begin{cases} \frac{m_{ph}}{m_c} \cdot \left(1 - \frac{V_{mid}}{V_{rst}}\right) + \frac{T_s}{C \cdot V_{rst}} \cdot \left(I_{ph_{m,n}} - \overline{I_{ph}}\right), & T_{int_{m,n}} = T_{mid_{m,n}} \\ \frac{I_{ph_{m,n}}}{C \cdot V_{rst}} \cdot T_{max}, & T_{int_{m,n}} = T_{max}. \end{cases} \quad (3.16)$$

Figure 3.3 illustrates the evolution of the control node voltage and the capture node voltage at an arbitrary pixel of the array, based on the expressions determined in this section. The relevant voltage values and time instants are also shown.

The expressions derived so far describe the circuit operation in the actual focal plane. Since in this work digital simulations of the FPTMO are considered (whose inputs are digital raw images), it is more convenient to express the resulting equations in terms of the raw pixel value p of the digital input image, rather than photocurrents I_{ph} generated in the real circuit. A relation that estimates the photocurrents from the raw pixel values is derived as follows. The capture node voltage value $V_{px_{m,n}}$ is associated with the (digital) raw pixel value p through the expression:

$$V_{px_{m,n}} = V_{rst} \cdot (1 - p) + V_{min} \cdot p. \quad (3.17)$$

where V_{min} is the lowest possible voltage for the capture (and control) node. This relation agrees with the circuit operation, since the white pixel yields the lowest possible voltage at the capture node. The relation between the photocurrent and the voltage is the same as in a capacitor discharge relation:

$$I_{ph_{m,n}} = C \cdot \frac{\Delta V}{\Delta t} = C \cdot \frac{\left(V_{rst} - V_{px_{m,n}}\right)}{T_{int_{m,n}}}, \quad (3.18)$$

58

where $T_{int_{m,n}}$ is the integration time of the pixel, that also corresponds to the pixel exposure time. Substituting Equation (3.17) in (3.18) and establishing $V_{min} = 0$ V, a relation between the photocurrent and the raw pixel value is obtained as follows:

$$I_{ph_{m,n}} = \frac{C \cdot V_{rst}}{T_{int_{m,n}}} \cdot p = k \cdot p. \tag{3.19}$$

The constant k is the maximum photocurrent that can be generated in the sensor array (and is, therefore, associated with the white pixel). Using the estimated values of $C = 20$ fF (arbitrarily chosen to assure that the estimated photocurrent values are in a reasonable range of tens or hundreds of picoamperes), $V_{rst} = 2.7$ V (assuming $V_{DD} = 3.3$ V and $V_{th} = 0.6$ V) and $T_{int_{m,n}} = 1$ ms, the value of this constant is determined as $k = 54$ pA. The $T_{mid_{m,n}}$ and p_{TM} variables are then expressed as functions of the raw pixel value p by substituting Equation (3.19) in (3.16) and (3.11), respectively. In the resulting equations, $\bar{p}$ is the average pixel value of the digital raw image.

$$T_{mid_{m,n}} = \frac{C}{k \cdot p} \cdot \frac{m_{ph}}{m_c} \cdot (V_{rst} - V_{mid}) + T_s \cdot \left(1 - \frac{\bar{p}}{p}\right). \tag{3.20}$$

$$p_{TM} = \begin{cases} \frac{m_{ph}}{m_c} \cdot \left(1 - \frac{V_{mid}}{V_{rst}}\right) + \frac{T_s}{C \cdot V_{rst}} \cdot k \cdot (p - \bar{p}), & T_{int_{m,n}} = T_{mid_{m,n}} \\ \frac{k \cdot p}{C \cdot V_{rst}} \cdot T_{max}, & T_{int_{m,n}} = T_{max}. \end{cases} \tag{3.21}$$

If the expressions in Equation (3.21) result in a tone-mapped pixel value p_{TM} outside the [0,1] range, such value is clamped to 0, if $p_{TM} < 0$, or 1, if $p_{TM} > 1$.

3.2 Operator Parameters

From Equation 3.21, the tone-mapped pixel value depends on component values and time-domain waveforms. Consequently, perceptual attributes of the resulting images, such as brightness and contrast, are directly affected by them. Since the digital tone-mapping operators are usually defined by parameters that influence some perceptual attributes of the images, six parameters are set for the FPTMO digital version: V_{mid}, T_{max}, T_s, C, m_{ph} and m_c.

The user has access to three of these parameters: T_{max} and T_s (through the control signals TXS and GL_EN), and V_{mid} (through an external control). With these parameters, the user is able to control the overall brightness and contrast of the scene. Since users do not have an access to the circuit internal devices, they cannot adjust the remaining three parameters, namely m_{ph}, m_c and C. Once the circuit is manufactured, the parameters m_{ph}, m_c and C have their values fixed by

(a) $V_{mid} = 0.85$ V. (b) $V_{mid} = 1.35$ V. (c) $V_{mid} = 1.85$ V.
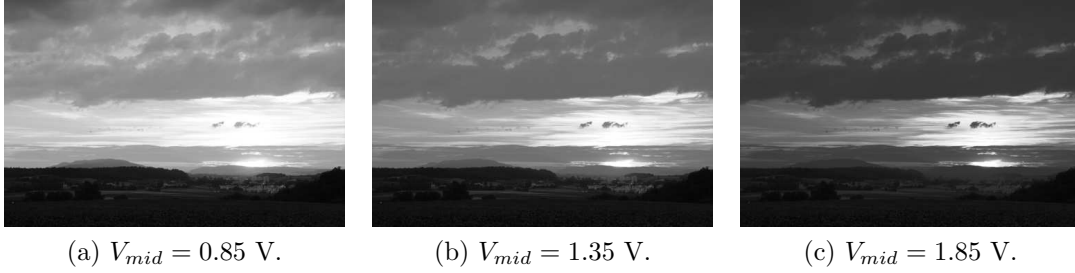
Figure 3.4: Differences on the perceived brightness of the images caused by different values of the V_{mid} parameter.

the circuit designer. The designer must define the relative sizes of the capture and control photodiodes. The designer may add extra capacitance to the control node or to the capture node (or to both), if particular capacitance values are required at those nodes. These decisions set the values of m_{ph}, m_c and C. Each parameter (and its influence on perceptual attributes of the images) is discussed in more detail next.

- Parameters defined by the user:

  i) Voltage V_{mid}

  The parameter V_{mid} is a voltage value that, once reached by the control node voltage of the pixel $V_{ctr_{m,n}}$, stops the photocurrent integration at the corresponding capture node, thereby making its voltage ready for readout. In order to achieve a reasonable dynamic range usage, this voltage threshold is typically mapped into the pixel value which corresponds to the middle of the dynamic range. By doing so, pixel values above or below the voltage threshold are represented equally. In this case, the voltage threshold is given by:

  $$V_{mid} = \frac{V_{rst} + V_{min}}{2}. \qquad (3.22)$$

  As the voltage threshold V_{mid} is set closer to V_{rst}, the resulting images get darker, because the control node voltage reaches the voltage threshold faster, which then decreases the duration of the photocurrent integration at the corresponding capture node (see Figure 3.3). On the contrary, if the voltage threshold V_{mid} is set closer to V_{min}, then brighter images are produced. Figure 3.4 shows the effects on the perceived brightness of the image for different values of the parameter V_{mid}, and Figure 3.5 shows the operator tone-mapping curves for different values of the parameter V_{mid}.

  ii) Time interval T_{max}
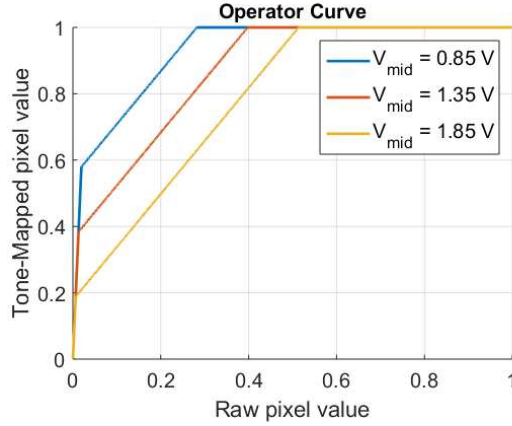
Figure 3.5: Operator tone-mapping curves for different values of the parameter V_{mid}.

The parameter T_{max} is the maximum integration time. Pixels whose control nodes voltages do not reach the voltage threshold V_{mid} have the photocurrent integration at the corresponding capture nodes interrupted at $t = T_{max}$. As T_{max} decreases, fewer pixels have their control node voltages reach V_{mid}, as less time is given for the control node to discharge. Consequently, for such pixels, the photocurrent integration at the respective capture nodes is stopped earlier, which then yields darker pixels.

Once the control node voltage of a pixel reaches V_{mid}, the pixel value remains unaffected by increasing values of T_{max}. Hence, an increase in T_{max} only affects the darker pixels of the image. Figure 3.6 illustrates this effect on the operator tone-mapping curves. Only the slope of the first part of the curve, that corresponds to pixels whose $T_{mid} > T_{max}$, is affected by the different values of T_{max}. The slope of the second part of the curve remains the same.

Because more time is given for the photocurrent integration in dark pixels as T_{max} increases, the resulting capture node voltages are closer to V_{mid}. Since V_{mid} is usually set to the middle of the dynamic range (which maps to pixel values that correspond to gray), dark regions of the image tend to become gray for high values of T_{max}, while other regions remain unaffected. The net effect corresponds to a decrease in the overall contrast of the scene, as shown in Figure 3.7.

iii) Time interval T_s

The parameter T_s determines how long the control signal GL_EN remains active or, alternatively, how long the control nodes of all pixels remain connected. During the T_s time interval, the voltages of all control nodes evolve in the same manner, according to the average photocurrent of the
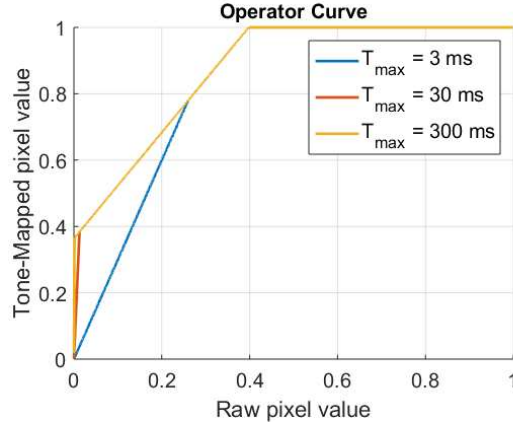
Figure 3.6: Operator tone-mapping curves for different values of the parameter T_{max}.
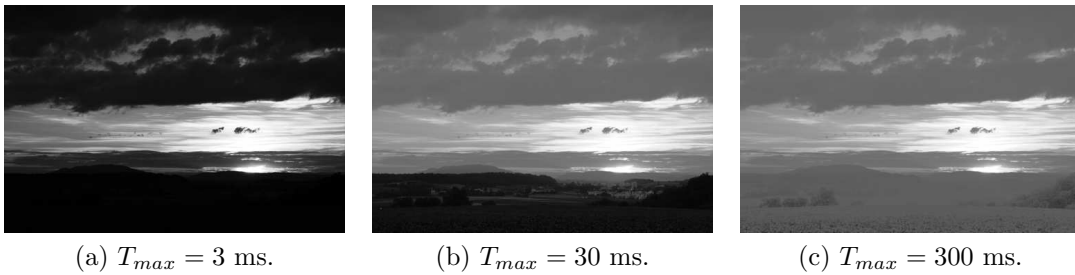
(a) $T_{max} = 3$ ms.   (b) $T_{max} = 30$ ms.   (c) $T_{max} = 300$ ms.

Figure 3.7: Resulting images generated by different values of the parameter T_{max}.

scene. After this time, the voltage of each control node changes according to the local photocurrent. This parameter defines the relative importance of the global information (represented by the average photocurrent) and the local information on the control node discharge of every pixel. Because the control node discharge is influenced by T_s, the parameter also affects the integration time of the pixels.

Low values of T_s make the control nodes connected for a short period of time and, thus, the global information has less influence on the discharge of the control nodes. In such cases, the voltage curves of both the capture and control nodes are more similar. When the control node voltage reaches V_{mid}, the capture node voltage is also close to V_{mid}. As a consequence, the resulting image tends to look dark and gray and to have low contrast.

High values of T_s cause all pixel control node voltages to evolve identically for a longer period of time, according to the average photocurrent of the sensor array. This affects the integration time of the pixels differently: the integration time is increased for pixels whose local photocurrent value is greater than the average photocurrent value, and it is decreased for pixels whose local photocurrent value is less than the average photocur-
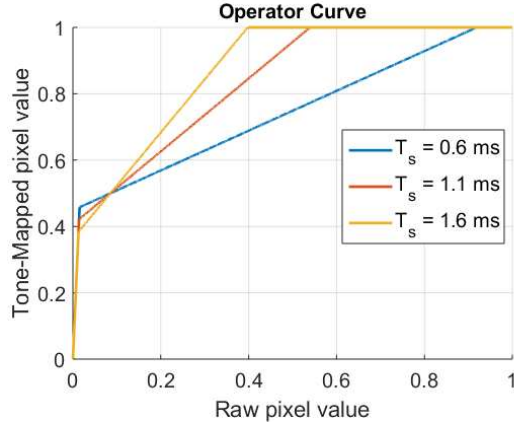
Figure 3.8: Operator tone-mapping curves for different values of the parameter T_s. The crossing point of the three curves, mapped to 0.5, corresponds to the input image average pixel value (which is 0.0854).

(a) $T_s = 0.6$ ms. (b) $T_s = 1.1$ ms. (c) $T_s = 1.6$ ms.
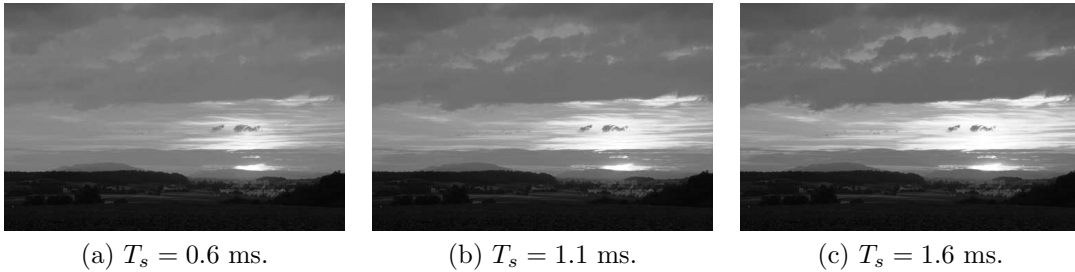
Figure 3.9: Resulting images generated by different values of the parameter T_s.

rent value. In this case, bright regions become brighter and dark regions become darker, thus enhancing the contrast of the resulting images. This effect is illustrated in Figure 3.8, which shows the corresponding operator tone-mapping curves. Raw pixel values above the crossing point are mapped to higher values, while raw pixel values below the crossing point are mapped to lower values as T_s increases. Figure 3.9 shows the resulting images for different values of T_s.

The value of the parameter T_s cannot be indefinitely high. An initial upper bound for this parameter is the parameter T_{max}, since it is the maximum integration time. If T_s is high enough, then all control node voltages reach the voltage threshold V_{mid} at the same time T_{mid}. The discharge of the control nodes is thus entirely controlled by the sensor array average photocurrent, with $T_{mid} < T_{max}$. The T_s value for this case can be found from Equation (3.5), by making $V_{ctr_{m,n}} = V_{mid}$ and solving for t (which is, then, $t = T_{s,th}$):

$$T_{s,th} = \frac{m_{ph}}{m_c} \cdot \frac{C}{\overline{I_{ph}}} \cdot \left(V_{rst} - V_{mid}\right). \qquad (3.23)$$

63

The upper bound for the value of the parameter is then $T_{s,max} = \min(T_{s,th}, T_{max})$. In the limit case $T_s = T_{s,max}$, the operator becomes fully linear and, consequently, details in dark and/or bright areas are lost.

- Parameters defined by the circuit designer:

  i) Capacitance C

  The parameter C is the capture node overall capacitance. The overall capacitance accounts for the photodiode model capacitance and the parasitic capacitances of the node. Figure 3.10 shows the resulting images for different values of C. Higher C values make node discharge (either capture or control) slower at every pixel. Because the control node discharge is slower, the integration time is increased for all pixels (which agrees with Equation (3.20)).

  For pixels whose integration time $T_{mid_{m,n}}$ is fixed at T_{max} (because $T_{mid_{m,n}} \geq T_{max}$), an increase in the value of C makes such pixels darker, because of the slower discharge of the control node. Hence, increasing values of C tends to decrease the overall brightness of the image, since more pixels have their integration times fixed at T_{max}. This tendency is noticed at the bottom part of the resulting images shown in Figure 3.10. Figure 3.11 shows a map of the integration times of every pixel in each corresponding image from Figure 3.10. The integration time of the pixels located at the bottom part is already fixed at T_{max} and, consequently, higher values of C make that region darker. Details present in this region, like the village buildings at the image bottom, are eventually no longer visible.

  For pixels whose integration time is not fixed at T_{max} (that is, $T_{mid_{m,n}} < T_{max}$), the increase in C only leads to darker pixels if the raw pixel value is above the average raw pixel value of the image (or, equivalently, if the generated photocurrent of the pixel is greater than the average photocurrent of the circuit). This is observed in the region which is delimited by the upper and bottom clouds in the sky, located at the central part of the images shown in Figure 3.10. Pixels in this region become darker as C increases. If the raw pixel value is below the average raw pixel value, than higher values of C yield brighter pixels. This is the case, for example, for parts of the clouds that are dark in Figure 3.10a and become brighter when C increases in Figure 3.10b. Figure 3.12 illustrates the effect of increasing values of the capacitance C in the tone-mapping curves.

(a) $C = 10$ fF. (b) $C = 20$ fF. (c) $C = 40$ fF.
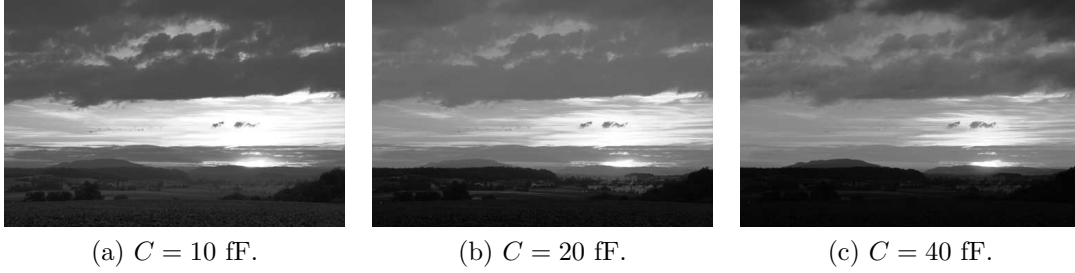
Figure 3.10: Resulting images generated by different capacitance C.

(a) $C = 10$ fF. (b) $C = 20$ fF. (c) $C = 40$ fF.
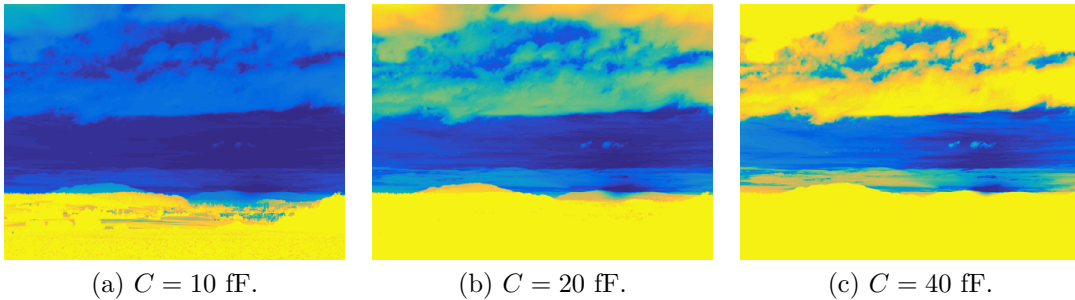
Figure 3.11: Integration times of the pixels from each image shown in Figure 3.10. The darkest blue color corresponds to the lowest integration time, while the yellow color corresponds to the maximum integration time (that is, $T_{mid_{m,n}} = T_{max}$).
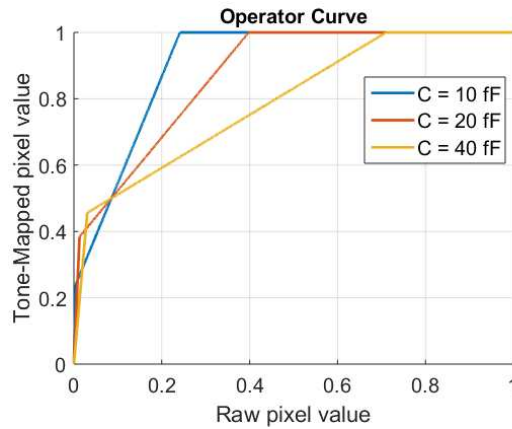
Figure 3.12: Operator tone-mapping curves for different values of the parameter C. The crossing point of the three curves, mapped to 0.5, corresponds to the average pixel value of the input image (which is 0.0854).

ii) Ratios m_{ph} and m_c

The capture and control photodiodes may have different sizes, which lead to possibly different sensitivities: the same light intensity generates different photocurrents in the capture and control nodes. The overall capacitances in both nodes may also be different, since each node has its own photodiode model capacitance and parasitic capacitances associated with it. These differences are represented by parameters m_{ph} and m_c.

65

(a) $r = 2$.   (b) $r = 1$.   (c) $r = 0.5$.

Figure 3.13: Resulting images generated by different values of the ratio $r = m_{ph}/m_c$.

The parameter m_{ph}, defined as the ratio between the generated photo-current values at the capture and the control node, takes into account the effects caused by any differences between the sensitivity of the photo-diodes from each node. The parameter m_c, defined as the ratio between the overall capacitances at the capture and the control node, models the effects caused by the different capacitances present in each node.

The node (either capture or control) discharge rate depends on the node photocurrent and capacitance values. The parameters m_{ph} and m_c have complementary effects on the node discharge rate: if the control node generates half the photocurrent of the capture node ($m_{ph} = 2$), but also has half the capacitance value of the capture node ($m_c = 2$), then the discharge rate is the same as if both nodes have equal photocurrents and capacitances ($m_{ph} = m_c = 1$). Since one parameter compensates the other, instead of analyzing their effects separately, defining a ratio between them is more convenient. The ratio $r = m_{ph}/m_c$ is the discharge ratio between the capture and the control node. It is also present in Equation (3.21). If $r > 1$, then the capture node discharge is faster than the control node discharge, thus yielding brighter pixels. On the contrary, darker pixels are produced if $r < 1$, since the capture node discharge is slower than the control node discharge. Figure 3.13 shows the resulting images and Figure 3.14 shows the tone-mapping curves for different values of the ratio r.

Besides the parameters discussed, there is one more variable that affects the circuit operation. As previously mentioned, the operator adjusts the pixel value based on global and local information. This means that the operator adapts itself according to global information such as the average luminance value. Thus, for a set of fixed parameter values, the operator yields different pixel values depending on the average luminance (represented by the average photocurrent $\overline{I_{ph}}$ in Equation (3.16) or, equivalently, by the average raw pixel value $\overline{p}$ in Equation (3.21)). The average luminance is not regarded as an FPTMO parameter, because it is not an FPTMO
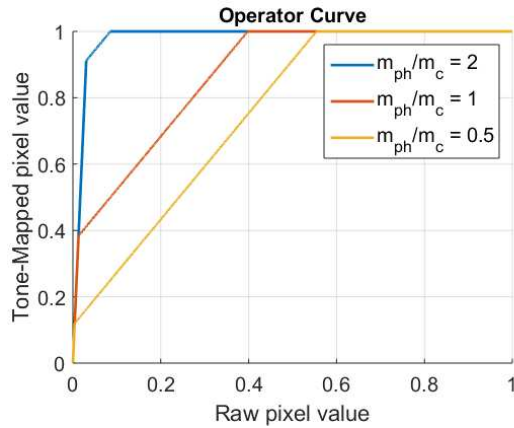
Figure 3.14: Operator tone-mapping curves for different values of the ratio $r = m_{ph}/m_c$.
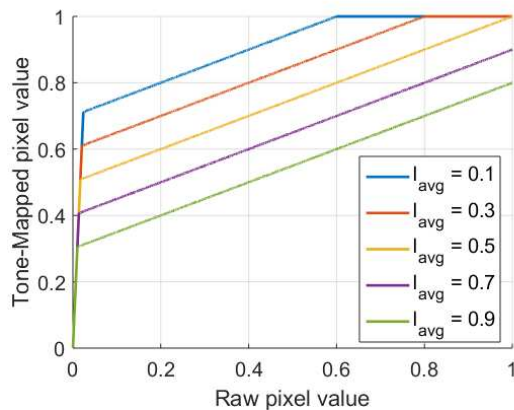
Figure 3.15: Resulting operator tone-mapping curves for scenes with different average luminance value. I_{avg} represents the average raw pixel value of the input image.

circuit feature, but rather an intrinsic attribute of the scene. Figure 3.15 shows the operator tone-mapping curves for different average raw pixel values. For dark scenes (with low average value), the operator has a tendency to increase the pixel values, which brightens dark regions of the image and makes details more visible. For bright scenes (with high average value), the operator does the opposite: it decreases the pixel values, in order to not saturate large regions of the image and lose details.

3.3 Color Treatment

Up to this point, the tone-mapped images resulting from the FPTMO were achromatic (i.e. grayscale), since each photodiode registers a single value that represents locally incident light intensity. Color reconstruction of the captured scene is not possible, because the sensors must record at least three values, representing average intensities within different wavelength intervals of incident light spectral distribu-

tion. As previously mentioned, the human visual system has three cone types, each with a different spectral sensitivity. Color perception is based on a vector of three values. Each component corresponds to the intensity of the incident light filtered by one of the cones.

In order to include color information in the captured images, different solutions have been proposed. Foveon Inc., a company that manufactures and distributes imaging sensor technology, has developed one possible solution. This company designed an image sensor called *Foveon X3* that uses three vertically stacked photodiodes per pixel [30]. Each photodiode responds to different light wavelength ranges, by taking advantage of the fact that each wavelength penetrates differently in the silicon (as explained in Chapter 2). For imaging sensors that use this technology, the output images are already colored (represented in the RGB-space), without requiring any further processing. To generate color images, most digital cameras use another solution, that has been proposed before Foveon X3, and is discussed next.

3.3.1   Bayer Filter Mosaic

The most widely adopted solution for color reconstruction in imaging sensors is known as *Bayer filter mosaic* (or *Bayer pattern array*), proposed by the american scientist Bryce Bayer in 1976 [31]. The Bayer filter array consists in a layer of three different color filters, each one with a particular spectral sensitivity (corresponding to the red, green or blue range of the visible light spectrum), placed atop the sensor array. The color filters are arranged in a repeated interleaved pattern along the rows and columns of the sensor array and there are as many green filters as red and blue filters together. This arrangement is also based on another aspect of the human visual system. The human eye is more sensitive to luminance distortions than to chrominance distortions. This fact is illustrated in Figure 3.16, which shows images generated after applying a Gaussian filter only on the luminance and only on the chrominances. The luminous efficiency curve, which defines how the human eye perceives luminance, changes depending on the overall illumination conditions.

Under photopic conditions (i.e. highly illuminated scenes), which is when the cones are more sensitive to light variations than the rods, the green cone spectral sensitivity curve is more similar to the luminous efficiency curve than the red and blue cone curves are [16], indicating that the human eye has a greater sensitivity to green color variations than red and blue color variations. This also relates to the larger weight that is given to the green color channel in the calculation of the luminance in Equation (2.5) when compared to the red and blue channels. Because of this, in the Bayer mosaic array, more pixels represent the green color channel than the red or blue channels. The sensors associated with the green color channel

(a) Original image.   (b) Only luminance filtered.   (c) Only chrominances filtered.
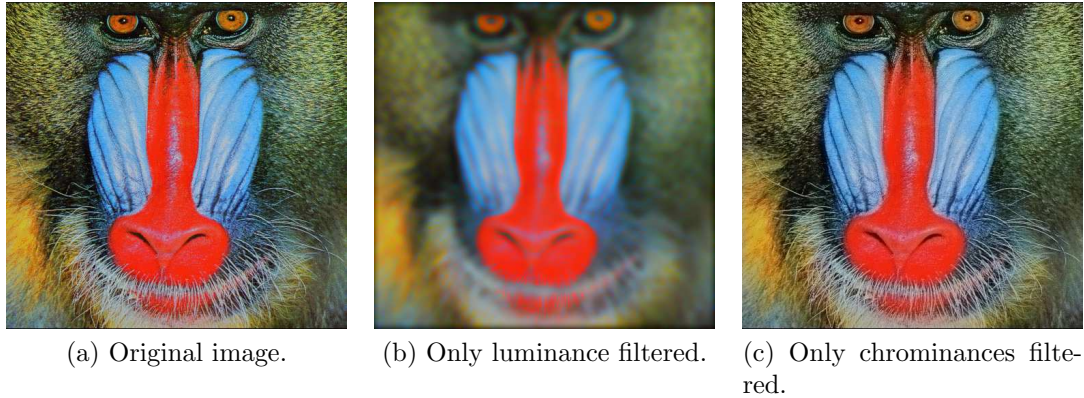
Figure 3.16: Resulting mandrill images after applying a Gaussian filter on the luminance, while preserving the chrominances, and on the chrominances, while preserving the luminance. The Gaussian kernel size used is 20×20, with a standard deviation of 5.
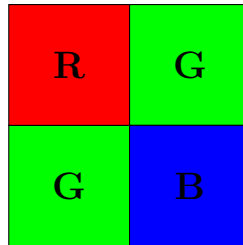
Figure 3.17: A typical Bayer pattern sensor arrangement (RGGB) for a 2×2 block. This pattern is repeated along the entire sensor array.

are also referred to as *luminance-sensing elements*, while sensors associated with the red and blue channels are referred to as *chrominance-sensing elements*. Figure 3.17 shows a 2×2 sensor block arranged in a typical Bayer pattern.

3.3.2   Demosaicing

With the addition of the Bayer pattern Color Filter Array (CFA), each pixel records one value that represents the intensity of one color channel at a certain position in the sensor array. Some further processing must still be done in the Bayer pattern image, in order to generate a color image in the RGB space, since each pixel is missing the information from two color channels. The operation that reconstructs the missing color channel values at each pixel of the sensor array is called *demosaicing* [1] (or *demosaicking*). In order to keep the size and complexity of the focal-plane circuit low, the demosaicing operation is carried out off the focal plane.

---

[1]The pixel values of the color image obtained after the demosaicing operation correspond to coordinates in a color space that is different from conventional display device color space. To make sure that a given color in the image obtained after demosaicing is represented as the same color in a display device, the pixel values must be converted to appropriated coordinates in the color space of the display device. This conversion is covered in Appendix A.

Many demosaicing methods exist in the literature [32]. One approach is to apply a bilinear filter to the Bayer pattern image, but this also introduces color artifacts in the resulting images. More complex solutions have been proposed to overcome this limitation and generate better quality color images. The reader is referred to [32] and [33] for some different developed solutions for the demosaicing problem and their particularities. Besides the demosaicing, another color processing operation required is the *white-balance* operation, which is discussed in more detail later, in Section 3.3.4.

3.3.3 FPTMO Color Processing

The introduction of the Bayer pattern CFA also allows for new circuit configurations to be explored, each one with a different operating principle that takes into account the new color information in the sensor array, in order to perform the tone mapping. In this section, three FPTMO digital implementations, simulating different operating principles, that yield color images are considered and explained. In this section, the white-balance operation is assumed to be performed before the tone-mapping. In a real focal-plane implementation, white balance is performed after tone mapping. The problems associated with this change in the processing order are addressed in Section 3.3.4.

The first digital implementation is a straightforward extension of the original signal-processing chain, which basically consists in two stages: the HDR image capture and the tone mapping. In this version, additional processing stages, consisting of the white-balance and demosaicing operations, take place after tone mapping. The circuit operating principle is not modified, that is, the tone-mapping operation does not make any special use of the color information. The average raw pixel value of the input image (which is later used to determine each pixel integration time and the corresponding tone-mapped value, as in Equations (3.20) and (3.21), respectively) is calculated considering all pixels of the image. In the circuit, this is equivalent to using the average photocurrent value, computed over the entire sensor array during $t < T_s$, to determine the discharge of every control node. Figure 3.18 shows the color images resulting from this version of the operator.

The second digital implementation is based on the idea of performing the tone-mapping operation on the luminance values and then scaling the original color channels by the ratio between the tone-mapped and original luminance values. This version operation principle is defined as follows: first, before tone mapping, a copy of the input raw image, which is in a particular Bayer pattern, is stored in a buffer. This copy is later used to determine the original (raw) color channel values at each pixel. Then, the tone-mapping operation is performed only on the input raw pixels

(a) Parasol.   (b) Sunset.   (c) Color Checker.

Figure 3.18: Results from the FPTMO first digital version, which uses the average raw pixel value of the input image to perform the tone mapping on all pixels, regardless of the color channel that each pixel represents.

(a) Parasol.   (b) Sunset.   (c) Color Checker.

Figure 3.19: Results from the FPTMO second digital version, which performs tone mapping only on sensor array green pixels (using their average photogenerated current). This operator then scales the original red and blue channels by the ratio between pixels in the (full) tone-mapped and original green color channels.

that correspond to the green channel. The green channel provides a better approximation to the luminance channel than the red and blue channels individually (i.e. in the luminance formula defined in Equation (2.5), the green channel weight is higher than the red and blue channel weights individually).

After the tone-mapping operation, tone-mapped green channel values are still missing at positions that correspond to red and blue pixels and therefore must be determined. These values are calculated through an interpolation of the computed tone-mapped green pixel values that are closest to each red and blue pixel. This interpolation yields the full tone-mapped green channel. The full tone-mapped red and blue channels are scaled versions of the corresponding original color channels, in which the scale factor is given by the ratio between the tone-mapped and original green channels. In order to obtain the original red, green and blue color channel values, a demosaicing operation is applied to the stored copy of the input raw image. The result is a raw RGB image, in which the each color channel value is defined at each pixel. Once the original color channel values are determined, the full tone-mapped red and blue channels are then obtained by multiplying the original red and blue channels by the corresponding scale factors at each pixel. The images resulting from this implementation are shown in Figure 3.19.

In a real focal-plane implementation, this version requires less interconnections between the pixels of the sensor array during $t < T_s$, since only green pixels must remain connected. This version does not require a control photodiode for red and blue pixels, because only green pixels are tone-mapped in the focal plane. This leads to imaging sensor area reduction. On the other hand, this version requires two scene captures: one for the storage of the raw values, and another for the actual tone mapping of the green pixels. The analog-to-digital conversion performed after each capture is also different: for the tone-mapped green pixels, conversion uses eight bits, while for the entire raw scene, it uses a higher number of bits (typically twelve or fourteen bits). This leads to processing time increase, which makes this implementation in hardware less appealing, despite the overall quality of the generated images (the colors of the images resulting from different FPTMO versions and other digital tone-mapping operators are discussed in more detail in Chapter 4). A different version of the operator, which incorporates some ideas from this implementation and generates similar results is discussed later in this chapter.

The third digital implementation consists of performing the tone-mapping operation independently on each color channel. In this case, the tone mapping occurs three times. At each time, only pixels of a particular color channel are considered, in order to determine the tone-mapping curve. As an example, consider the red color channel. The integration time and tone-mapped value of the pixels representing this color channel in the Bayer pattern image are determined by using the average value of pixels representing the red color channel in the raw image. The operating principle for the green and blue channels is similar. After tone mapping, the demosaicing operation is carried out, yielding the color image. In the circuit, the interconnections between pixels during $t < T_s$ are different, since only pixels representing the same color channel must be connected to each other. Two extra nodes are needed for reading the average photocurrent of the other color channels. These requirements lead to an increase in circuit complexity. The resulting images have a minor blue color cast (i.e. a blue tint), as shown in Figure 3.20. Because this implementation increases the circuit complexity and image quality is similar to that of the images obtained by the first implementation, which was discussed earlier in this section (and which requires a simpler circuit), the chip using this implementation is also not considered for fabrication. As such, problems associated with this implementation, like the blue color cast in the resulting images, were not further studied.

| (a) Parasol. | (b) Sunset. | (c) Color Checker. |

Figure 3.20: Results from the FPTMO third digital version, which performs the tone mapping on each channel individually, using their corresponding average pixel value.

| (a) With white balance. | (b) Without white balance. |

Figure 3.21: FPTMO first version results obtained with and without white balance. The same set of parameters is used for both images.

3.3.4 White Balance

Unlike the human visual system, the sensors of a camera do not possess the capability of automatically adapting themselves to the color of the light source illuminating a scene. They simply record the incident light intensity, without correcting their values to take into account any possible color deviations in the scene caused by the illuminant color. Moreover, the sensors also have different sensitivities to different light wavelengths, which are determined by the quantum efficiency curves of the sensor (as explained in Chapter 2) and by the color filter array sensitivities. Since the recorded sensor value is influenced by these effects (namely, the light source color and the different wavelength sensitivities), additional signal processing is required to correct these values and avoid color distortion in the output images. The color processing task that handles this problem is known as *white balance*. To illustrate the importance of such operation, Figure 3.21 shows a comparison between two images obtained from the FPTMO first version. In Figure 3.21b, no white balance is performed, which leads to a severe color distortion in the whole image. When the white balance is applied, such as in Figure 3.21a, this color distortion is not observed.

The white-balance operation consists of multiplying each color channel by a dif-

ferent constant gain. Because image colors are defined according to the proportions between the color channels, rather than their absolute values, a channel is chosen as the reference. The gains of the remaining color channels are then given relative to the reference channel. In this work, the green channel is the reference and, consequently, only the red and blue channels are affected by the white-balance operation. The choice for the green channel as the reference has two explanations. First, the green channel typically contains most of the information about the scene. Second, the green channel provides a better approximation to the luminance values than the red and blue channels separately. Since the tone-mapping operation affects the luminance values, it is desired that the effects of the white-balance operation in the luminance values of the tone-mapped scene be minimal. In that way, modifications in the luminance (and also in the overall quality) of the image are mainly caused by the tone-mapping operation.

Different algorithms for white-balance gain computation are available. Digital cameras typically have predefined white-balance profiles, in which the gains of each channel are already determined for different illuminants, such as candlelight, fluorescent lamps or daylight. Besides the predefined profiles, a "manual white balance" option is available, in which the user takes the picture of a white reference and indicates to the camera that such reference should be used to determine the gains, rather than using one of the predefined profiles. Some cameras also include an "auto white balance" option. In this option, for each image, an algorithm determines the gains after automatically finding white points in the image. The white-balance gains used in each image shown in this work were obtained through the *DCRaw* software [34]. The software calculates the gains based on the camera model and the corresponding white-balance option used.

The white-balance operation typically occurs right before or after the demosaicing operation. Figure 3.22 shows the usual cascade of image processing stages that apply when the tone mapping is performed by digital operators. The input of the digital operators are color raw images (RGB), rather than raw images in the Bayer pattern. As such, white balance and demosaicing are performed before tone mapping. For an FPTMO simulation, the order of operations is different. Besides tone mapping, no additional operations are carried out in the focal plane, in order to keep minimal circuit size and complexity, as stated in Section 3.3.2. In this case, the white-balance and demosaicing operations occur after the tone mapping. Figure 3.23 shows a sequence of image processing stages that are used in a digital simulation of the FPTMO. Some white-balance algorithms, considering such chain of operations, are discussed next in detail.
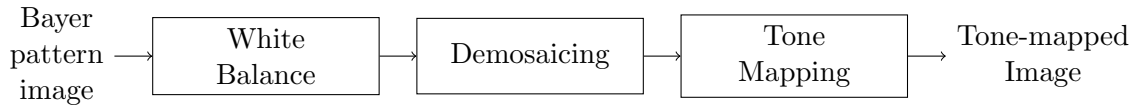
Figure 3.22: Cascade of image processing stages supporting fully digital tone-mapping.
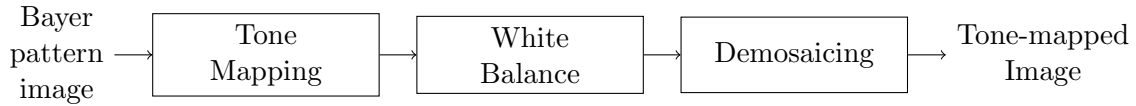
Figure 3.23: Image processing stages in a digital simulation of the FPTMO.

(a) Before demosaicing   (b) After demosaicing
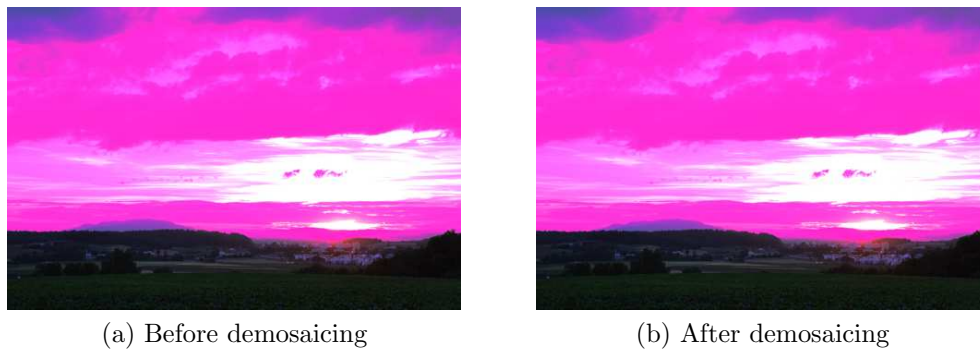
Figure 3.24: Images resulting from the multiplication of the red and blue tone-mapped pixels by the white-balance gains obtained from the DCRaw software. In (a) and (b), the white-balance gains are applied before and after the demosaicing, respectively.

White-Balance Algorithms

The white-balance gains obtained from DCRaw software are calculated according to the raw image pixel values. These gains should be applied to the sensor values before any (non-linear) processing is performed on them. In the FPTMO approach, white balance is carried out after tone mapping (which is a non-linear function). The DCRaw white-balance gains might then not be suitable for correcting the colors of the tone-mapped image. This is confirmed in Figure 3.24, which shows the images obtained after multiplying the color channels of the tone-mapped image by such gains. Applying the gains before or after the demosaicing does not produce any difference on the resulting images color.

After verifying that the color distortion problem cannot be solved by multiplying the FPTMO output color channels by the DCRaw gains, other white-balance algorithms were tested. The first white-balance algorithm tested is called *Gray World*. This algorithm is based on the assumption that, on average, the reflectance of the scene is achromatic [35], which means that the three color channels have the same average value. To make the average pixel values equal for all three color channels,

the average pixel value of each channel is calculated and one channel is chosen as reference. In this work, the reference channel is the green channel. Then, to achieve the same average pixel value as the green color channel, the red and blue color channels are multiplied by gains that are equal to the ratio between the average green and red pixel values and between the average blue and green pixel values, respectively. This transformation is summarized in Equations (3.24) and (3.25) (assuming an M $\times$ N image size).

$$
\begin{cases}
R_{avg} = \frac{1}{MN} \cdot \sum_{n=1}^{N} \sum_{m=1}^{M} R(m,n) \\
G_{avg} = \frac{1}{MN} \cdot \sum_{n=1}^{N} \sum_{m=1}^{M} G(m,n) \\
B_{avg} = \frac{1}{MN} \cdot \sum_{n=1}^{N} \sum_{m=1}^{M} B(m,n),
\end{cases}
\tag{3.24}
$$

$$
\begin{cases}
R_{corrected}(m,n) = \frac{G_{avg}}{R_{avg}} \cdot R(m,n) \\
G_{corrected}(m,n) = G(m,n) \\
B_{corrected}(m,n) = \frac{G_{avg}}{B_{avg}} \cdot B(m,n).
\end{cases}
\tag{3.25}
$$

Figure 3.25 shows the resulting image after the application of the Gray World algorithm. This algorithm is executed after the demosaicing operation, because it assumes that the input image is RGB, in which each pixel has defined values for the three color channels. For input images in the Bayer pattern (i.e. before demosaicing), a slightly modified version of the algorithm is required, in which each color channel average pixel value is calculated considering only the pixels representing the same color channel in the Bayer pattern array. In this modified version, the average pixel value of each color channel can be computed in Equation (3.24) by using either the raw pixel values or the tone-mapped pixel values. Two additional implementations of the Gray World algorithm are considered. The images resulting from each implementation are shown in Figure 3.26. None of the implementations of this algorithm yield appropriate white-balance gains that correct the color distortion present in the tone-mapped image.

The second white-balance algorithm tested is called *White Patch*. It is based on the assumption that the perception of white by the human visual system is associated with the maximum cone signals [35]. This algorithm is similar to the Gray World algorithm, in that it equalizes the maximum pixel value of each channel, rather than their average pixel values. The maximum pixel value of each color channel is calculated and the green channel is chosen as reference. Then, the white-balance gains for the red and blue color channels are given by the ratio between the maximum green and red pixel values and between the maximum green and blue pixel values, respectively. These steps are summarized in Equations (3.26) and (3.27).

(a) Input image.

(b) Resulting image.

Figure 3.25: Application of the Gray World white-balance algorithm to the tone-mapped input image, after the demosaicing.

(a) Using raw values.

(b) Using tone-mapped values.

Figure 3.26: Images obtained after applying the Gray World white-balance algorithm to the tone-mapped image before the demosaicing. In (a) and (b), the average values of each color channel are calculated using the raw pixel values and the tone-mapped pixel values, respectively.

$$\begin{cases} R_{max} = \max_{(m,n)} R(m,n) \\ G_{max} = \max_{(m,n)} G(m,n) \\ B_{max} = \max_{(m,n)} B(m,n), \end{cases} \tag{3.26}$$

$$\begin{cases} R_{corrected}(m,n) = \frac{G_{max}}{R_{max}} \cdot R(m,n) \\ G_{corrected}(m,n) = G(m,n) \\ B_{corrected}(m,n) = \frac{G_{max}}{B_{max}} \cdot B(m,n). \end{cases} \tag{3.27}$$

Figures 3.27 and 3.28 shows images that result from the White Patch white-balance algorithm. Like the Gray World algorithm, the White Patch algorithm is executed after the demosaicing (that is, on an RGB image), but can also be modified, in order to be executed before the demosaicing, on an image in the Bayer pattern. In this case, each color channel maximum value is taken considering only pixels in the Bayer pattern image that represent the same color channel. The maximum value of

(a) Input image.   (b) Resulting image.

Figure 3.27: (a) Digital FPTMO output image, after demosaicing, which is an input for the White Patch white-balance algorithm; (b) White Patch algorithm result.

(a) Using raw values.   (b) Using tone-mapped values.

Figure 3.28: Application of the White Patch algorithm to the digital FPTMO output image, before demosaicing: (a) white-balance gain computation based on maximum raw pixel values at each color channel; (b) white-balance gain computation based on maximum tone-mapped pixel values at each color channel.

each color channel can refer either to the maximum raw pixel value (i.e. before tone mapping) or to the maximum tone-mapped pixel value of the corresponding color channel. Both versions are considered and implemented. The white-balance gains obtained from different versions of this algorithm do not correct color distortions either.

The third white-balance algorithm tested is the one proposed by *Huo et al.* [36], which is denoted in this work as *Robust Auto White Balance* (RAWB) algorithm. In this algorithm, the white-balance gains of the red and blue channels are automatically adjusted through an iterative process. The algorithm consists of two stages: the illuminant color estimation and the white-balance gain adjustment. In the first stage, the algorithm searches for possible achromatic pixels of the image. The input image is initially converted from RGB to YUV color space. In the YUV color space, Y represents the luminance component, while $U = B - Y$ and $V = R - Y$ represent the chrominance components of the image. Under a neutral (i.e. white) illuminant, achromatic pixels have equal values for the red, green and blue color channels (R = B = G = Y), or equivalently, have the corresponding chrominance components U and

V equal to zero. Under a illuminant of different color, the chrominance components U and V of such pixels deviate from zero. In this case, in order to be considered an achromatic pixel, the values of the chrominance components must be within a given tolerance. By defining the function $F(Y,U,V)$ as the ratio between the sum of the absolute values of the chrominances and the luminance value of the pixel, an achromatic pixel then satisfies the following condition:

$$F(Y,U,V) = \frac{|U| + |V|}{Y} < T, \tag{3.28}$$

where T is a threshold value (below 1) that depends on the illuminant color [36]. Once the set of achromatic pixels is determined, the algorithm second stage takes place. In this stage, the white-balance gains of the red and blue color channels are initialized to unity and are then adjusted iteratively. At each iteration, only one white-balance gain (associated with either red or blue color channel) is modified. In order to determine which white-balance gain to modify in the current iteration, the average chrominance components U_{avg} and V_{avg} of the achromatic pixel set are calculated. If $U_{avg} > V_{avg}$, then the image color is more biased towards a blue tint than towards a red tint (since the $B_{avg} - Y$ difference is greater than the $R_{avg} - Y$ difference). The blue color channel must thus be corrected. On the contrary, if $V_{avg} > U_{avg}$, then the red color channel needs correction.

After the determination of the channel to be corrected, an error signal is defined: $\epsilon = d - \phi$, where d denotes the target value for the average chrominance components (which is zero) and $phi = \max(|U_{avg}|, |V_{avg}|)$. This error signal defines how the current white-balance gain is modified (whether it must be incremented or decremented). The gains are adjusted according to Equations (3.29) and (3.30):

$$\begin{cases} R_{g,i} = R_{g,i-1} + \mu \cdot K(\epsilon) \\ B_{g,i} = B_{g,i-1} \end{cases} \quad \text{if } \phi = V_{avg}, \tag{3.29}$$

$$\begin{cases} R_{g,i} = R_{g,i-1} \\ B_{g,i} = B_{g,i-1} + \mu \cdot K(\epsilon) \end{cases} \quad \text{if } \phi = U_{avg}. \tag{3.30}$$

In Equations (3.29) and (3.30), i denotes the current iteration, μ is the adjustment value that is applied to the current white-balance gain and $K(\epsilon)$ is a function that controls the convergence speed of the algorithm, based on the error signal. If the error is high enough (above an user-defined threshold a), then function $K(\epsilon)$ doubles the adjustment value μ, thereby accelerating the convergence process. If the error is still not low enough (below the user-defined threshold a but above the error tolerance b), then the adjustment value μ is used to update the current gain. If the error is within the tolerance b, then it is assumed that convergence has been

(a) Input image.   (b) Resulting image.

Figure 3.29: Application of the RAWB white-balance algorithm to the tone-mapped image after demosaicing.

achieved by the algorithm, indicating that the white-balance gain associated with the corresponding color channel does not need further adjustment. Equation (3.31) shows the possible values of function $K(\epsilon)$. The whole iteration lasts until the error associated with each average chrominance component is minimized (that is, until both average chrominances U_{avg} and V_{avg} converge), then establishing the final white-balance gains for the red and blue color channels.

$$
K(\epsilon) = \begin{cases} 2, & \text{if } |\epsilon| \geq a \\ 1, & \text{if } b \leq |\epsilon| < a \\ 0, & \text{if } 0 < |\epsilon| < b. \end{cases} \tag{3.31}
$$

Figure 3.29 shows the image resulting from the RAWB white-balance algorithm, along with the input tone-mapped image, without any white-balance. Because the algorithm involves a color conversion from the RGB to the YUV space, it is only executed after demosaicing. Color distortions are still visible in the image after the application of this algorithm.

The fourth white-balance procedure tested consists in applying histogram equalization independently to each color channel. In this procedure, pixels in a particular color channel are multiplied by different gains, instead of constant gains. These white-balance gains depend on the value of the corresponding color channel at each pixel. The gains are given by the cumulative distribution function (CDF) associated with the color channel histogram. Figure 3.30 shows the image after applying the histogram equalization to each color channel of the input image. Although some color artifacts can be observed, such as the light green tint in the clouds near the sun, the main color distortion present in the image has been corrected and colors of the scene are arguably natural.

(a) Input image.   (b) Resulting image.

Figure 3.30: Application of histogram equalization independently to each color channel of the tone-mapped image after the demosaicing.

FPTMO White-Balance Algorithm

The results from Section 3.3.4 suggest that the multiplication of each color channel by different constant gains is not appropriate for solving the FPTMO color distortion problem. This leads to the formulation of the following hypothesis: since, in the focal plane, each pixel (representing a particular color channel) is non-linearly transformed through the tone-mapping operation, in order to properly correct the colors of the image, the white-balance adjustments must not be made globally, based solely on global information extracted from a large group of pixels that share some common attributes (like, for example, adjusting the pixels by the average value of the color channel they represent). Instead, such adjustments must be made locally, taking into account each pixel value individually. Based on this hypothesis and motivated by the results obtained from the histogram equalization method, an alternative white-balance approach is proposed, in which the gains depend not only on the color channel a pixel represents, but also on the own pixel value.

Finding the correct constant white-balance gains for each color channel of an image, without any indication of what they should be, is a difficult task. It becomes impractical when the gains also vary according to the color channel value at each pixel, because of the large number of combinations that can be considered. To guide the calculations of the gains for each pixel, some white-balance method, which yields images without color distortions, must be used as reference. A model of operation that corrects the color distortions of the image consists in applying the constant white-balance gains obtained from DCRaw software to the sensor raw values (that is, before tone mapping takes place). Figure 3.31 shows the image resulting from this model. As stated in Section 3.3.2, this model is not considered for focal-plane implementation, because it requires a white-balance operation to be performed before tone mapping. However, because little color distortion is observed in the resulting images, it can be used as reference, providing a good starting point

81

(a) Input image.   (b) Resulting image.

Figure 3.31: Constant white-balance gains (obtained from DCRaw software) applied before tone mapping.
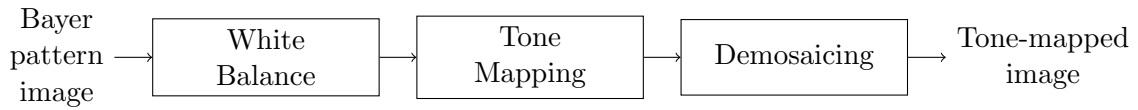
Figure 3.32: Cascade of raw image processing steps that is used as reference for the computation of the FPTMO white-balance gains.

for the calculations of the new white-balance gains.

Figure 3.32 depicts the reference model cascade of raw image processing stages. The relevant expressions for obtaining the tone-mapped pixel value of the reference model are derived next, considering an arbitrary red pixel. The expressions for an arbitrary blue pixel can also be derived using the same procedure.

Initially, a raw red pixel at position (m, n) in the sensor array $R_{m,n}$ is multiplied by the red white-balance gain obtained from the DCRaw software K_R. By making $p = K_R \cdot R_{m,n}$, the time instant $T_{mid_{m,n}}$ in Equation (3.20) can be computed as follows:

$$T_{mid_{m,n}} = \frac{1}{K_R \cdot R_{m,n}} \cdot \frac{C}{k} \cdot \frac{m_{ph}}{m_c} \cdot (V_{rst} - V_{mid}) + T_s \cdot \left(1 - \frac{\overline{I}}{K_R \cdot R_{m,n}}\right), \quad (3.32)$$

where $\overline{I}$ is the average raw pixel value. Denoting $\beta_1 = \frac{C}{k} \cdot \frac{m_{ph}}{m_c} \cdot (V_{rst} - V_{mid})$ and regrouping the terms in Equation (3.32) yields the following expression for the time instant $T_{mid_{m,n}}$:

$$T_{mid_{m,n}} = \frac{1}{K_R \cdot R_{m,n}} \cdot \left(\beta_1 - T_s \cdot \overline{I}\right) + T_s. \quad (3.33)$$

The red tone-mapped pixel $R_{TM_{m,n}}$ is then calculated according to Equation (3.15), which leads to:

$$R^{ref}_{TM_{m,n}} = 1 - \frac{V_{cap_{m,n}}(T_{int_{m,n}})}{V_{rst}} = \frac{k}{C} \cdot \frac{K_R \cdot R_{m,n} \cdot T_{int_{m,n}}}{V_{rst}}, \quad (3.34)$$

82

where $V_{cap_{m,n}}(T_{int_{m,n}})$ is the capture node voltage associated with the red raw pixel after the white balance $K_R \cdot R_{m,n}$ and $T_{int_{m,n}}$ is the corresponding pixel integration time. Equation (3.34) is obtained after substituting this voltage value by the expression from Equation (3.6) (with $I_{ph_{m,n}} = k \cdot K_R \cdot R_{m,n}$). Since $T_{int_{m,n}} = \min(T_{mid_{m,n}}, T_{max})$, there are two possible expressions for the red tone-mapped pixel $R_{TM_{m,n}}$. These are then determined by making $T_{int_{m,n}} = T_{mid_{m,n}}$ (given by Equation (3.33)) and $T_{int_{m,n}} = T_{max}$ in Equation (3.34), and defining $\beta_2 = \frac{k}{C}$:

$$
R_{TM_{m,n}}^{ref} = \begin{cases} \frac{\beta_2}{V_{rst}} \cdot \left[\beta_1 + T_s \cdot \left(K_R \cdot R_{m,n} - \overline{I} \right) \right], & T_{int_{m,n}} = T_{mid_{m,n}} \\ \frac{\beta_2}{V_{rst}} \cdot K_R \cdot R_{m,n} \cdot T_{max}, & T_{int_{m,n}} = T_{max}. \end{cases} \tag{3.35}
$$

The tone-mapped pixel values obtained from the expressions defined in Equation (3.35) (which are here given for red pixels, but are similar for blue pixels) are assumed to be the correct values, since the resulting image presents little color distortion when the demosaicing is performed using these values.

In the real focal-plane implementation, the expressions for the time instant $T_{mid_{m,n}}$ and the red tone-mapped pixel value $R_{TM_{m,n}}$ are the same as the ones derived for the reference model (Equations (3.33) and (3.35), respectively), with the exception that the raw red pixel value $R_{m,n}$ in these equations is not multiplied by the white-balance gain K_R, since no white balance is performed before tone mapping (the same applies for blue pixels). Both equations are summarized below:

$$
T_{mid_{m,n}} = \frac{1}{R_{m,n}} \cdot \left(\beta_1 - T_s \cdot \overline{I} \right) + T_s. \tag{3.36}
$$

$$
R_{TM_{m,n}} = \begin{cases} \frac{\beta_2}{V_{rst}} \cdot \left[\beta_1 + T_s \cdot \left(R_{m,n} - \overline{I} \right) \right], & T_{int_{m,n}} = T_{mid_{m,n}} \\ \frac{\beta_2}{V_{rst}} \cdot R_{m,n} \cdot T_{max}, & T_{int_{m,n}} = T_{max}. \end{cases} \tag{3.37}
$$

The goal is to find the corresponding white-balance gain K'_R that must be multiplied after the tone mapping to the red pixel $R_{TM_{m,n}}$, in order to yield the correct red tone-mapped pixel value $R_{TM_{m,n}}^{ref}$:

$$
K'_{R_{m,n}} \cdot R_{TM_{m,n}} = R_{TM_{m,n}}^{ref}. \tag{3.38}
$$

Because $R_{TM_{m,n}}$ and $R_{TM_{m,n}}^{ref}$ have each two possible expressions, different combinations satisfy Equation (3.38), depending on the integration time value $T_{int_{m,n}}$ associated with the same pixel in the reference and in the real model. At first, four possible combinations should be considered. However, since the white-balance gain

K_R is higher than unity, it can be stated from Equations (3.33) and (3.36) that, in the real focal-plane model, the integration time $T_{int_{m,n}}$ of the corresponding red pixel is never lower than that of the same red pixel in the reference model. Therefore, the case in which $T_{int_{m,n}} = T_{mid_{m,n}}$ in the real focal-plane model and $T_{int_{m,n}} = T_{max}$ in the reference model does not occur and must not be considered. This leads to a total of three possible cases. Each case is listed as follows:

- Case 1: $T_{int_{m,n}} = T_{mid_{m,n}}$ in both models

  In this case, the control node voltage of the pixel located at position (m, n) in the sensor array reaches the voltage threshold V_{mid} before the time instant T_{max} in both models (that is, $T_{mid_{m,n}} < T_{max}$ in both models). Consequently, $R_{TM_{m,n}}$ and $R_{TM_{m,n}}^{ref}$ in Equation (3.38) are substituted by the corresponding expressions given by Equations (3.37) and (3.35), respectively, when $T_{int_{m,n}} = T_{mid_{m,n}}$. After some simplifications, the following expression for the red white-balance gain for this case is obtained:

$$K'_{R_{m,n}} = \frac{\beta_1 + T_s \cdot \left(K_R \cdot R_{m,n} - \overline{I} \right)}{\beta_1 + T_s \cdot \left(R_{m,n} - \overline{I} \right)}. \tag{3.39}$$

- Case 2: $T_{int_{m,n}} = T_{max}$ in both models

  In this case, the control node voltage of the pixel located at position (m, n) in the sensor array reaches the voltage threshold V_{mid} after the time instant T_{max} in both models. As a consequence, the integration time of the pixel in each model is not determined by the corresponding expressions for $T_{mid_{m,n}}$ (Equations (3.36) and (3.33)), but rather fixed at T_{max}. The expression for the red white-balance gain in this case is then obtained by replacing $R_{TM_{m,n}}$ and $R_{TM_{m,n}}^{ref}$ by the corresponding expressions given by Equations (3.37) and (3.35) respectively, when $T_{int_{m,n}} = T_{max}$:

$$K'_{R_{m,n}} = K_R. \tag{3.40}$$

- Case 3: $T_{int_{m,n}} = T_{max}$ in the real focal-plane model and $T_{int_{m,n}} = T_{mid_{m,n}}$ in the reference model

  In this case, the red pixel at position (m, n) has its integration time fixed at T_{max} in the real focal-plane model. In the reference model, the same red pixel has a higher value, since the red white-balance gain K_R (from the DCRaw software), that multiplies the raw pixel value, is higher than unity. It is assumed that the value of such pixel in the reference model is high enough, so that its corresponding integration time $T_{int_{m,n}}$ is lower than T_{max} (and, consequently,

(a) Reference model   (b) Real focal-plane model

Figure 3.33: Images resulting from (a) the reference model and (b) the real focal-plane model. Each model uses a different white-balance approach.

is no longer fixed at T_{max}). In Equation (3.38), $R_{TM_{m,n}}$ must be substituted by the expression in Equation (3.37) that corresponds to $T_{int_{m,n}} = T_{max}$, while $R^{ref}_{TM_{m,n}}$ must be substituted by the expression in Equation (3.35) that corresponds to $T_{int_{m,n}} = T_{mid_{m,n}}$, in order to yield the red white-balance gain expression for this case:

$$K'_{R_{m,n}} = \frac{\beta_1 + T_s \cdot \left(K_R \cdot R_{m,n} - \overline{I} \right)}{R_{m,n} \cdot T_{max}}.$$ (3.41)

The red white-balance gain expressions for each case are summarized in Equation (3.42). The blue white-balance gain expressions are the same as the ones in Equation (3.42), with $K'_{R_{m,n}}$, $R_{m,n}$ and K_R replaced by $K'_{B_{m,n}}$, $B_{m,n}$ and K_B, respectively. Figure 3.33 shows the images resulting from the reference model and the real focal-plane model, using the white-balance gains from Equation (3.42). Both images present no color distortion and are identical, as verified in Figure 3.34, which shows the difference of the corresponding pixels from the images of Figure 3.33.

$$K'_{R_{m,n}} = \begin{cases} \frac{\beta_1 + T_s \cdot \left(K_R \cdot R_{m,n} - \overline{I} \right)}{\beta_1 + T_s \cdot \left(R_{m,n} - \overline{I} \right)}, & \text{if case 1} \\ K_R, & \text{if case 2} \\ \frac{\beta_1 + T_s \cdot \left(K_R \cdot R_{m,n} - \overline{I} \right)}{R_{m,n} \cdot T_{max}}, & \text{if case 3.} \end{cases}$$ (3.42)

The white-balance gains derived previously depend on the DCRaw gains K_R and K_B, which are not available in the real focal-plane implementation and may be different for each image. The particular problem of developing methods for estimating the DCRaw gains in the focal plane is not handled by the present work and, hence, the white-balance gains of the images resulting from the digital simulations of the real focal-plane implementation were calculated using the corresponding DCRaw gains for each image. However, because it was observed that the red and blue DCRaw gains of every image tested in this work presented variations much

Figure 3.34: Image of the difference per pixel between the images shown in Figure 3.33, showing that they are identical.

below one order of magnitude (the maximum and minimum registered gains for the red channel are 1.55 and 2.26, respectively, and for the blue channel, 1.17 and 2.21), tests were made by approximating such gains by constant values within the corresponding observed ranges. The color distortion introduced by this approximation was not regarded as significant (see Appendix B). This might indicate that using the same DCRaw gains for all images in a real focal-plane implementation does not greatly deteriorate the quality of the resulting images, but further studies are required to confirm this observation. The use of the same DCRaw gains has the obvious advantage of avoiding the implementation of a possibly complex focal-plane algorithm for estimating them.

Besides the DCRaw gains, the white-balance gains depend on the raw pixel values ($R_{m,n}$ and $B_{m,n}$ for the red and blue gains, respectively). These values are not available either, because raw values undergo tone mapping immediately after image capture. Storage of such values off the focal plane requires using an ADC (with more than 8 bits) after image capture and before tone mapping. This is not desirable, because it increases the processing time. In order to overcome this limitation and calculate the white-balance gains, the raw pixel values need to be reconstructed from the tone-mapped pixel values.

Taking into account the FPTMO curves presented in Section 3.2, a perfect reconstruction may be achieved for raw values that lie outside the saturation region of the operator. Raw values that lie within this region are mapped to unity, thereby making it impossible to perfectly reconstruct the original raw value (the tone-mapped pixel value of unity always reconstructs the same raw value). The reconstruction for a red raw pixel value is considered next.

The red tone-mapped pixel expression in the real focal-plane implementation (and before tone mapping) is similar to Equation (3.34), only omitting the gain K_R, since no white balance is performed a priori in this case:

86

$$R_{TM_{m,n}} = \frac{\beta_2 \cdot R_{m,n} \cdot T_{int_{m,n}}}{V_{rst}}. \tag{3.43}$$

The red raw value associated with the red tone-mapped value can be reconstructed from Equation (3.43), by solving it for $R_{m,n}$:

$$R_{rec_{m,n}} = \frac{R_{TM_{m,n}} \cdot V_{rst}}{\beta_2 \cdot T_{int_{m,n}}}. \tag{3.44}$$

Because Equation (3.44) depends on the integration time $T_{int_{m,n}}$, which is either equal to $T_{mid_{m,n}}$ or T_{max}, two possible raw pixel value expressions exist. In the real focal-plane implementation, the integration time of each pixel is not measured and, therefore, it is unknown. In order to decide which curve to use, a condition must be stated in terms of the tone-mapped pixel value, which is available, rather than its integration time.

Using the relation defined in Equation (3.44), the first possible expression for $R_{m,n}$ is obtained by substituting $T_{int_{m,n}} = T_{max}$:

$$R_{m,n} = \frac{R_{TM_{m,n}} \cdot V_{rst}}{\beta_2 \cdot T_{max}}. \tag{3.45}$$

Similarly, the second expression is obtained by making $T_{int_{m,n}} = T_{mid_{m,n}} = \frac{1}{R_{m,n}} \cdot \left(\beta_1 - T_s \cdot \overline{I} \right) + T_s$ and regrouping the terms:

$$R_{m,n} = \frac{V_{rst} \cdot R_{TM_{m,n}} - \beta_2 \cdot \left[\beta_1 - T_s \cdot \overline{I} \right]}{\beta_2 \cdot T_s}. \tag{3.46}$$

A threshold value for the tone-mapped red pixel R_{th} that defines the reconstruction curve to be used is the point at which both curves intersect each other. Thus, this threshold value is determined by equating the expressions from (3.45) and (3.46) and solving for $R_{TM_{m,n}}$ (which is denoted as R_{th}):

$$R_{th} = \frac{T_{max} \cdot \beta_2}{V_{rst}} \cdot \left[\frac{\beta_1 - T_s \cdot \overline{I}}{T_{max} - T_s} \right]. \tag{3.47}$$

If the tone-mapped pixel value is below the threshold, then its integration time is fixed at T_{max}. On the contrary, if the tone-mapped pixel value is above the threshold, then its integration time is determined by the $T_{mid_{m,n}}$ expression from Equation (3.36). Typically, this threshold is within the [0,1] range, but it is also possible for this threshold to lie outside this range, depending on the operator parameters. If $R_{th} < 0$ or $R_{th} > 1$, then all the pixels have their integration times either equal to or below T_{max}, respectively. In either case, only one curve is needed to reconstruct the raw value and the tone-mapping operator has the same response as a linear operator. The reconstructed raw values are then defined according to Equation (3.48):

(a) Using original raw values.   (b) Using reconstructed raw values.

Figure 3.35: Images resulting from the white-balance methods using (a) the original raw values and (b) the reconstructed raw values.

Figure 3.36: Image of the difference per pixel between the images shown in Figure 3.35, showing that they are identical.

$$R_{rec_{m,n}} = \begin{cases} \frac{V_{rst}}{\beta_2 \cdot T_{max}} \cdot R_{TM_{m,n}}, & R_{TM_{m,n}} \leq R_{th} \\ \frac{V_{rst} \cdot R_{TM_{m,n}} - \beta_2 \cdot \left(\beta_1 - T_s \cdot \bar{I}\right)}{\beta_2 \cdot T_s}, & R_{TM_{m,n}} > R_{th} \end{cases} \qquad (3.48)$$

Figure 3.35 shows the images obtained after the application of the white balance gains, calculated by using the reconstructed raw values and the original raw values. Both white-balance methods produce the same result, as confirmed in Figure 3.36, which shows the differences between the corresponding pixels in the images of Figure 3.35.

3.4   FPTMO Modified Circuit

While the addition of the color filter array extends the capabilities of the original operator (by allowing the generation of color tone-mapped images), it does not bring any particular advantage to the focal-plane circuit regarding its size, complexity or processing time. The FPTMO three digital implementations discussed in Section 3.3.3 correspond to different circuit configurations in the focal plane. Concerning the

hardware and the corresponding processing time demanded by each version, neither the second nor the third implemented versions are more advantageous, regarding fabrication, than the first version. The FPTMO first version corresponds to a direct extension of the original operator to include color in the output images, without any modifications in the focal-plane circuit (except for the color filter array).

Results from the FPTMO second digital version suggest that tone-mapped images with more saturated (i.e. vivid) and natural-looking colors can be generated without requiring that blue and red pixels control their own tone mapping individually. In this version, the tone-mapping function is applied only on the green pixels, which are then used, along with the original green channel pixel values, to determine the tone-mapped values of the remaining channels. Blue or red pixel tone mapping thus depends on the green pixels. This operation principle can be further exploited, in order to reduce the focal-plane circuit size. Motivated by the performance of this version, a modification in the original focal-plane circuit is proposed in [37] and a new (fourth) digital version of the FPTMO, which is a modified version of the first digital implementation, is implemented.

In the original circuit, each pixel has its integration time and tone-mapped value determined solely based on the voltages at their own control and capture nodes, respectively, without considering the control and capture nodes from other pixels. Each pixel cell then requires two photodiodes, one for each node. The fourth FPTMO digital version maps the red and blue pixels using some information from the green pixels (which roughly represent the luminance values of the scene). The modification with respect to the FPTMO first digital version consists in using the same integration time for a red-green (or blue-green) pixel pair in the Bayer array.

With this modification, the integration time of each red and blue pixel is determined by the control node voltage of the green pixel located above or below them, instead of being determined by their own control node voltage. Since the red and blue pixels no longer need a control node, their control photodiodes can be removed, thereby leading to sensor array area reduction. For a 2×2 pixel block in the sensor array (RGGB), six photodiodes are needed in the modified version (two photodiodes for each green pixel, one for the red pixel and one for the blue pixel), while eight photodiodes are required in the original circuit (two photodiodes for each pixel). In the modified version, the integration times depend on the average green photocurrent, instead of the average photocurrent of the entire sensor array. During $t < T_s$, every control node voltage varies according to the average green pixel photocurrent, because only those pixels are connected together.

Figure 3.37 shows the schematic diagram of a red-green pixel pair. Besides the reduction in the sensing circuit area, the white-balance gain calculation is also simplified in the FPTMO fourth digital version. Previously, corresponding red and blue
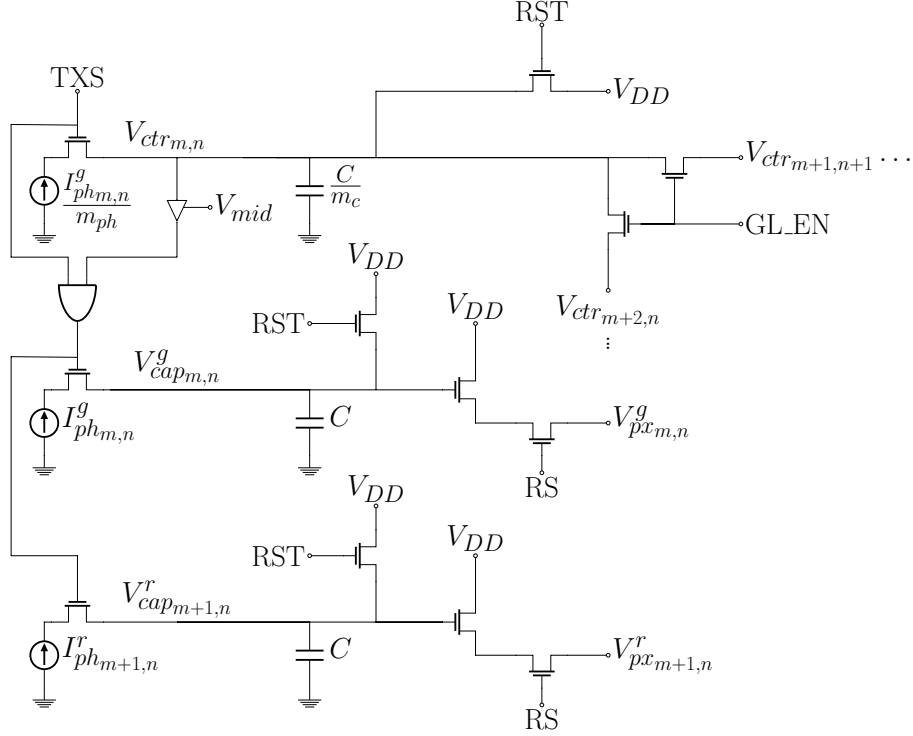
Figure 3.37: Schematic diagram for a red-green pixel pair of the modified circuit proposed in [37]. All symbols in this circuit follow the same definitions given in Section 3.1 for the corresponding symbols in the original focal-plane circuit. The superscripts "g" and "r" denote the green and red pixel, respectively.

pixels in the reference and in the real focal-plane models had different integration times, because these times were determined based on red or blue pixel values, which varied according to the model (in the reference model, the original raw values were multiplied by the white-balance gains obtained from the DCRaw software). The difference in the integration times for each model led to different combinations that satisfied Equation (3.38), each yielding a particular white-balance gain expression, as discussed in Section 3.3.4. In the fourth version, the integration time of corresponding red and blue pixels in the reference and in the real focal-plane model are exactly the same, since they are determined by the green pixels next to them, which have the same value in both models (because the DCRaw software green white-balance gains are equal to one). From Equation (3.38), considering a red pixel, $R_{TM_{m,n}}$ and $R_{TM_{m,n}}^{ref}$ have both $T_{int_{m,n}} = T_{max}$ or $T_{int_{m,n}} = T_{mid_{m,n}}$. Thus, the red white-balance gain expression in the fourth version is obtained by equating the expressions from Equations (3.35) and (3.37), associated with the same $T_{int_{m,n}}$ (the same is valid for blue pixels):

$$K'_{R_{m,n}} = K_R, \quad T_{int_{m,n}} = T_{mid_{m,n}} \text{ or } T_{int_{m,n}} = T_{max}. \tag{3.49}$$

Figure 3.38 shows the image resulting from the FPTMO fourth digital version.

| (a) Parasol. | (b) Sunset. | (c) Color Checker. |

Figure 3.38: Color images resulting from the fourth FPTMO digital implementation, in which the red and blue pixel integration times are determined by the neighboring green pixel.

In comparison with the second digital version previously implemented, the fourth version also uses less photodiodes in the sensor array, but does not require additional hardware outside the focal plane, in order to complete the tone-mapping operation. The colors of the images resulting from the fourth version look similar to those of the second version (a more detailed discussion on the color of the resulting tone-mapped images is provided in Chapter 4).

3.5   Complexity Analysis

Besides the quality of the resulting images, another important aspect to consider for each tone-mapping operator is the demanded execution time. Each operator has its own complexity, which is directly related to the computational and hardware resources required by the operator. Highly complex operators need a higher amount of resources and, thus, demand higher execution times. The evaluation of the tone-mapping operators based on their complexity is called *complexity analysis*. In this work, the complexity analysis is divided into two cases. In the first case, all tone-mapping operators are assumed to be digitally implemented and executed in the same hardware. In the second, the differences between the tone-mapping operator implemented inside the focal plane and other digital tone-mapping operators implemented outside the focal plane are considered.

3.5.1   Digital Complexity Analysis

The digital version of a tone-mapping operator consists in representing the corresponding tone-mapping operation as an algorithm, that is, as a set of computations that must be performed by some digital processing unit, in order to yield the resulting tone-mapped images. Using any programming language, the algorithm is implemented as a code (or a script), which contains the necessary computation instructions. For the digital implementations, all signal processing tasks are per-

formed by a typical computer central-processing unit (CPU). In the present work, the expression "digital processor" is henceforth used to refer to a typical CPU. The complexity of each operator is analyzed according to the following procedure.

Every instruction of each tone-mapping operator is divided into basic operations supported by the digital processor, such as summation, multiplication and comparison. Instructions that correspond to constant assignments and scalar operations are not considered in the complexity analysis, since their cost is significantly lower than the cost of operations in which the arguments are matrices representing the images. The code execution flow of some tone-mapping operators may change according to the values of some variables, which directly affects the number of operations required to finish the code execution. In order to take into account every possible execution flow, every instruction whose execution depends on a certain condition to be met is associated with a probability value representing the chance pertaining to such condition. For the digital implementations of the FPTMO, the probability values are determined based on pessimistic scenarios, in which the operator requires more clock cycles and, hence, more time than usual to finish its code execution, thereby worsening its performance. For the other digital tone-mapping operators, these values are determined based on the idea of improving their performance. This is achieved by considering optimistic scenarios, in which the operators require less clock cycles than usual to finish their code execution.

All operations are assumed to be executed sequentially (that is to say, no parallelism is considered in digital implementations) and are defined in terms of number of clock cycles required by the processor, in order to yield the result of the operation. The arithmetic operations are cumulative, which means that the result of the arithmetic operation is automatically assigned to the register that contains one of the operands. The attribution operation comprises pre-allocation of variables in the code and assignment of a value to a variable. Table 3.1 shows the basic operations considered in the analysis and the number of clock cycles associated with each operation, which are estimated according to their latency values defined in the Intel 64 and IA-32 architecture reference manual [38] (assuming a processor in the Skylake architecture). Since the power (x^K), exponential (e^x) and logarithm ($\log_K x$) operations are not defined in the manual, they are estimated as follows.

The power function is represented as a succession of multiplications. Considering that the lowest integer exponent for the power operation is K = 2, this operation requires a minimum of one multiplication and, hence, the minimum cost of this operation is the cost of a multiplication. For simplicity, the power function cost is estimated to be constant at this minimum value. The exponential operation e^x can be represented as a Taylor series of the form: $e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!}$. Assuming that the exponentiation is approximated by the first three terms of this series ($n = 2$), the

Operation	Number of Clock Cycles
Multiplication	3
Sum	1
Subtraction	1
Division	85
Attribution	1
Comparison	1
Power*	3
Exponential*	90
Logarithm*	273

Table 3.1: Basic operations considered in the complexity analysis and the number of clock cycles required per operation. The number of clock cycles are estimated according to [38], except for the operations marked with (*), which are arbitrarily estimated.

total cost of this operation consists in summing the costs of one multiplication, one division and two sums.

The natural logarithm function $\log_e x = \ln x$ also has a Taylor series representation: $\ln x = \sum_{n=1}^{\infty} \frac{(-1)^{n+1}}{n} \cdot (x-1)^n$. Approximating this function by the first three terms of the series yields the following expression (after expanding the binomials and regrouping the terms): $\ln x \approx \frac{x^3}{3} + 3x - \left(\frac{3x^2}{2} + \frac{11}{6} \right)$. The cost of calculating this expression is equal to summing the costs of five multiplications, three divisions, two sums and one subtraction. The calculation of a logarithm function in any base is then assumed to have the same cost of calculating the natural logarithm approximation.

More complex operations are defined in terms of the basic operations. This is the case, for example, for the image filtering operation, which is used by some tone-mapping operators. The operation consists of a 2-D convolution in the space domain between the filter and the input image, which is expressed in Equation (3.50) (with $x(m, n)$ denoting the pixel at location (m, n) of an image with M rows and N columns, f denoting the filter and y denoting the output image):

$$y(m, n) = \sum_{n'=0}^{N-1} \sum_{m'=0}^{M-1} x(m', n') \cdot f(m' - m, n' - n). \tag{3.50}$$

Efficient algorithms which optimize the calculation of the convolution, such as overlap-add and overlap-save methods and fast fourier transform (FFT) algorithms, are not considered in this analysis. Considering that the filter size is $M_{\text{filter}} \times N_{\text{filter}}$, each output pixel value of Equation (3.50) requires $M_{\text{filter}} N_{\text{filter}}$ multiplications and $M_{\text{filter}} N_{\text{filter}} - 1$ additions. The full 2-D convolution yields an image of size $(M + M_{\text{filter}} - 1) \times (N + N_{\text{filter}} - 1)$. In order to keep the size of the output image the same as the original image, the full 2-D convolution is not considered, but rather a reduced

version of it. This version assumes that the center of the filter kernel is always within the original image boundaries. The output pixel values that are calculated are only those that satisfy this condition, thus yielding an output image of size M × N. In this case, the total cost of the reduced 2-D convolution operation is $(M_{filter} \cdot N_{filter}) \cdot MN$ multiplications and $(M_{filter} \cdot N_{filter} - 1) \cdot MN$ additions.

The demosaicing operation can also be represented by the basic operations. However, since the same demosaicing algorithm is used in every implemented tone-mapping operator, the total cost of each tone-mapping operator is offset by the same constant value, which represents the demosaicing operation cost. Because in this analysis the tone-mapping operators are ranked according to the number of required clock cycles and this operation does not change such ranking, this operation cost is arbitrarily defined to be unitary.

For tone-mapping operators whose input is a color (RGB) raw image, rather than a raw image in the Bayer pattern, some pre-processing operations that transform the raw image in the Bayer pattern into a color raw image are performed before the tone mapping takes place (see Appendix A for more details on these operations). For such operators, the cost of the tone mapping performed by them also includes the total cost of this pre-processing stage.

3.5.2 Focal-Plane Complexity Analysis

In the previous analysis, it is considered that the scene is captured and digitized by the camera hardware and then made available to the digitally implemented tone-mapping operators, which are all executed on the same hardware (a digital processor outside the focal plane). In this case, every tone-mapping operator uses the same hardware configuration and the respective execution times depend only on the number of instructions (and their associated cost expressed as a number of digital processor clock cycles) required by each operator to complete the tone mapping. The real FPTMO implementation (that is, the hardware implementation of the tone-mapping operator inside the focal plane) uses a hardware configuration different from that used by the other digital operators which perform tone mapping outside the focal plane. The hardware configuration used for each operator has a direct influence on the overall processing time, since in each configuration the raw image processing is performed differently. The complexity associated with each configuration is analyzed in the present section.

Figure 3.39 shows a cascade of image processing stages that support the FPTMO implementation. The tone mapping is performed in the focal plane as the image is acquired, and the same signal processing task within each pixel is executed in parallel, that is to say, every pixel is simultaneously tone mapped. The maximum

Figure 3.39: Cascade of image processing stages that support focal-plane tone mapping. The FPTMO itself is denoted by the "Tone Mapping" block.

execution time necessary to complete the tone-mapping operation on the entire image is T_{max}, because this is the maximum time required for tone mapping a single pixel in the sensor array. After the tone mapping, the resulting analog pixel values must then be converted to digital values, in order to display the tone-mapped image on a conventional display medium. This task is performed by an ADC. There exists a wide variety of ADC types for image sensors, each one with a different architecture. Depending on the application, the ADCs are designed for different bit resolutions and operate with different clock frequencies, which are not necessarily equal to the clock used in other circuit parts, such as the sensing array. The ADC choice influences overall image processing time. Two different ADC types are considered in the present work: the ramp ADC and the successive approximation register (SAR) ADC. The ADC timing analysis presented in this work is based on [39]. In that reference, the authors presented a detailed comparative time and energy analysis involving different ADC architectures for image sensors.

The last image processing stage is the color treatment. It consists of the white-balance and demosaicing operations. Since these operations are performed on the digitized images outside the focal plane, they are implemented as software in a digital computer. Their complexity is analyzed using the same procedure for the digital tone-mapping operators in Section 3.5.1. Depending on the implemented version of the FPTMO, the white balance involves different calculations. The two versions considered for implementation in the focal plane are: i) the direct extension of the original FPTMO to include color information (denoted as first version in Section 3.3.3), and ii) the modified circuit version, in which the tone mapping of red and blue pixels is controlled by neighboring green pixels (denoted as fourth version and discussed in Section 3.4). In the first version, the white-balance operation comprises raw values reconstruction, gains calculation, and multiplication of the tone-mapped image by the calculated gains. In the fourth version, this operation consists only in the multiplication of the tone-mapped image by previously calibrated constant white-balance gains. The demosaicing operation is the same for both versions. Assuming that the sensor array has size M × N, the FPTMO overall processing time, taking into account the signal processing stages shown in Figure 3.39, can be expressed as follows:

$$T_{FPTMO} = T_{TM} + T_{ADC} + T_{CT}$$
$$= T_{max} + \frac{\text{MN} \cdot T_{conv\_ADC\_type}}{N_{ADC}} + N_{clk\_cycles\_CT} \cdot \tau_{clk\_digital}. \qquad (3.51)$$

In Equation (3.51), T_{TM} is the processing time of the tone-mapping operation (which is assumed to be equal to the maximum tone-mapping processing time T_{max}), T_{ADC} is the time required for converting all pixels of the array to digital values (which depends on the number of ADCs N_{ADC} used in the circuit and the conversion time of one pixel $T_{conv\_ADC\_type}$) and T_{CT} is the processing time of the color treatment operations (which is determined based on the number of clock cycles required for the color treatment operations $N_{clk\_cycles\_CT}$ and on the operating clock period of the digital processor $\tau_{clk\_digital}$). The conversion time for one pixel depends on the ADC type considered. For the ramp ADC, this time is given by:

$$T_{conv\_Ramp\_ADC} = 2^{N_{bits}} \cdot \tau_{clk\_Ramp\_ADC}, \qquad (3.52)$$

where N_{bits} is the bit resolution of the ADC and $\tau_{clk\_Ramp\_ADC}$ denotes its operating clock period. For this hardware configuration, the ADC must have 8-bit resolution, because the tone-mapped pixels are in the 8-bit dynamic range. In [39], the authors establish a relation, which varies according to the ADC type, between the ADC clock period $\tau_{clk\_ADC\_type}$ and the global clock period of the circuit τ_{clk}:

$$\tau_{clk\_ADC\_type} = K_{ADC\_type} \cdot \tau_{clk}. \qquad (3.53)$$

Defining the global clock period as $\tau_{clk} = T_{max}$ (the time required to perform the tone mapping in the focal plane), substituting Equation (3.52) in Equation (3.51), and considering the ramp ADC with $N_{bits} = 8$ leads to the following expression:

$$T_{FPTMO} = T_{max} + \frac{\text{MN}}{N_{ADC}} \cdot 2^8 \cdot K_{Ramp\_8bits} \cdot T_{max} + N_{clk\_cycles\_CT} \cdot \tau_{clk\_digital}. \qquad (3.54)$$

The parameter T_{max} is defined by the user. The constant K_{Ramp_8bits} is arbitrarily set to the median operating clock of the set of ramp ADCs, which is obtained by substituting the median conversion time value reported in [39] (which is normalized to eight bits) for this type of ADC in Equation (3.52). The clock period of the digital processor $\tau_{clk\_digital}$ can also be expressed in terms of the focal-plane operating clock period T_{max}:

$$\tau_{clk\_digital} = K_{DP} \cdot T_{max}. \qquad (3.55)$$

The digital processor clock period can be estimated based on the clock periods of recent commonly used CPUs. After the estimation of the digital processor clock period, the constant K_{DP} can be obtained. This leads to the final expression for the FPTMO processing time using a ramp ADC given in terms of the focal-plane circuit clock period:

$$
\begin{aligned}
T_{FPTMO} &= T_{max} + \frac{MN}{N_{ADC}} \cdot 2^8 \cdot K_{Ramp\_8bits} \cdot T_{max} + N_{clk\_cycles\_CT} \cdot K_{DP} \cdot T_{max} \\
&= T_{max} \cdot \left(1 + \frac{MN}{N_{ADC}} \cdot 256 \cdot K_{Ramp\_8bits} + N_{clk\_cycles\_CT} \cdot K_{DP} \right).
\end{aligned}
\tag{3.56}
$$

For the SAR ADC, its conversion time for one pixel is given by:

$$
T_{conv\_SAR\_ADC} = N_{bits} \cdot \tau_{clk\_SAR\_ADC},
\tag{3.57}
$$

where $\tau_{clk\_SAR\_ADC}$ is the operating clock period of the SAR ADC, which can be related to the clock period of the ramp ADC as follows: $\tau_{clk\_SAR\_ADC} = K_{SAR} \cdot \tau_{clk\_Ramp\_ADC} = K_{SAR\_8bits} \cdot K_{Ramp\_8bits} \cdot T_{max}$. As stated in [39], the SAR ADC runs at a clock speed 16 times slower than the ramp ADC, thus yielding $K_{SAR\_8bits} = 16$. Substituting Equation (3.57) in (3.51) and using $N_{bits} = 8$ leads to the final expression for the FPTMO processing time using the SAR ADC:

$$
\begin{aligned}
T_{FPTMO} &= T_{max} + \frac{MN}{N_{ADC}} \cdot 8 \cdot K_{SAR} \cdot K_{Ramp\_8bits} \cdot T_{max} + N_{clk\_cycles\_CT} \cdot K_{DP} \cdot T_{max} \\
&= T_{max} \cdot \left(1 + \frac{MN}{N_{ADC}} \cdot 8 \cdot 16 \cdot K_{Ramp\_8bits} + N_{clk\_cycles\_CT} \cdot K_{DP} \right) \\
&= T_{max} \cdot \left(1 + \frac{MN}{N_{ADC}} \cdot 128 \cdot K_{Ramp\_8bits} + N_{clk\_cycles\_CT} \cdot K_{DP} \right).
\end{aligned}
\tag{3.58}
$$

Figure 3.40 shows a cascade of image processing stages that support digital tone mapping implementations. All image processing tasks run on the digitized raw image outside the focal plane. Because no tone mapping is performed inside the focal plane, the analog raw pixel values may be in a dynamic range higher than the 8-bit dynamic range (which is the case for HDR scenes). To preserve the original values registered in the camera sensor array, the analog raw pixel values must be converted to digital values that use more than eight bits. In this work, the scene dynamic range is assumed to be entirely contained within the 12-bit dynamic range and, as such, a 12-bit ADC is used in this hardware configuration. Because of the higher bit resolution, such ADC requires longer conversion times than the 8-bit resolution counterpart.

The white-balance operation for the digital tone-mapping operators consists in
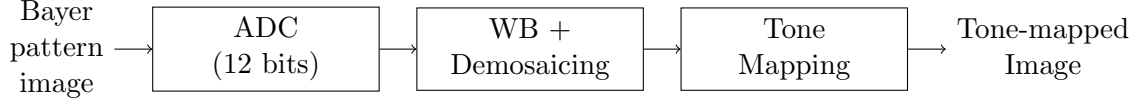
Figure 3.40: Cascade of image processing stages that support digital tone mapping implementations. The digital tone-mapping operator itself is denoted by the "Tone Mapping" block.

multiplying each pixel of the 12-bit digitized Bayer pattern image by the corresponding gain obtained from the DCRaw software. The demosaicing operation is the same one used for the FPTMO. After these operations, the tone mapping function of the corresponding digital operator, implemented in software, is applied to the 12-bit digitized color raw image. The tone-mapping and the color treatment operations are executed on the same hardware, using the same digital processor. The total processing time for a digital tone-mapping operator (DTMO), taking into account the signal processing stages shown in Figure 3.40, is expressed as follows:

$$
\begin{aligned}
T_{DTMO} &= T_{ADC} + T_{TM} + T_{CT} \\
&= \frac{\text{MN} \cdot T_{conv\_ADC\_type}}{N_{ADC}} + (N_{clk\_cycles\_TM} + N_{clk\_cycles\_CT}) \cdot \tau_{clk\_digital}.
\end{aligned}
\tag{3.59}
$$

In Equation (3.59), T_{ADC} is the time required for analog-to-digital conversion of all pixels of the array (which depends on the 12-bit ADC operating clock period $\tau_{clk\_ADC\_type}$ and on the number of ADCs N_{ADC} used in the camera internal circuit), T_{CT} is the required time to fully execute the color treatment operations (which is determined as a function of the number of clock cycles $N_{clk\_cycles\_CT}$ required by both the white balance and demosaicing and the digital processor clock period $\tau_{clk\_digital}$), and T_{TM} is the tone-mapping operator algorithm execution time (which is based on the number of clock cycles demanded by the corresponding tone-mapping operator, calculated in the previous analysis, and on the digital processor clock period $\tau_{clk\_digital}$). As in the FPTMO processing time expression, both clock periods $\tau_{clk\_digital}$ and $\tau_{clk\_ADC\_type}$ can be expressed in terms of one global clock period, which is defined as T_{max}. Considering a 12-bit ramp ADC, Equation (3.59) is rewritten as follows:

$$
\begin{aligned}
T_{DTMO} &= \frac{\text{MN} \cdot 2^{12} \cdot \tau_{clk\_ADC\_ramp}}{N_{ADC}} + (N_{clk\_cycles\_TM} + N_{clk\_cycles\_CT}) \cdot \tau_{clk\_digital} \\
&= \frac{\text{MN}}{N_{ADC}} \cdot 4096 \cdot K_{Ramp\_12bits} \cdot T_{max} + (N_{clk\_cycles\_TM} + N_{clk\_cycles\_CT}) \cdot K_{DP} \cdot T_{max} \\
&= T_{max} \cdot \left[\frac{\text{MN}}{N_{ADC}} \cdot 4096 \cdot K_{Ramp\_12bits} + (N_{clk\_cycles\_TM} + N_{clk\_cycles\_CT}) \cdot K_{DP} \right]
\end{aligned}
\tag{3.60}
$$
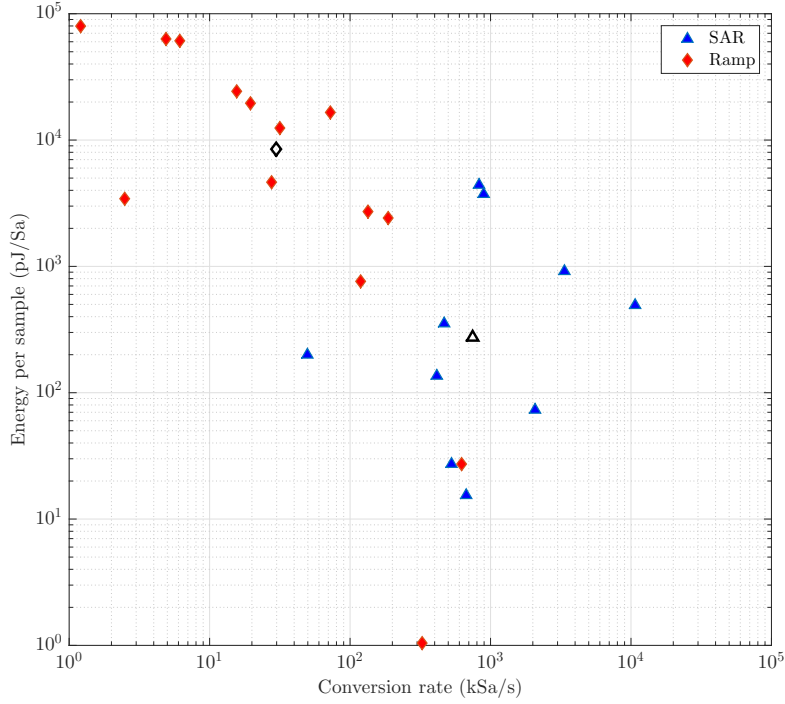
Figure 3.41: Energy per sample versus data conversion rate for the considered ramp and SAR ADCs, normalized to twelve bits. The black unfilled markers represent the median values of the corresponding ADC types. The data are a courtesy from the authors of [38].

The constant K_{DP} is the same as the one calculated in the previous derivation of the FPTMO processing time expression. In order to obtain the constant value K_{Ramp_12bits}, it is necessary to calculate the median conversion rate normalized to twelve bits of the set of ramp ADCs analyzed in [39]. The authors gently shared the data they gathered for the different ADCs reported in the literature for image sensor applications. Using this data, each ADC conversion rate is normalized according to the same procedure used by the authors, except that the number of bits considered is twelve, instead of eight. After the normalization, the median conversion rate of each ADC type is calculated and the constant K_{Ramp_12bits} can then be obtained. Figure 3.41 shows the conversion rates of the considered ramp and SAR ADCs.

The SAR ADC operating clock period can also be expressed in terms of the ramp ADC operating clock period: $\tau_{clk\_ADC\_SAR} = K_{SAR} \cdot \tau_{clk\_ADC\_ramp} = K_{SAR\_12bits} \cdot K_{ramp\_12bits} \cdot T_{max}$. The results from Figure 3.41 indicate that the SAR normalized median conversion rate is approximately 25 times higher than the one of the ramp ADC. Based on this observation and following the same procedure in [39], it is concluded that the constant K_{SAR_12bits} is approximately equal to 14. This leads to the final expression for the DTMO overall processing time, using the SAR ADC:

$$
\begin{aligned}
T_{DTMO} &= T_{max} \cdot \left[\frac{MN}{N_{ADC}} \cdot N_{bits} \cdot K_{SAR\_12bits} \cdot K_{Ramp\_12bits} + (N_{clk\_cycles\_TM} + N_{clk\_cycles\_CT}) \cdot K_{DP} \right] \\
&= T_{max} \cdot \left[\frac{MN}{N_{ADC}} \cdot 12 \cdot 14 \cdot K_{Ramp\_12bits} + (N_{clk\_cycles\_TM} + N_{clk\_cycles\_CT}) \cdot K_{DP} \right] \\
&= T_{max} \cdot \left[\frac{MN}{N_{ADC}} \cdot 168 \cdot K_{Ramp\_12bits} + (N_{clk\_cycles\_TM} + N_{clk\_cycles\_CT}) \cdot K_{DP} \right].
\end{aligned}
\tag{3.61}
$$

Chapter 4

Results and Discussions

In this chapter, a comparison between the images resulting from all operators implemented in this work is presented. Furthermore, the execution times of each operator are shown. Since each operator has its own set of parameters, which generates a different image, some conditions must be stated, in order to rank the output images. The parameters can then be tuned accordingly to yield the best ranked images in each case. The criteria established for the image selection of every operator were the following ones:

1) Details in darker and brighter regions of the image should be visible (i.e. regions are not over- or underexposed);

2) The colors of the scene look natural.

The implemented versions of every DTMO are the basic ones, which do not attempt to simulate several more complex effects of human vision, such as glare and visual acuity. This decision is based in two reasons: first, the main purpose of the tone-mapping task defined in this work, which is to display details in bright and dark regions, is achieved without requiring these steps; second, only some operators include such features. The additional steps required to include these features would increase the complexity and execution times of only some operators. In order to make the comparison between the execution times fair, the implementation of every operator is kept as simple as possible.

Three digital versions of the FPTMO are considered. The first version uses the same framework of other DTMOs, which is defined in Figure 3.40. The second and third FPTMO versions use the real focal-plane framework, which is defined in Figure 3.39. In this chapter, the second FPTMO version corresponds to the first FPTMO digital implementation discussed in Chapter 3. In this version, each pixel integration time is determined by the own raw pixel value and by the average raw pixel value. The third version corresponds to the fourth FPTMO digital implementation presented in Chapter 3, in which the integration time of each red and blue

Image	Dynamic Range $\left(\log_{10}\left(\frac{L_{max}}{L_{min}}\right)\right)$
Parasol	3.3269
Fire Extinguisher	4.0224
Flowers	4.3235
Sunset	5.4438
Courtyard	5.4438
Color Chart 1	5.4438
Beach	4.4705
Palace	5.6678
Stone Tower	4.4705
Color Chart 2	5.6204

Table 4.1: HDR images that are used in the present work, and an order-of-magnitude indication of their dynamic ranges. LDR images have $\log_{10} L_{max}/L_{min}$ below 2.41.

pixel is determined by the raw green pixel located above (or below) them and by the average green raw pixel value. The demosaicing operation used for all operators is based on [33].

4.1 Overall DTMO Image Quality

The results of the application of every tone-mapping operator implemented in this work applied in ten different HDR images are presented in Appendix C. Besides the tone-mapped images, the original raw images without any tone mapping applied are also shown. The HDR images were obtained from the Empa HDR database [40]. In this chapter, only results that are relevant to the following discussion topics are shown. The luminance dynamic ranges of each image are presented in Table 4.1.

In most images, Ward Contrast-Based operator and Ferwerda operator do not exhibit details simultaneously in bright and dark regions; only one region is made visible at the cost of losing the other. This happens because such operators are linear and, as indicated in Chapter 2, linear operators are generally not suited for the tone mapping of HDR images. These operators do not change the original dynamic range and, thus, the display dynamic range can only accommodate part of the original range. If dark (or bright) areas of the image are chosen to be correctly displayed, then pixels from bright (or dark) areas have their values clamped. This limitation of linear operators is illustrated in Figure 4.1, in which the results of the application of the Ward and Ferwerda operators on the Courtyard image are presented, along with the results obtained after applying other non-linear DTMOs on the same image.

Exceptions may occur if the image contains only a few pixels, compared to the total number of pixels, that have values outside the displayable range. This is the

(a) Ward Contrast-Based.

(b) Ferwerda.
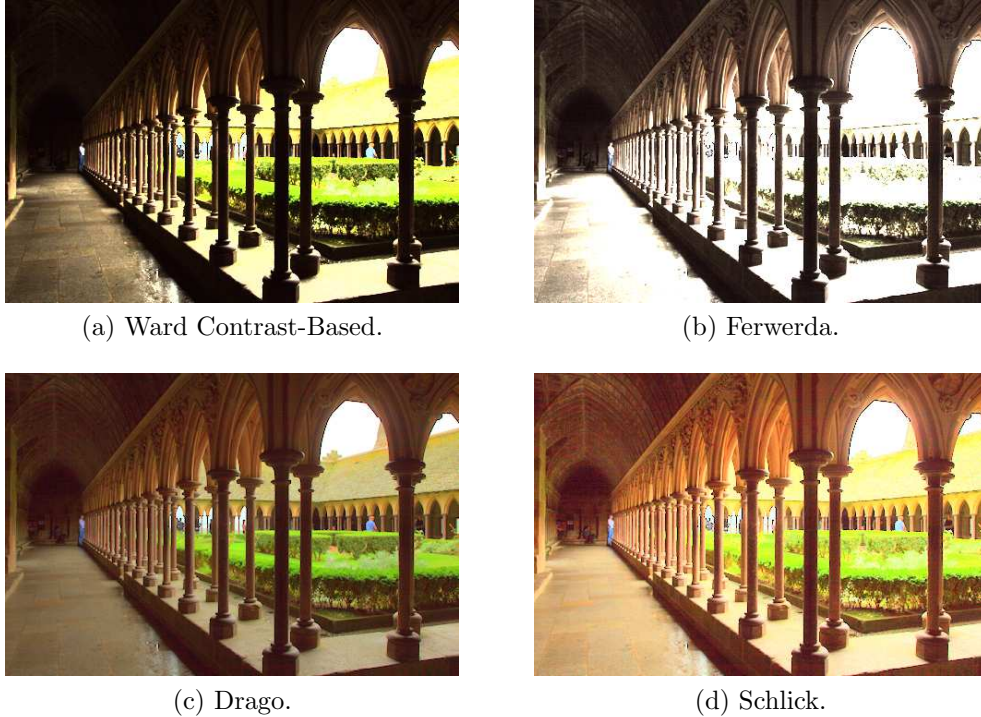
(c) Drago.

(d) Schlick.

Figure 4.1: Linear (Ward Contrast-Based and Ferwerda) and non-linear (Drago and Schlick) DTMOs applied to the Courtyard image.

case, for example, for the Flowers image, which has very few areas that are very dark or very bright, such as the plant in the upper right corner and the daisies petals. Details that are lost in these regions are barely noticeable, and thus do not affect the overall image quality impression. In such cases, linear operators could yield results comparable to non-linear operators. Figure 4.2 shows the resulting tone-mapped Flowers image, obtained after applying the same operators used for the Courtyard image in Figure 4.1.

4.1.1 Color Differences in the Tone-Mapped Images

Some differences can be noted in the colors of the images resulting from different operators. Since most DTMOs implemented in this work are applied only on the luminance channel of the color raw images, the colors of such tone-mapped images are obtained through a linear scaling of the original color channels. This procedure aims at minimizing color shifts in the resulting images by preserving the ratios between the original color channels. Non-linear DTMOs that are applied directly on the original color channels do not preserve these ratios. This is the case, for example, for the Reinhard Global operator and the Rahman operator, which generate less saturated (and, therefore, slightly different) colors in some images when compared to the colors in the same images generated by other DTMOs. That efffect can be

(a) Ward Contrast-Based.

(b) Ferwerda.

(c) Drago.

(d) Schlick.

Figure 4.2: Linear (Ward Contrast-Based and Ferwerda) and non-linear (Drago and Schlick) DTMOs applied to the Flowers image.

seen, for example, in the Parasol image, as shown in Figure 4.3.

Images obtained from the Ferwerda operator also present desaturated colors. The desaturation effect is illustrated, for example, by the Flowers image in Figure 4.2. Such results may not be expected at first, because of the linear nature of the operator, which should preserve the original color ratios. These ratios are preserved, as long as the tone-mapped values of each color channel are within the displayable range. If the tone-mapped value lies outside the displayable range, it is clamped. Thus, since this operator is applied directly on the color channels, the color ratios at a particular pixel may not be preserved, if at least one color channel has its tone-mapped value outside the displayable range. In the Flowers image generated by the Ferwerda operator, many tone-mapped pixels have two (or even all three) of their corresponding color channel values clamped, yielding color ratios equal to or close to unity. Hence, colors are less saturated. The same analysis holds for other images generated by this operator.

To further illustrate the color differences caused by not preserving the original color channel ratios, Figure 4.4 shows the Fire Extinguisher images generated from two versions of the Logarithm operator. The first version applies the logarithmic curve to the luminance channel of the color raw image and then scales the original color channels by the ratio between the tone-mapped and the original luminance values at each pixel. The second version applies the logarithmic curve to the raw

(a) Reinhard Global.

(b) Rahman.

(c) Exponential.

(d) Ward Histogram Adjustment.

Figure 4.3: DTMOs applied to the Parasol image. The Reinhard Global and Rahman operators are applied directly on the original color channels. The Ward Histogram Adjustment and Exponential operators are applied on the original luminance channel.

(a) First version.

(b) Second version.

Figure 4.4: Different implementations of the Logarithm operator applied to the Fire Extinguisher image.

sensor data, followed by white-balance and demosaicing operations, in order to yield a color tone-mapped image. Each pixel in the raw image represents the value of a specific color channel, defined by a Bayer pattern. Figure 4.5 shows the scatter plot of the RGB values of every pixel in the corresponding images, along with the RGB values of the original color raw pixels. The average color saturation value $\overline{S}$ in each case, obtained by transforming the images from the RGB to the HSV space, is also shown. The $\bar{S}$ values are normalized to the [0,1] range.

The colors are less saturated in the second version of the Logarithm operator

(a) Original $\left(\overline{S} = 0.4928\right)$. (b) First version $\left(\overline{S} = 0.4961\right)$. (c) Second version $\left(\overline{S} = 0.2538\right)$.
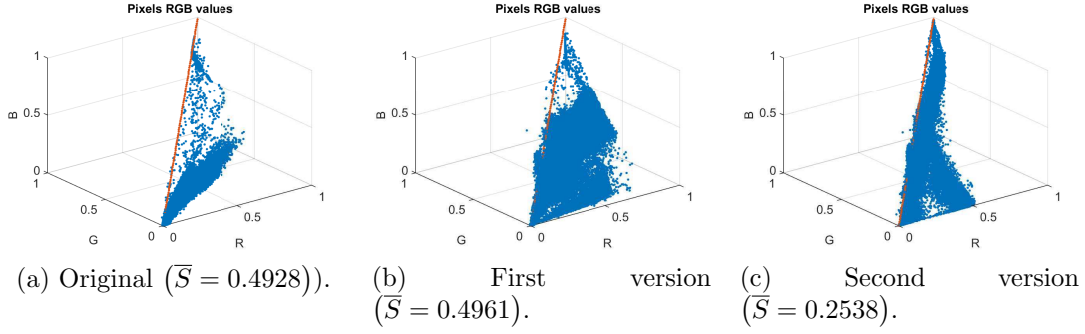
Figure 4.5: Scatter plot of RGB values of (a) raw pixels, (b) tone-mapped pixels obtained from the Logarithm operator first version, and (c) tone-mapped pixels obtained from the Logarithm operator second version. The red dots correspond to achromatic pixels (R = G = B).

than in the first version. The logarithmic curve tends to reduce the discrepancy between the raw pixel values, since the gain is higher for pixels with lower values. Consequently, the ratios between the color channels are also reduced and RGB values of the pixels, after demosaicing, are more concentrated around the achromatic line, as can be seen in Figure 4.5c. Because the first version attempts to preserve the original color channel ratios by multiplying the original color channel values of each pixel by the same gain, the resulting distribution of the pixels RGB values is more similar to the original distribution. Since the RGB values of the tone-mapped pixels obtained from the first version are more scattered from the achromatic line than the RGB values of the tone-mapped pixels obtained from the second version, the first version yields more saturated colors.

Color saturation can be increased through the application of a color saturation matrix to each color channel, defined in [41] as:

$$
\begin{bmatrix} R' \\ G' \\ B' \end{bmatrix} = \begin{bmatrix} (1-s) \cdot r_w + s & (1-s) \cdot r_w & (1-s) \cdot r_w \\ (1-s) \cdot g_w & (1-s) \cdot g_w + s & (1-s) \cdot g_w \\ (1-s) \cdot b_w & (1-s) \cdot b_w & (1-s) \cdot b_w + s \end{bmatrix} \cdot \begin{bmatrix} R \\ G \\ B \end{bmatrix}. \quad (4.1)
$$

where r_w, g_w and b_w are the weights associated with each color channel in Equation (2.5) and s is a parameter that controls the saturation of the color channels. To better understand the effect of this transformation on the colors, the resulting RGB image is converted to the YIQ space using the following relation:

$$
\begin{bmatrix} Y' \\ I' \\ Q' \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.274 & -0.322 \\ 0.211 & -0.523 & 0.312 \end{bmatrix} \cdot \begin{bmatrix} R' \\ G' \\ B' \end{bmatrix}. \quad (4.2)
$$

By solving Equation (4.1) and making $Y = r_w \cdot R + g_w \cdot G + b_w \cdot B$, three relations are obtained:

$$\begin{cases} R' = Y + s \cdot (R - Y) \\ G' = Y + s \cdot (G - Y) \\ B' = Y + s \cdot (B - Y). \end{cases} \qquad (4.3)$$

Substituting Equation (4.3) in Equation (4.2) and solving the system for Y', I', and Q' yields the final relations between the new luminance Y' and chrominances I' and Q' in terms of the original luminance Y and chrominances I and Q:

$$\begin{cases} Y' = Y \cdot (1 - s) + s \cdot Y = Y \\ I' = s \cdot (0.596 \cdot R - 0.274 \cdot G - 0.322 \cdot B) = s \cdot I \\ Q' = s \cdot (0.211 \cdot R - 0.523 \cdot G + 0.312 \cdot B) = s \cdot Q. \end{cases} \qquad (4.4)$$

From Equation (4.4), only the original chrominances are affected, while the original luminance is preserved. For $s < 0$, the new chrominances are mirrored versions of the original chrominances and, because of that, the colors of the new image are reversed. For $0 < s < 1$, the new chrominances are more concentrated around the origin of the chrominance space (which corresponds to a grayscale image) than the original chrominances, thus yielding less saturated colors. For $s > 1$, the new chrominances are more dispersed from the origin than the original chrominances and, hence, the colors of the new images are more saturated. Figure 4.6 shows the results of the application of the color saturation matrix to the Fire Extinguisher image generated by the Logarithm operator second version. Figure 4.7 shows the corresponding chrominances of each resulting image.

4.2 Overall FPTMO Image Quality

The FPTMO first version is a theoretical proof of concept. It represents a "fully digital tone-mapping" version of the FPTMO from [11], that is, it is designed without the restrictions imposed by the focal-plane implementation. By using the same framework of the DTMOs, this version offers a more direct way of comparing the FPTMO performance, regarding the resulting image quality, to its digital competitors. Images resulting from this version present a detail level in dark and bright regions similar to the detail level observed in images produced by most DTMOs implemented in this work. For example, Figure 4.8 shows a comparison between the resulting Sunset image, generated by the FPTMO first version and by three other DTMOs. Details in the bottom part of the image, such as the village buildings, are

(a) $s = -1$. (b) $s = 0$. (c) $s = 1$.

(d) $s = 2$. (e) $s = 3$. (f) $s = 4$.
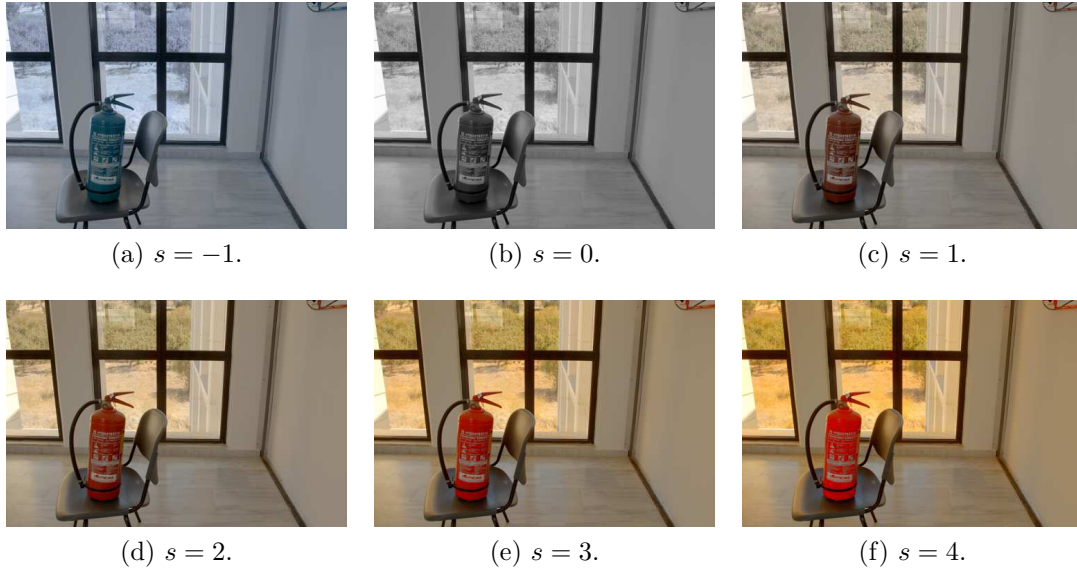
Figure 4.6: Color saturation matrix applied to the image generated from the Logarithm operator second version.

(a) $s = -1$. (b) $s = 0$. (c) $s = 1$.

(d) $s = 2$. (e) $s = 3$. (f) $s = 4$.
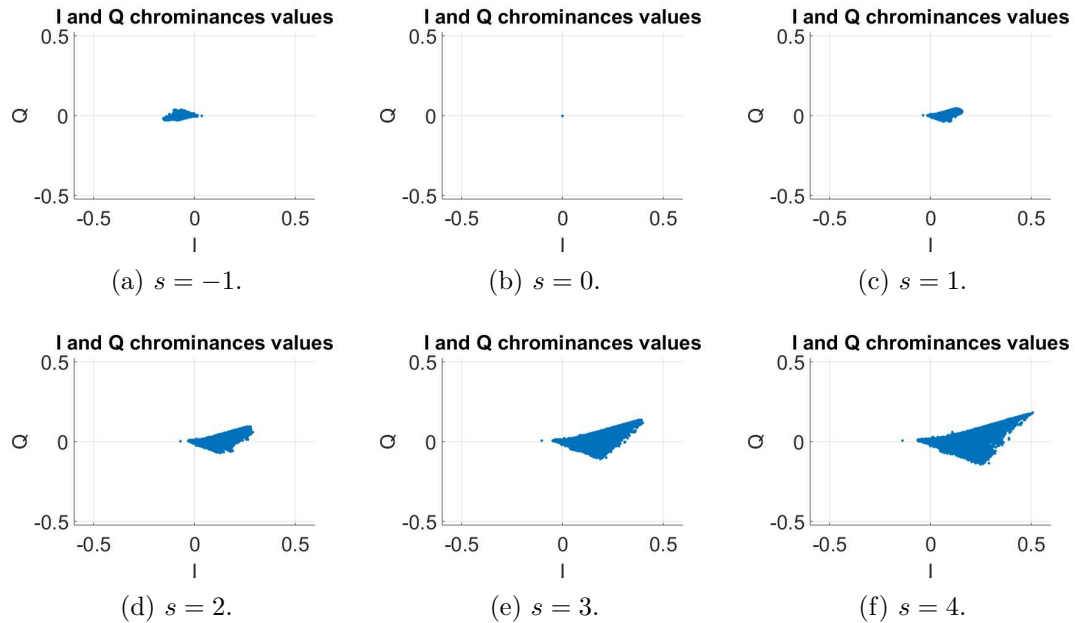
Figure 4.7: Chrominance scatter plots for the respective images in Figure 4.6.

visible, and so are the clouds in the sky in the brightest part of the image. This version also produce colors that resemble the colors of images resulting from other DTMOs.

The FPTMO second version is a direct extension of the focal-plane operator proposed in [11]. The original operating principle is maintained and only color processing is incorporated (white balance and demosaicing). No modifications to the original circuit are made, except for the addition of the Bayer CFA. This version yields images whose quality is also comparable to the quality of other DTMOs,

(a) FPTMO first version.

(b) Drago.

(c) Rahman.

(d) Ashikhmin.

Figure 4.8: FPTMO first version and different DTMOs (Drago, Rahman and Ashikhmin) applied to the Sunset image.

although the colors look different. In this version, each pixel integration time is controlled by the pixel value itself and by the average pixel value of the camera pixel array. The integration time increases as the pixel value decreases (according to Equation (3.20)). The relation between the tone-mapped pixel p_{TM} and the raw pixel p is given by:

$$p_{TM} = \frac{T_{int_{m,n}} \cdot \beta_2}{V_{rst}} \cdot p. \tag{4.5}$$

As verified empirically, pixels representing the same color channel in the camera pixel array tend to vary slowly in the space domain. Under this assumption, the original ratio between the red and green channels at a position (m,n) in the camera pixel array is approximated by the ratio between the red (raw) pixel value $R_{m,n}$ and the green (raw) pixel value $G_{m+1,n}$ located next to the red pixel, defined as $r = R_{m,n}/G_{m+1,n}$. Then, the ratio between the tone-mapped pixels is given by:

$$\frac{R_{TM_{m,n}}}{G_{TM_{m+1,n}}} = \frac{T^r_{int_{m,n}}}{T^g_{int_{m+1,n}}} \cdot \frac{R_{m,n}}{G_{m+1,n}} = \frac{T^r_{int_{m,n}}}{T^g_{int_{m+1,n}}} \cdot r, \tag{4.6}$$

where $T^r_{int_{m,n}}$ and $T^g_{int_{m+1,n}}$ are the integration times for the red and green pixels, respectively. A similar relation can be derived for the red-blue and blue-green ratios. From Equation (4.6), the original ratio r is preserved only if $T^r_{int_{m,n}}/T^g_{int_{m+1,n}} = 1$. In this version, it only happens if the $R_{m,n} = G_{m+1,n}$. Since this does not happen

(a) FPTMO second version.
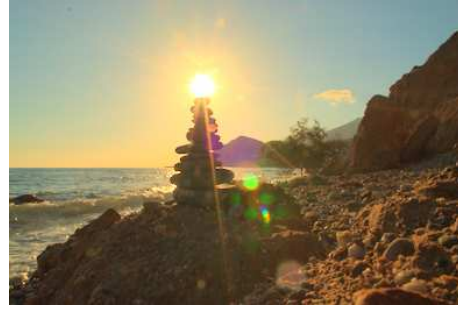
(b) Drago.

(c) Rahman.

(d) Ashikhmin.

Figure 4.9: FPTMO second version and different DTMOs (Drago, Rahman and Ashikhmin) applied to the Stone Tower image.

often, the original ratios between the color channels are usually not preserved in this version and, hence, different colors are generated.

The low color saturation can also be explained by the circuit operating principle. Since each pixel integration time is determined by the very pixel value, the duration of the photocurrent integration varies for each pixel, lasting longer for darker pixels and shorter for brighter pixels. This not only prevents pixel values next to both ends of the dynamic range from being clamped, but also tends to concentrate the pixel values around the middle of the dynamic range. Because each pixel value of the camera pixel array represents a color channel, the discrepancy between color channel values is reduced and, thus, after demosaicing, less saturated colors are generated. Figure 4.9 shows the Stone Tower image, generated by this FPTMO version and other DTMOs. Aside from the color differences, details in bright and dark regions, like the clouds in the sky, the waves in the sea and the rocks on the ground in the lower part of the image, are observable.

The FPTMO third version corresponds to the focal-plane operator circuit modified version proposed in [37]. In this version, the red and blue pixel integration time is determined by the neighboring green pixel located above or below them. Regarding colors, images resulting from this version look more similar to the FPTMO first version and other DTMOs than the FPTMO second version. Since in the third version the integration time of red and blue pixels is the same as the neighboring

(a) FPTMO third version.

(b) Drago.

(c) Rahman.

(d) Ashikhmin.

Figure 4.10: FPTMO third version and different DTMOs (Drago, Rahman and Ashikhmin) applied to the Beach image.

green pixel, the ratio $T^r_{int_{m,n}}/T^g_{int_{m,n}}$ and $T^b_{int_{m,n}}/T^g_{int_{m,n}}$ equals unity (which also leads to $T^r_{int_{m,n}}/T^b_{int_{m,n}} = 1$). Hence, from Equation (4.6), the ratio between the color channels is preserved. The detail level of the generated images is also comparable to the detail level of the images obtained from other DTMOs. As shown in Figure 4.10, the FPTMO third version exhibits details of the rocks near the sea and the tree branches in the upper right corner of the image, while also showing the cloud in the sky and, partially, the waves in the sea.

Figure 4.11 shows a comparison between the three FPTMO versions and other DTMOs applied to the Courtyard image, which contains many details in dark regions and in bright regions. The three FPTMO versions yield images comparable to the images yielded by other DTMOs. Details can be seen in the bright regions and in the dark regions: see, for example, the inner hallway, the leaves of the bushes in the garden and the roof tiles in the outside. The colors of the FPTMO first and third versions are more similar to the colors of the other DTMOs, because these FPTMO versions preserve the original color ratios, while the FPTMO second version generates different colors. The FPTMO second version does not aim at preserving original color ratios.

(a) FPTMO first version. (b) FPTMO second version. (c) FPTMO third version.

(d) Drago. (e) Rahman. (f) Ashikhmin.
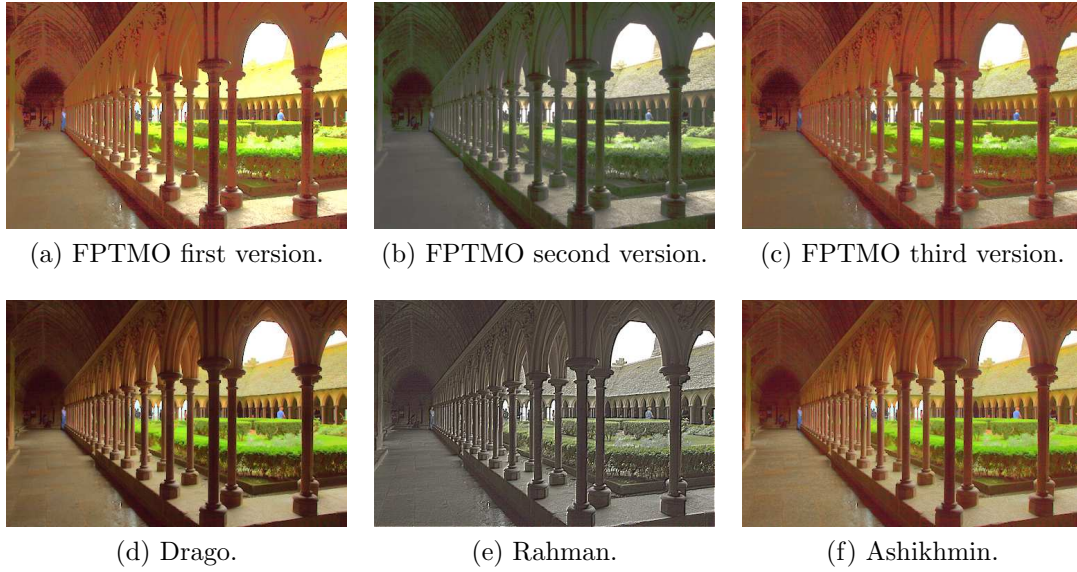
Figure 4.11: All FPTMO versions and other DTMOs (Drago, Rahman and Ashikhmin) applied to the Courtyard image.

4.3 Noise Considerations

Camera sensor noise affects tone-mapping operators. The camera sensor noise floor is defined as the non-zero value registered by the camera sensor when no light is striking it. Some images have dark regions, in which the raw pixels have very low values that are close to the camera sensor noise floor. This is the case, for example, for the hallway of the Courtyard image. When the input image of the tone-mapping operators is obtained through a combination of multiple exposures of the same scene, noise tends to be reduced, because it is averaged over the multiple exposures. However, as reported in [42], noise becomes particularly a problem when only a single exposure serves as input to the tone-mapping operator, because such averaging is not performed. Therefore, depending on the characteristics of the applied tone-mapping curve, a salt-and-pepper noise may arise, specially in these darker regions, thereby degrading image quality.

To suppress noise effects, an additional low-pass filtering stage may be required after demosaicing. Noise is not observed in the images shown in Appendix C and in this chapter, because they were pre-filtered by a low-pass bicubic filter, and then rescaled to 10% of their original sizes. Figures 4.12 and 4.13 show non-filtered and filtered versions of windows cropped from the original resolution (3273×4916) Courtyard image. They were processed by the FPTMO third version and by the Rahman operator. The low-pass filter used is a simple 6x6 moving average filter. Small windows are shown rather than the full resolution images, because image viewing software usually cannot show the image in the original resolution if the original resolution is too big. The software usually exhibits a resized version instead,

(a) Non-filtered ceiling third version of FPTMO.

(b) Non-filtered hallway third version of FPTMO.

(c) Non-filtered pillar third version of FPTMO.

(d) Filtered ceiling third version of FPTMO.

(e) Filtered hallway third version of FPTMO.

(f) Filtered pillar third version of FPTMO.
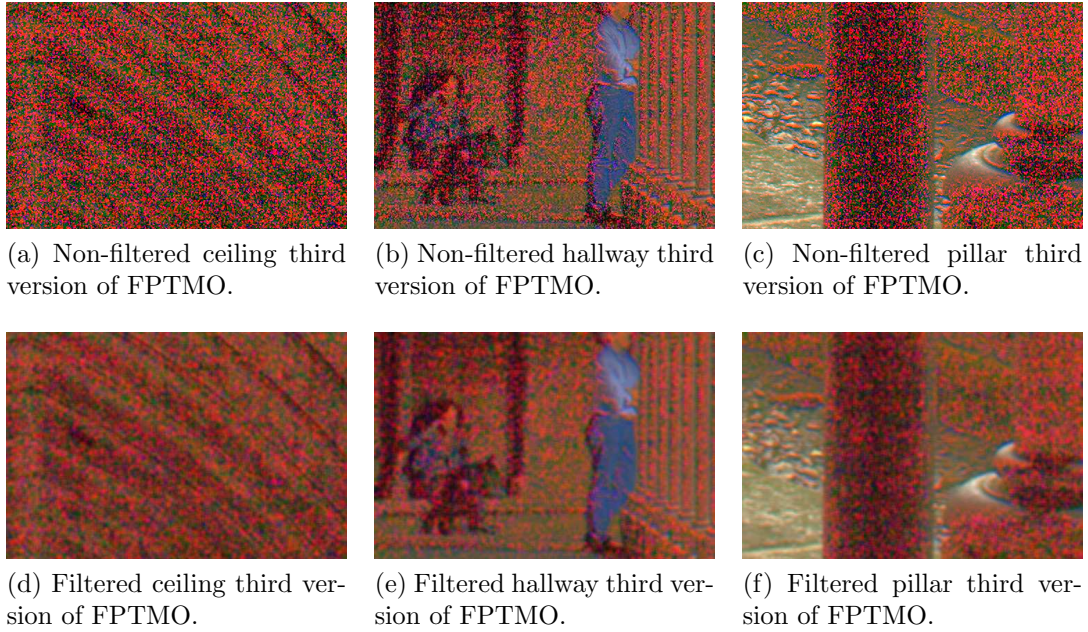
Figure 4.12: Non-filtered and filtered versions of windows cropped from the Courtyard image in original resolution (3273×4916), generated after the application of the FPTMO third version.

(a) Non-filtered ceiling Rahman operator.

(b) Non-filtered hallway Rahman operator.

(c) Non-filtered pillar Rahman operator.

(d) Filtered ceiling Rahman operator.

(e) Filtered hallway Rahman operator.

(f) Filtered pillar Rahman operator.
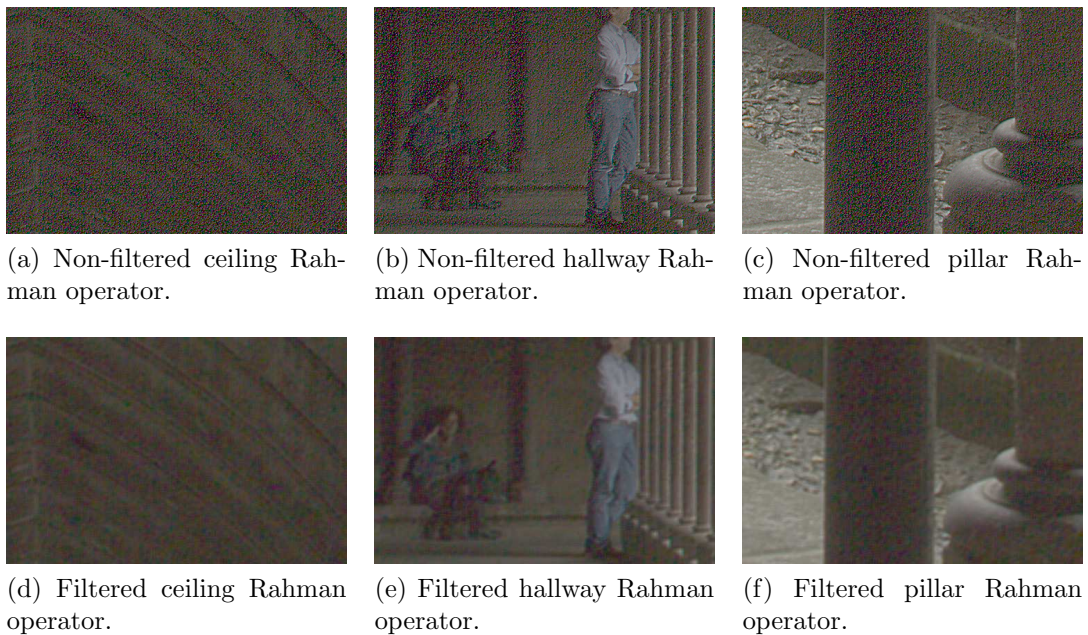
Figure 4.13: Non-filtered and filtered versions of windows cropped from the Courtyard image in original resolution (3273×4916), generated after the application of the Rahman operator.

which is already pre-filtered.

4.4 Objective Quality Assessment

The images resulting from every tone-mapping operator were also evaluated by an objective quality assessment algorithm called *Tone-Mapped image Quality Index* (TMQI) [43]. This algorithm takes into account the different dynamic ranges of the tone-mapped image and the original HDR image when comparing them. It predicts the score a person would give to the tone-mapped image, based on its overall quality. In order to calculate the score, the algorithm uses a combination of two other image quality assessment methods, namely a modified version of the structural similarity (SSIM) [44] and the natural scene statistics (NSS) [45]. A limitation of the algorithm is that color information is not considered for the calculation of the scores, since the algorithm is applied on the original and tone-mapped luminance channels. The reader is referred to [43] for more details on algorithm implementation.

Table 4.2 shows the scores of the images resulting from every tone-mapping operator. Figure 4.14 shows two graphical representations of the data in the table: the errorbar plot and the boxplot. The errorbar helps quantifying the average performance of each operator, by showing the average value (denoted by the circle) and the associated standard deviation (denoted by the bars) of the scores distribution from each operator. The boxplot gives a better indication of how these scores are distributed for each operator. The bottom and top edges of the boxes denote, respectively, the 25th and 75th percentile of the scores distribution from the corresponding operator and the red bar denotes its median value. The dark bars represent the range of score values that are not regarded as outliers, while the red crosses correspond to outliers.

The scores lie within the [0,1] interval, with 1 and 0 representing the best and the worst quality, respectively. The Ashikhmin operator achieves the highest average score computed over the ten HDR images. Considering each image individually, this operator achieves the highest score in three images: Flowers (0.973), Courtyard (0.947) and Beach (0.981). For the rest of the images, different operators achieve the highest scores. The results suggest that there is no operator which is the best for every image, as can be inferred from the plots in Figure 4.14. The results also suggest that different operators can perform equally well in a given image. All FPTMO versions achieve high scores that are also similar to scores of other digital tone-mapping competitors.

To check the statistical significance of the differences observed between each operator TMQI score distribution, a preliminary analysis using four versions of Student's T test was performed. The null hypothesis is that all distributions are equal. In all versions, the TMQI score distributions from every operator are compared pairwise. For a given pair of distributions, the test results indicate whether both
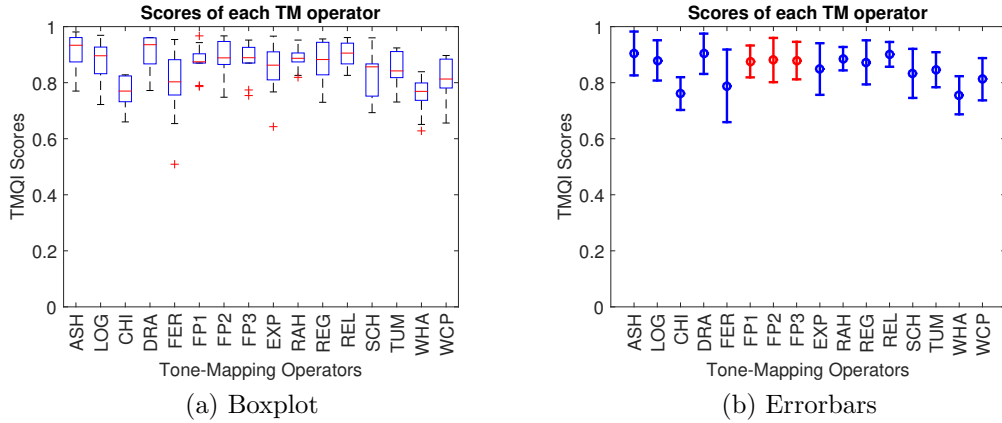
Figure 4.14: Plots of the data from Table 4.2, showing the performance of each operator, using the TMQI algorithm.

distributions are equal. The first and second versions assume that there is no relationship between the scores obtained from different operators for a given image. These versions are called independent sample T test. The first version considers a two-tailed test, while the second version considers an one-tailed test. The third and fourth versions, called dependent sample T test, assume that there is some relationship between the scores of different operators for a given image. The third and fourth versions consider a two-tailed test and an one-tailed test, respectively. For every version, the confidence interval considered was 95%.

In the first version, 82% of pairs corresponded to two equal distributions (that is, the null hypothesis was accepted in 82% of the cases). In the second version, this percentage is equal to 71%. In the third version, 66% of pairwise comparisons indicated equal distributions. In the fourth version, this percentage is equal to 51%. From the graphics of Figure 4.14, a higher percentage of pairs would be expected to correspond to two equal distributions in the third and fourth versions of this test. A more careful analysis of these preliminary results and the methodology used to obtain them must be performed before definitive conclusions can be drawn. In all T test versions, almost all pairwise comparisons between each FPTMO version and every DTMO indicated equal distributions. The only pair that indicated different distributions was the one corresponding to the third FPTMO version and the Ashikhmin operator, in the third T test version.

Global (and particularly simple) operators, like Drago and Logarithm operators, achieve scores in each image that are close to (or even higher than) scores obtained by local and more complex operators, like Reinhard Local and Rahman operators. This trend is also observed in [22]: two subjective tests (with and without the reference image) were conducted in order to assess the performance of some tone-mapping operators on three HDR images. The subjects evaluated the images under different

Operator	Parasol	Fire Ext.	Flowers	Sunset	Courtyard	Color C. 1	Beach	Palace	Stone T.	Color C. 2	Mean
Ashikhmin	0.961	0.893	**0.973**	0.874	**0.947**	0.771	**0.981**	0.952	0.920	0.770	**0.904**
Logarithm	0.905	0.928	0.885	0.832	0.887	0.722	0.969	0.924	0.927	0.816	0.880
Chiu	0.686	0.782	0.732	0.826	0.660	0.824	0.768	0.828	0.772	0.733	0.761
Drago	0.940	0.952	0.961	0.867	0.902	0.786	0.960	0.960	0.931	0.772	0.903
Ferwerda	0.779	0.954	0.801	0.509	0.756	0.654	0.837	0.910	0.882	0.805	0.789
FPTMO First V.	**0.967**	0.903	0.871	0.870	0.943	0.786	0.879	0.873	0.790	0.876	0.876
FPTMO Second V.	0.886	0.865	0.870	0.891	0.922	0.748	0.963	**0.967**	0.947	0.748	0.881
FPTMO Third V.	0.870	0.925	0.870	0.893	0.939	0.774	0.885	0.952	0.926	0.754	0.879
Exponential	0.859	**0.966**	0.873	0.861	0.810	0.643	0.910	0.864	0.934	0.767	0.849
Rahman	0.889	0.893	0.885	**0.936**	0.876	0.819	0.874	0.826	**0.952**	0.906	0.886
Reinhard G.	0.944	0.932	0.956	0.856	0.857	0.730	0.945	0.828	0.908	0.769	0.873
Reinhard L.	0.904	0.867	0.884	0.907	0.850	**0.826**	0.948	0.961	0.941	**0.922**	0.901
Schlick	0.864	0.849	0.867	0.866	0.843	0.712	0.925	0.960	0.693	0.752	0.833
Tumblin	0.845	0.839	0.920	0.869	0.818	0.731	0.924	0.824	0.911	0.781	0.846
Ward Hist. Adj.	0.839	0.784	0.819	0.771	0.737	0.651	0.799	0.756	0.767	0.628	0.755
Ward Con. Pre.	0.740	0.894	0.897	0.798	0.808	0.656	0.847	0.818	0.781	0.884	0.812
Mean	0.867	0.889	0.879	0.839	0.847	0.740	0.901	0.888	0.874	0.793	0.852

Table 4.2: TMQI scores of every tone-mapping operator for each HDR image. Boldface indicates the highest score achieved in a given image.

aspects defined by the authors, like, for example, overall quality, overall brightness and others. In both experiments, global operators achieved higher scores than local operators under the "overall quality" aspect (which is defined as "how close the overall match in appearance is to the real-world scene"), including a simple linear clipping operator developed by the authors.

Another algorithm used to objectively assess the quality of the resulting images is called *HDR Visual Difference Predictor* (HDR-VDP) [46]. This algorithm estimates the probabilities of detecting visual differences between two images (both HDR or LDR) at each pixel and gives an overall quality score to the tested image, based on the magnitude of the visual differences. The algorithm takes advantage of human vision aspects, such as contrast perception under different illumination conditions, in order to predict these probabilities. It assumes that pixel values of both images represent absolute luminance values (calibrated in cd/m^2). Each image is then normalized to the [0,1] range and multiplied by a calibration factor L_{max}, before the algorithm is applied on them. This algorithm does not consider colors either. The scores range from 0 (totally different - worst quality) and 100 (no differences - best quality).

Since the real scene illumination conditions are unknown, the algorithm was executed using different values for the calibration factor L_{max}. This algorithm scores for each tone-mapping operator using $L_{max} = 1$, $L_{max} = 100$ and $L_{max} = 10000$ are shown in Tables 4.3, 4.4 and 4.5, respectively. The corresponding errorbar plots and boxplots are shown in Figures 4.15, 4.16 and 4.17. Under low illumination conditions, the human eye has lower visual acuity, and is thus less capable of discerning details and visual differences between two scenes. Under these conditions, the tone-mapped images look more similar to the original HDR images. Distortions induced by the tone-mapping operators are less likely to be visible, thereby leading to higher scores for each operator. As scenes become more illuminated, the differences caused by the tone-mapping operation in the resulting images also become more evident, thus generally decreasing operator score. The HDR-VDP results also indicate that it is difficult to define one best tone-mapping operator for every possible different scene (and even for the same scene, when considering it under different illumination conditions). In all scenarios that were tested using HDR-VDP, the performances of all FPTMO versions and of all DTMOs are similar as well.

Besides directly assessing the resulting image quality through objective evaluators, the *visual-attention* (or *saliency*) *maps* of the raw and each tone-mapped image were also estimated and compared. Visual-attention maps show the salient regions of the images, in which the attention of the viewer tends to be mostly fixed. Such regions can be interpreted as representative regions of the image. Since the original scene with all its details cannot be exhibited on a conventional display wit-

Operator	Parasol	Fire Ext.	Flowers	Sunset	Courtyard	Color C. 1	Beach	Palace	Stone T.	Color C. 2	Mean
Ashikhmin	60.91	58.88	62.38	60.01	58.77	63.86	58.86	59.89	60.41	63.47	60.74
Logarithm	58.12	59.22	57.51	59.38	58.42	63.67	58.71	59.40	59.39	57.71	59.15
Chiu	55.78	57.30	56.78	58.47	55.32	54.41	54.99	55.64	55.36	54.27	55.83
Drago	59.80	61.44	60.75	60.29	58.14	59.02	57.54	58.44	59.01	59.34	59.38
Ferwerda	56.10	57.24	58.65	56.18	52.81	58.51	54.67	56.27	56.47	52.59	55.95
FPTMO First V.	60.35	60.99	63.61	58.33	56.50	58.27	55.50	59.84	55.56	58.02	58.70
FPTMO Second V.	58.85	58.47	64.02	62.64	58.36	**68.24**	59.12	57.96	58.01	64.62	61.03
FPTMO Third V.	58.05	59.81	64.26	62.23	58.01	61.88	55.19	57.12	55.96	61.05	59.36
Exponential	58.58	58.52	59.68	60.91	55.86	62.63	57.51	58.93	58.85	56.52	58.80
Rahman	62.91	64.67	63.01	63.43	58.39	58.90	62.85	64.40	59.19	59.64	61.74
Reinhard G.	58.51	59.66	62.07	59.06	55.54	58.80	56.39	61.07	57.79	59.88	58.88
Reinhard L.	59.12	62.38	64.45	60.17	55.78	54.05	57.25	57.72	58.83	55.33	58.51
Schlick	57.83	63.84	58.66	58.67	55.26	59.68	56.53	56.53	66.47	59.24	59.27
Tumblin	58.77	64.54	65.70	58.09	**59.36**	62.87	58.57	63.03	58.71	60.17	60.98
Ward Hist. Adj.	**67.22**	**66.32**	**67.52**	**72.08**	58.06	65.36	**66.49**	**75.40**	**67.46**	**74.17**	**68.01**
Ward Con. Pre.	54.89	59.39	61.95	57.78	53.63	57.94	55.14	58.76	55.43	53.44	56.83
Mean	59.11	60.79	61.94	60.48	56.76	60.50	57.83	60.02	58.93	59.34	59.57

Table 4.3: HDR-VDP scores of every tone-mapping operator for each HDR image, using $L_{max} = 1$. Boldface indicates the highest score achieved in a given image.

Operator	Parasol	Fire Ext.	Flowers	Sunset	Courtyard	Color C. 1	Beach	Palace	Stone T.	Color C. 2	Mean
Ashikhmin	52.31	49.78	54.96	47.22	52.09	50.98	52.06	53.09	52.25	52.44	51.72
Logarithm	48.90	50.69	48.67	45.28	47.57	50.22	52.32	50.46	49.60	46.14	48.99
Chiu	51.29	51.40	54.77	51.35	50.47	48.72	53.81	50.30	53.17	48.58	51.39
Drago	49.21	54.03	53.28	47.83	51.49	46.27	49.64	51.71	49.17	49.68	50.23
Ferwerda	44.02	44.48	56.78	39.64	42.34	**54.32**	42.07	46.57	44.05	35.61	44.99
FPTMO First V.	48.71	51.53	**64.10**	47.06	45.78	45.10	47.63	54.93	43.88	48.15	49.69
FPTMO Second V.	50.85	49.90	63.92	51.47	50.25	50.86	50.85	51.43	49.14	51.78	52.05
FPTMO Third V.	49.10	52.09	63.86	51.06	49.39	46.30	46.70	50.10	45.61	50.80	50.50
Exponential	50.08	49.17	54.09	48.74	48.57	52.20	49.36	53.98	48.21	47.19	50.16
Rahman	57.74	57.09	60.34	54.40	51.88	51.36	**58.47**	56.37	**54.20**	52.59	55.44
Reinhard G.	52.88	53.01	55.91	48.03	47.81	50.06	47.25	55.96	47.15	51.89	50.99
Reinhard L.	50.89	57.39	60.76	48.63	48.31	47.82	49.39	50.81	48.96	44.73	50.77
Schlick	48.22	58.73	57.22	47.02	46.00	45.33	46.47	47.92	52.69	50.09	49.97
Tumblin	50.82	**60.96**	61.88	46.67	**54.00**	52.87	50.20	57.87	47.18	51.20	53.36
Ward Hist. Adj.	**58.33**	57.98	57.42	**57.04**	52.41	52.39	53.97	**59.85**	52.97	**55.46**	**55.78**
Ward Con. Pre.	44.32	50.70	62.41	45.96	45.68	52.07	43.54	55.56	43.45	39.01	48.27
Mean	50.48	53.06	58.15	48.59	49.00	49.80	49.61	52.93	48.86	48.46	50.89

Table 4.4: HDR-VDP scores of every tone-mapping operator for each HDR image, using $L_{max} = 100$. Boldface indicates the highest score achieved in a given image.

Operator	Parasol	Fire Ext.	Flowers	Sunset	Courtyard	Color C. 1	Beach	Palace	Stone T.	Color C. 2	Mean
Ashikhmin	51.17	49.52	52.98	47.28	51.50	54.05	51.79	52.90	52.07	52.47	51.57
Logarithm	48.38	50.49	48.29	45.70	53.41	53.32	52.80	52.65	52.15	45.49	50.27
Chiu	50.27	50.85	54.29	53.36	50.61	53.20	52.06	51.82	**54.01**	**54.47**	52.49
Drago	48.54	53.94	52.92	47.13	52.92	44.52	49.61	51.64	48.96	49.75	49.99
Ferwerda	43.21	43.71	56.92	38.28	41.23	53.01	41.20	45.98	43.35	33.93	44.08
FPTMO First V.	47.21	49.59	66.63	46.63	44.91	42.14	47.53	52.71	43.27	45.95	48.66
FPTMO Second V.	49.52	49.41	**67.28**	51.87	48.37	49.95	49.33	51.43	48.49	53.93	51.96
FPTMO Third V.	47.98	50.93	67.11	51.59	45.77	43.16	46.40	49.47	45.17	50.92	49.85
Exponential	49.83	48.66	53.90	47.91	49.68	54.56	49.52	54.18	47.93	46.85	50.30
Rahman	58.02	56.28	61.11	**55.00**	53.62	54.90	**56.92**	53.49	53.79	52.63	**55.58**
Reinhard G.	52.86	52.97	55.62	47.75	48.51	54.00	47.16	56.82	46.91	53.42	51.60
Reinhard L.	50.51	57.54	60.90	48.06	49.48	49.11	49.49	50.76	48.76	44.19	50.88
Schlick	47.82	58.83	57.51	46.41	45.51	43.68	46.30	47.82	44.51	50.09	48.85
Tumblin	50.54	61.18	61.52	46.03	**57.33**	**56.81**	50.37	**59.14**	46.89	51.42	54.12
Ward Hist. Adj.	**64.21**	**62.45**	63.23	50.46	45.33	47.82	55.46	57.97	50.85	47.00	54.48
Ward Con. Pre.	44.03	50.30	63.54	45.06	45.66	55.82	43.22	56.06	42.74	37.29	48.37
Mean	50.26	52.91	58.98	48.03	48.99	50.63	49.32	52.80	48.12	48.11	50.82

Table 4.5: HDR-VDP scores of every tone-mapping operator for each HDR image, using $L_{max} = 10000$. Boldface indicates the highest score achieved in a given image.

(a) Boxplot

(b) Errorbars
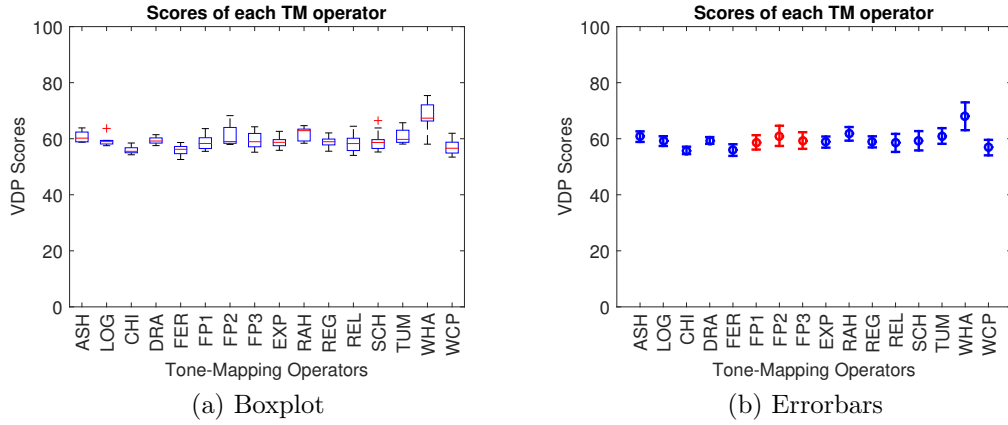
Figure 4.15: Plots of the data from Table 4.3, showing the performance of each operator, using the HDR-VDP algorithm and a calibration factor $L_{max} = 1$.

(a) Boxplot

(b) Errorbars
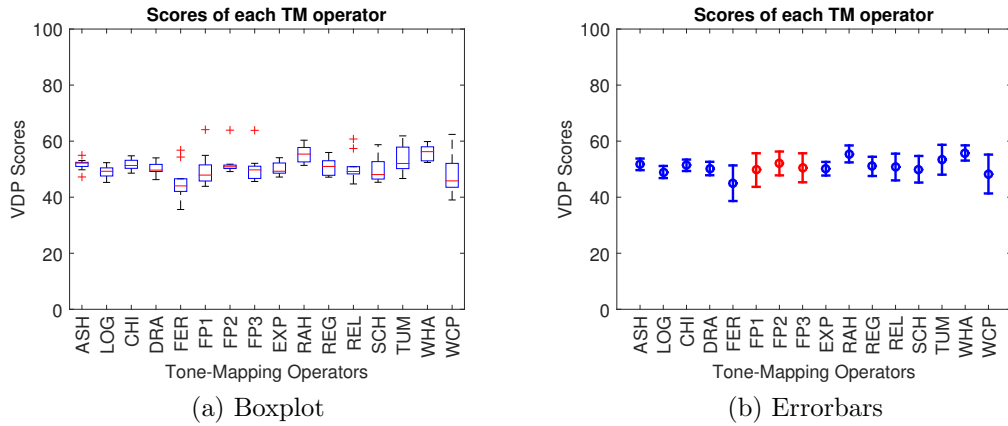
Figure 4.16: Plots of the data from Table 4.4, showing the performance of each operator, using the HDR-VDP algorithm and a calibration factor $L_{max} = 100$.

(a) Boxplot

(b) Errorbars
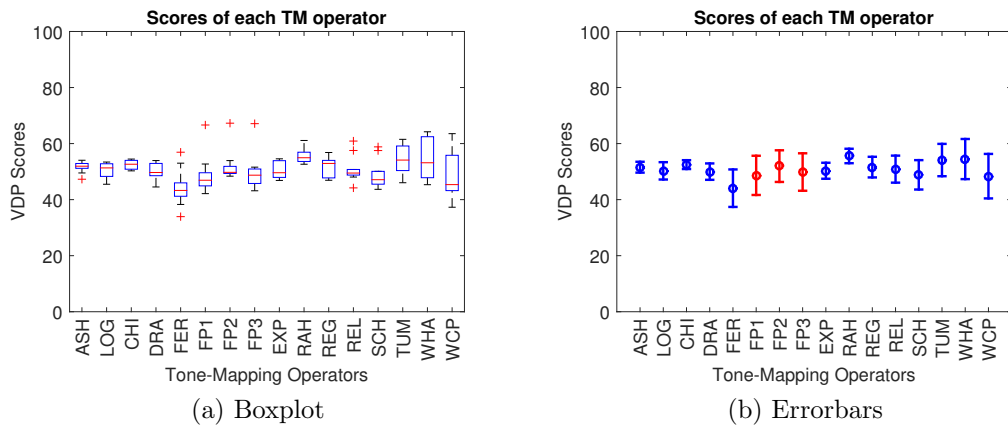
Figure 4.17: Plots of the data from Table 4.5, showing the performance of each operator, using the HDR-VDP algorithm and a calibration factor $L_{max} = 10000$.

hout the previous application of a tone-mapping operator on it, the visual-attention maps provide a way of representing the original scene before any signal processing

121

task takes place. They also give an idea of how each tone-mapping operator affects the original scene by possibly showing different salient regions in the resulting images. Visual-attention maps and image visual quality are aspects that affect the user overall quality of experience [47]. A relationship between image quality and visual-attention maps possibly exists, so the use of visual-attention maps as objective image quality evaluator is justified.

There exists a number of algorithms that attempt to predict the salient regions of an image. Their performances are evaluated using benchmark datasets, which contain the images of original scenes and their corresponding visual-attention maps estimated from several different observers. The reader is referred to [48] and [49] for more details on saliency detection algorithms as well as metrics and benchmark datasets used to evaluate them. In the present work, the visual-attention maps are generated by the context-aware saliency detection algorithm [50]. The mean squared errors (MSEs) between the saliency map of the raw and each tone-mapped image are calculated, in order to quantify the differences between them. The MSE values are then normalized by the maximum MSE that is found (which is $\mathrm{MSE_{max}} = 22152$).

Table 4.6 shows the MSE values associated with each operator for each different image and Figure 4.18 shows the errorbar plot and boxplot of this data. Results suggest that, on average, no tone-mapping operator has particularly more tendency of changing or preserving the original saliency maps than the others. This observation is valid for the images resulting from all DTMOs and all FPTMO versions. As stated before, saliency roughly indicates how each tone-mapping operator affects the original image, but no judgement concerning the tone-mapped image quality can be made by only comparing the resulting visual-attention map to the corresponding original visual-attention map. Low and high MSE do not necessarily correspond to worst and best image quality, respectively, since there is no clear relation between salient regions and image overall quality (e.g. tone-mapping operators may introduce or remove artifacts from certain regions of the original image, or make details more or less visible in such regions, thereby leading to a possibly different resulting visual-attention map).

4.5   Complexity Analysis Results

In this section, the overall processing times of each tone-mapping operator implemented in this work are presented. These times are calculated considering two scenarios. The first scenario assumes that all tone-mapping operators, including the three FPTMO versions, are digitally implemented and run on the same digital hardware outside the focal plane. The second scenario considers that the FPTMO second and third versions are implemented in hardware (i.e. at the focal plane of a

Operator	Parasol	Fire Ext.	Flowers	Sunset	Courtyard	Color C. 1	Beach	Palace	Stone T.	Color C. 2	Mean
Ashikhmin	0.281	0.112	0.698	0.315	0.280	0.225	0.192	0.293	0.272	0.304	0.297
Chiu	0.517	0.093	1.000	0.283	0.141	0.236	0.261	0.298	0.231	0.147	0.321
Drago	0.326	0.118	0.810	0.320	0.283	0.208	0.202	0.310	0.271	0.320	0.317
Exponential	0.382	0.121	0.951	0.410	0.341	0.223	0.308	0.358	0.299	0.311	0.370
Ferwerda	0.205	0.110	0.468	**0.195**	**0.085**	0.208	0.399	0.339	0.279	**0.056**	**0.234**
FPTMO First V.	0.286	0.124	0.530	0.371	0.219	0.210	0.232	0.338	0.279	0.238	0.283
FPTMO Second V.	0.190	0.088	0.547	0.371	0.387	0.168	0.371	0.256	0.259	0.322	0.296
FPTMO Third V.	0.392	0.150	0.538	0.454	0.304	0.211	0.254	0.314	0.269	0.313	0.320
Logarithm	0.377	0.119	0.857	0.294	0.285	0.190	0.211	0.267	0.278	0.264	0.314
Rahman	**0.149**	0.127	**0.407**	0.291	0.403	0.173	0.311	0.253	0.253	0.249	0.262
Reinhard G.	0.205	**0.075**	0.548	0.293	0.188	**0.137**	**0.183**	0.307	**0.221**	0.291	0.245
Reinhard L.	0.343	0.153	0.595	0.379	0.258	0.204	0.198	0.342	0.280	0.184	0.294
Schlick	0.393	0.164	0.884	0.413	0.110	0.187	0.229	0.309	0.294	0.331	0.331
Tumblin	0.351	0.167	0.605	0.395	0.392	0.233	0.353	0.302	0.305	0.313	0.342
Ward Con. Pre.	0.466	0.202	0.643	0.363	0.204	0.258	0.345	0.373	0.291	0.118	0.326
Ward Hist. Adj.	0.182	0.158	0.419	0.480	0.560	0.253	0.366	**0.250**	0.283	0.368	0.332
Mean	0.315	0.130	0.656	0.352	0.278	0.208	0.276	0.307	0.273	0.258	0.305

Table 4.6: MSE values between the saliency maps of the raw and each tone-mapped image. Boldface indicates the lowest MSE value achieved in a given image.

(a) Boxplot	(b) Errorbars
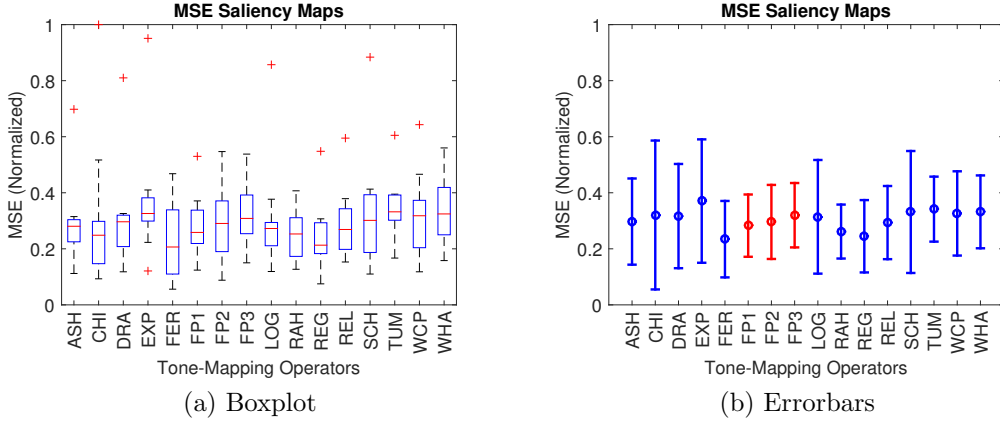
Figure 4.18: Data from Table 4.6.

CMOS image sensor), while the FPTMO first version and the other digital operators are implemented in software, as in the first scenario. In either scenario, each operator processing time is obtained from the complexity analysis expressions in Chapter 3.

In order to calculate the execution times using the complexity analysis expressions, some constants must be defined. In both scenarios, the camera sensor array is assumed to have M = 1024 rows and N = 2048 columns (thus yielding images of size 1024 x 2048) and that there is one ADC per column of the sensor array (which corresponds to $N_{ADC} = 2048$). As stated in Chapter 3, the constant K_{Ramp} (for either 8-bit or 12-bit ramp ADC) depends on the ramp ADC clock median reported in [39] and on the circuit global clock, which is defined as the maximum time required to complete the tone-mapping operation in the focal plane $\tau_{clk} = T_{max}$. The median conversion rate reported in [39] for the 8-bit ramp ADC is approximately 4.5×10^5 samples per second, which corresponds to a median conversion time of 2.2 μs. Using this value in Equation (3.52) yields a median operating clock value equal to 8.59 ns. By substituting this value in Equation (3.53) and assuming that $T_{max} = 1$ ms, the constant K_{Ramp_8bits} is then equal to 8.59×10^{-6}.

For the 12-bit ramp ADC, the median conversion rate from Figure 3.41 is approximately 3×10^4 samples per second. This leads to a median conversion time of 33 μs and, consequently from Equation (3.52), to a median operating clock of approximately 8 ns. Using this value and $\tau_{clk} = T_{max} = 1$ ms in Equation (3.53) yields $K_{Ramp\_12bits} = 8 \times 10^{-6}$. The constant K_{DP} is obtained from Equation (3.55), considering an arbitrarily chosen digital processor clock period of 0.25 ns (or, equivalently, a clock frequency of 4 GHz) and $T_{max} = 1$ ms, thus leading to $K_{DP} = 2.5 \times 10^{-7}$. The constant values are summarized in Table 4.7.

The constants $N_{clk\_cycles\_TM}$ and $N_{clk\_cycles\_CT}$ are the number of clock cycles required to complete the execution of the tone-mapping operation and additional color

Constant	Value
K_{Ramp_8bits}	8.59×10^{-6}
K_{Ramp_12bits}	8×10^{-6}
K_{DP}	2.5×10^{-7}
M	1024
N	2048
N_{ADC}	1024

Table 4.7: Constant values for complexity analysis expressions from Chapter 3.

treatment operations, respectively. The tone-mapping and color treatment operations are assumed to be implemented in software rather than in hardware. These constant values depend on the tone-mapping operator considered and they are determined according to the digital complexity analysis procedure described in Chapter 3. Table 4.8 shows the number of clock cycles and the corresponding time (assuming a digital clock period of 0.25 ns) demanded by the operations that are performed after tone mapping to include color in the resulting images. For the FPTMO first version and DTMOs that are executed on the input image luminance channel (i.e. executed on grayscale images), the operation to obtain the tone-mapped color channels consists in scaling the original color channels by the ratio between the tone-mapped and original luminance values defined for each pixel.

For the FPTMO second version, the operations executed after tone mapping are the following ones: raw value reconstruction, white-balance gain calculation and multiplication by the tone-mapped image in the Bayer pattern, and demosaicing. In the first scenario, in which this operator is digitally implemented, the raw value reconstruction must not be performed, because the raw image pixel values are available to all operators. Consequently, the total processing time required by this operator to yield a color tone-mapped image does not take into account the time cost of the raw value reconstruction operation. In the second scenario, in which this FPTMO version is implemented in the focal plane, the original raw values are not available. They must be reconstructed, to allow for correct white-balance gain computations for each pixel, as discussed in Chapter 3. Therefore, in the second scenario, this operator overall processing time must include the execution time of the raw value reconstruction operation. For the FPTMO third version, the color treatment operation consists in multiplying the tone-mapped image in the Bayer pattern by the corresponding white-balance gains obtained from the DCRaw software, and then performing demosaicing.

Operation	$N_{clk\_cycles\_CT}$	Time (ms)	Required By
Raw rec.	9.65×10^6	2.41	FPTMO second version
Color treat. 1	2.16×10^8	54.00	FPTMO first version and DTMOs defined for grayscale images
Color treat. 2	6.96×10^8	174.00	FPTMO second version
Color treat. 3	2.74×10^8	68.50	FPTMO third version

Table 4.8: Number of clock cycles and the associated execution time of post-processing operations. The last column identifies to which operator the corresponding post-processing operation is associated.

4.5.1 First Scenario

In this scenario, an HDR picture is assumed to be captured, converted to twelve bits and then tone-mapped by each operator. Additional color processing operations after tone mapping are also considered. Since the capture and the image analog-to-digital conversion process is the same for all operators (that is, the same image-processing cascade from Figure 3.40 is applied to all operators), the focus in this scenario is on evaluating the execution times of each tone-mapping algorithm.

Table 4.9 shows the execution times of the digital implementations of each tone-mapping operator. The T_{proc_gray} column shows the time required to produce grayscale images, that is, considering only the tone-mapping operation execution time and the analog-to-digital conversion time. The T_{proc_color} column shows the overall processing time, which includes the execution times of the additional operations performed, by each operator, in order to generate color images. The T_{ADC} column shows the time demanded to digitally convert all pixels of the sensor array, using 12-bit SAR ADCs. If 12-bit ramp ADCs are considered instead, then the values from the T_{ADC} column are all equal to 33.55 ms. Since this only corresponds to increasing all the values from the T_{proc_gray} and T_{proc_color} columns by an offset of $33.55 - 1.38 = 32.17$ ms, the ranking between the operators (regarding their execution times) is not changed. An additional table for the execution times, considering 12-bit ramp ADCs, is thus not shown.

Operators, such as Ashikhmin and Rahman operators, that produce images whose overall quality is good and achieve high average scores in the TMQI and VDP metrics also demand particularly high execution times. As indicated by the results from Table 4.9, local tone-mapping operators tend to demand more execution time than global ones, since they usually perform multiple filtering operations, in order to determine the contextual information of each pixel. Drago and Logarithm operators, which are both global and also achieve high scores in the TMQI and VDP metrics, are not among the fastest tone-mapping operators, despite their simplicity, because of the relatively high estimated cost for the logarithm operation.

Operators	$N_{clk\_cycles\_TM}$	T_{TM} (ms)	T_{ADC} (ms)	T_{proc_gray} $=$ $T_{TM}+T_{ADC}$ (ms)	T_{CT} (ms)	T_{proc_color} $=$ $T_{TM}+T_{ADC}+T_{CT}$ (ms)
FPTMO Third V.	4.07×10^8	101.75	1.38	103.13	68.50	171.63
Exponential	8.61×10^8	215.25	1.38	216.63	54.00	270.63
FPTMO Second V.	3.95×10^8	98.75	1.38	100.13	174.00	274.13
FPTMO First V.	8.76×10^8	219.00	1.38	220.38	54.00	274.38
Ferwerda	1.12×10^9	280.00	1.38	-	-	281.38
Reinhard G.	1.22×10^9	305.00	1.38	-	-	306.38
Schlick	1.07×10^9	267.50	1.38	268.88	54.00	322.88
Ward Con. Pre.	1.08×10^9	270.00	1.38	271.38	54.00	325.38
Logarithm	1.44×10^9	360.00	1.38	361.38	54.00	415.38
Drago	1.45×10^9	362.50	1.38	363.88	54.00	417.88
Tumblin	1.49×10^9	372.50	1.38	373.88	54.00	427.88
Ward Hist. Adj.	5.65×10^9	1,412.50	1.38	1,413.88	54.00	1,467.88
Ashikhmin	7.13×10^9	1,782.50	1.38	1,783.88	54.00	1,837.88
Chiu	9.45×10^9	2,362.50	1.38	2,363.88	54.00	2,417.88
Reinhard L.	9.46×10^{10}	23,650.00	1.38	23,651.38	54.00	23,705.38
Rahman	1.59×10^{11}	39,750.00	1.38	-	-	39,751.38

Table 4.9: Execution times for all the tone-mapping operators, considering that they are all digitally implemented (following the block diagram in Figure 3.40) and that the camera internal circuit uses SAR ADCs. The operators are sorted in ascending T_{proc_color} order.

All three FPTMO digital implementations have low execution times, since their corresponding tone-mapping operations are not highly complex and do not involve costly operations either. Regarding grayscale images, the FPTMO second version has the lowest execution time (98.75 ms), but because of the relatively high execution time of its color treatment operations, it has the third lowest execution time for producing color tone-mapped images. The high time cost of its color treatment operations is mainly caused by the white-balance gain calculation. Regarding color images, the FPTMO third version has the lowest execution time (171.63 ms). Aside from the Exponential operator, all three FPTMO versions are faster than the other DTMOs implemented in this work, considering the generation of either grayscale or color tone-mapped images. Figure 4.19 shows the performance of each tone-mapping operator considering the corresponding execution time for generating a color tone-mapping image in this first scenario and the corresponding average TMQI score.

4.5.2 Second Scenario

In this scenario, the FPTMO second and third versions are assumed to be implemented in hardware. So, they are implemented according to the block diagram in Figure 3.39 (capturing an HDR picture, performing the tone-mapping operation on it, digitizing the tone-mapped image using eight bits, and then executing the additional operations to include color in the digital tone-mapped image). The con-
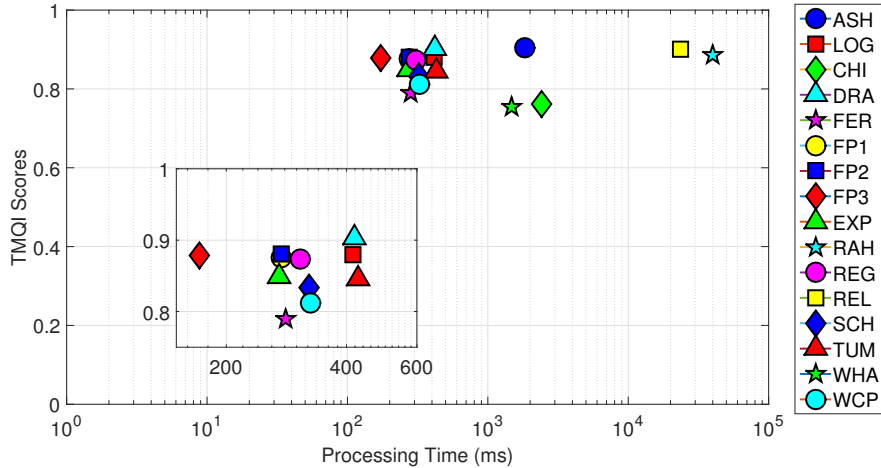
Figure 4.19: Average TMQI score from Table 4.2 versus processing time to generate a color tone-mapped image from Table 4.9 of each tone-mapping operator.

siderations for the FPTMO first version and the DTMOs are the same as in the first scenario: they are implemented in software according to the image-processing cascade defined in that scenario. The second scenario is devoted to the evaluation of the overall processing time required by tone-mapping operators implemented inside and outside the focal plane, thereby allowing to analyze the performance differences caused by implementing the tone mapping in hardware or in software.

Table 4.10 shows the execution times for each tone-mapping operator. The values from the column T_{ADC} were obtained assuming an 8-bit SAR ADC for the FPTMO second and third versions, and a 12-bit SAR ADC for the other operators. Table 4.11 shows the corresponding execution times, when ramp ADCs are considered, instead of SAR ADCs. In both cases, regarding the generation of grayscale tone-mapped images, the FPTMO second and third versions, which are implemented in the focal plane, are significantly faster than the other digital competitors. One of the reasons for that is because in the focal plane the raw image pixels are simultaneously tone mapped, thus requiring at most $T_{max} = 1$ ms to finish this operation. The analog-to-digital conversion is also faster when eight bits are used instead of twelve, especially for the ramp ADCs.

The Exponential operator with 12-bit SAR ADCs is the digital tone-mapping configuration that requires the shortest processing time until a grayscale tone-mapped image ready for display is obtained. In that statement, the tone mapping and analog-to-digital conversion times are both taken into account. The overall processing time for this configuration is 216.63 ms. The FPTMO second and third versions with 8-bit ramp ADCs, which is the configuration that demands the longest execution time for both versions to yield a grayscale tone-mapped image readily available for display, is 3.25 ms. This configuration is approximately 66 times faster

Operators	$N_{clk\_cycles\_TM}$	T_{TM} (ms)	T_{ADC} (ms)	T_{proc_gray} $=$ $T_{TM} + T_{ADC}$ (ms)	T_{CT} (ms)	T_{proc_color} $=$ $T_{TM} + T_{ADC} + T_{CT}$ (ms)
FPTMO Third V.	-	1.00	1.13	2.13	68.50	70.63
FPTMO Second V.	-	1.00	1.13	2.13	174.00	176.13
Exponential	8.61×10^8	215.25	1.38	216.63	54.00	270.63
FPTMO First V.	8.76×10^8	219.00	1.38	220.38	54.00	274.38
Ferwerda	1.12×10^9	280.00	1.38	-	-	281.38
Reinhard G.	1.22×10^9	305.00	1.38	-	-	306.38
Schlick	1.07×10^9	267.50	1.38	268.88	54.00	322.88
Ward Con. Pre.	1.08×10^9	270.00	1.38	271.38	54.00	325.38
Logarithm	1.44×10^9	360.00	1.38	361.38	54.00	415.38
Drago	1.45×10^9	362.50	1.38	363.88	54.00	417.88
Tumblin	1.49×10^9	372.50	1.38	373.88	54.00	427.88
Ward Hist. Adj.	5.65×10^9	1,412.50	1.38	1,413.88	54.00	1,467.88
Ashikhmin	7.13×10^9	1,782.50	1.38	1,783.88	54.00	1,837.88
Chiu	9.45×10^9	2,362.50	1.38	2,363.88	54.00	2,417.88
Reinhard L.	9.46×10^{10}	23,650.00	1.38	23,651.38	54.00	23,705.38
Rahman	1.59×10^{11}	39,750.00	1.38	-	-	39,751.38

Table 4.10: Execution times for all the tone-mapping operators, considering that the second and third versions of the FPTMO are implemented in the focal plane (following the block diagram in Figure 3.39), while the other operators are digitally implemented (following the block diagram in Figure 3.40). For the focal-plane implementations, the camera internal circuit uses 8-bit SAR ADCs. For the digital implementations, the camera internal circuit uses 12-bit SAR ADCs. The operators are sorted in ascending T_{proc_color} order.

than the fastest configuration that uses a DTMO. If the FPTMO second and third versions are used with 8-bit SAR ADCs, then they run approximately 100 times faster than the configuration that uses the Exponential operator with 12-bit SAR ADC.

The focal-plane implementations of the FPTMO second and third versions are also considerably faster than their corresponding first scenario digital implementations, regarding the generation of grayscale tone-mapped images that are readily available for display (i.e. digital 8-bit images). When considering the focal-plane implementations with 8-bit ramp ADCs and the digital implementations with 12-bit SAR ADCs, both focal-plane versions are approximately 30 times faster than their digital counterparts. When the focal-plane implementations are used with 8-bit SAR ADCs, they are about 50 times faster than their digital implementations with 12-bit SAR ADCs. If only the tone-mapping operations are considered (i.e. not taking into account the ADC processing time), then the focal-plane versions are about 100 times faster than their corresponding digital versions.

For generating color tone-mapped images, the focal-plane implementations are faster than the other digital operators, although the gain in processing time that occurs in this case is not as high as the gain in processing time that occurs when

Operators	$N_{clk\_cycles\_TM}$	T_{TM} (ms)	T_{ADC} (ms)	T_{proc_gray} = $T_{TM} + T_{ADC}$ (ms)	T_{CT} (ms)	T_{proc_color} = $T_{TM} + T_{ADC} + T_{CT}$ (ms)
FPTMO Third V.	-	1.00	2.25	3.25	68.50	71.75
FPTMO Second V.	-	1.00	2.25	3.25	174.00	177.25
Exponential	8.61×10^8	215.25	33.55	248.80	54.00	302.80
FPTMO First V.	8.76×10^8	219.00	33.55	252.55	54.00	306.55
Ferwerda	1.12×10^9	280.00	33.55	-	-	313.55
Reinhard G.	1.22×10^9	305.00	33.55	-	-	338.55
Schlick	1.07×10^9	267.50	33.55	301.05	54.00	355.05
Ward Con. Pre.	1.08×10^9	270.00	33.55	303.55	54.00	357.55
Logarithm	1.44×10^9	360.00	33.55	393.55	54.00	447.55
Drago	1.45×10^9	362.50	33.55	396.05	54.00	450.05
Tumblin	1.49×10^9	372.50	33.55	406.05	54.00	460.05
Ward Hist. Adj.	5.65×10^9	1,412.50	33.55	1,446.05	54.00	1,500.05
Ashikhmin	7.13×10^9	1,782.50	33.55	1,816.05	54.00	1,870.05
Chiu	9.45×10^9	2,362.50	33.55	2,396.05	54.00	2,450.05
Reinhard L.	9.46×10^{10}	23,650.00	33.55	23,683.55	54.00	23,737.55
Rahman	1.59×10^{11}	39,750.00	33.55	-	-	39,783.55

Table 4.11: Execution times for all the tone-mapping operators, considering that the second and third versions of the FPTMO are implemented in the focal plane (following the block diagram in Figure 3.39), while the other operators are digitally implemented (following the block diagram in Figure 3.40). For the focal plane implementations, the camera internal circuit uses 8-bit ramp ADCs. For the digital implementations, the camera internal circuit uses 12-bit ramp ADCs. The operators are sorted in ascending T_{proc_color} order.

considering the generation of grayscale tone-mapped images. This is because the color treatment operations associated with the focal-plane operators are implemented in software. The software implementation does not have the focal-plane parallel signal processing advantage.

The fastest digital configuration that yields color tone-mapped images is the one that uses the Exponential operator with 12-bit SAR ADCs, with an overall processing time equal to 270.63 ms. Considering that 8-bit ramp ADCs are used with both operators implemented in the focal plane, the FPTMO second version requires 179.66 ms to produce the color tone-mapped image. It is thus approximately 1.5 times faster than the fastest digital configuration. The FPTMO third version requires 71.75 ms (roughly 3.2 times faster than the fastest digital configuration). Using 8-bit SAR ADCs with the focal-plane implementations, the FPTMO second and third versions are, respectively, 1.5 times (178.54 ms) and 3.8 times (70.63 ms) faster than the fastest digital configuration.

In comparison to the overall processing time achieved by their corresponding digital implementations (Table 4.9), the focal-plane implementations of FPTMO versions 2 and 3 are faster as well. Considering 8-bit ADCs of either ramp or SAR types, the FPTMO second version focal-plane implementation is approximately 1.5
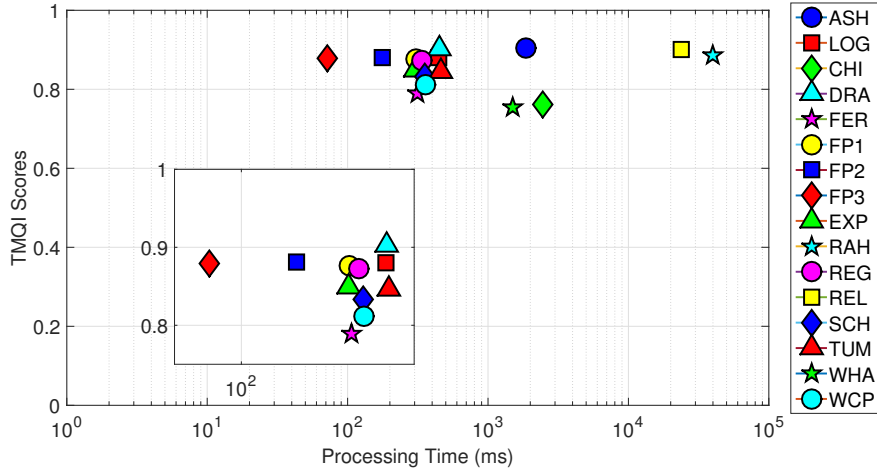
Figure 4.20: Average TMQI score from Table 4.2 versus processing time to generate a color tone-mapped image from Table 4.11 of each tone-mapping operator.

times faster than its digital counterpart, while the FPTMO third version focal-plane implementation is about 2.4 times faster than its digital counterpart. Figure 4.20 shows each tone-mapping operator performance, considering the associated average TMQI score and execution time required to yield a color tone-mapped image in this scenario, assuming that ramp ADCs are used.

Chapter 5

Conclusion

Based on the results presented in Chapter 4, the tone-mapping problem does not have one universally best solution, that is, no tone-mapping operator is always the best for all possible HDR scenes. This happens for several reasons. First, the definition of the best operator depends on what the purpose of the tone mapping is, as discussed in Chapter 2. The purpose can be, among others, to generate visually appealing images, or to generate images that accurately reproduce the same visual experience a person would have when observing the same scene in the real world. Some operators may focus on accomplishing one particular goal more than others. Second, even when the goal is established, it may be impossible to state that one operator performs better than any other operator for every single possible HDR image, especially when considering subjective preferences. The results obtained from different objective quality assessment algorithms also indicate the difficulty of selecting one best operator for all scenes and even for the same scene under different illumination conditions, as suggested by the VDP scores of the operators.

The results suggest that global tone-mapping operators are capable of producing images whose overall quality is comparable to the quality of the images generated by local tone-mapping operators. Particularly simple operators, such as Drago and Logarithm, can produce images with superior quality, regarding details and colors, than more complex operators, such as Reinhard Local and Rahman. This shows that the attributes that define the overall impression of quality in the image are not necessarily related to more intricate aspects, like, for example, local contrasts in some regions of the image. As a consequence, more complicated operations performed by local operators may not be required in order to generate high quality images.

Likewise, the results show that tone-mapping operators implemented in the focal plane generate images with quality similar to those produced by DTMOs. The FPTMO is not particularly better or worse than its digital competitors for any specific kind of image. Like the digital operators, its performance varies according to the image considered, as shown by the scores of different objective quality asses-

sment algorithms. The same observations made previously about the choice of one best operator are also valid for the FPTMO. As shown by the complexity analysis results, the main benefit of the FPTMO is that it runs faster than the other digitally implemented tone-mapping operators, since it benefits from the hardware implementation advantages.

5.1 Future Work

According to the complexity analysis results, for the real FPTMO implementation, most of the processing time required for generating color tone-mapped images is caused by the color treatment operations. Unlike the tone-mapping operation, the color treatment operations are not implemented in the focal plane. It would be interesting to study in more details the associated costs of implementing the color processing operations inside the focal plane, and also investigate other white-balance methods that automatically calculate the appropriate gains by using only information derived from the available pixel values. That would free the user or designer from making arbitrary assumptions, such as defining constant gains, or from using information obtained from external means, such as the gains calculated from the DCRaw software.

Although it is not clear that there is a relation between image overall quality and differences between the saliency maps of the original and tone-mapped image, it is possible that some correlation between the tone-mapped image saliency map and its quality exists. Different metrics other than the MSE, such as the Area Under Curve (AUC), might provide a better insight of how saliency maps relate to image overall quality and, consequently, to the tone-mapping operator performance. Subjective and objective evaluation of the images resulting from every tone-mapping operator also suggests that in some images it is harder to make details visible in dark and bright regions simultaneously (for example, the Color Chart, Sunset and Courtyard images), while in others it is easier (for example, the Beach, Palace and Flowers images). It would be interesting to investigate what kind of image features determine the difficulty of generating good quality results, regarding detail level in both bright and dark regions.

Another possibility is to further explore different circuit configurations that implement the same tone-mapping operation, by taking advantage of the color information introduced by the Bayer CFA. Some configurations may lead to further reductions in the complexity and/or size of the focal-plane circuit. The improved circuits might achieve shorter execution times, and yet yield good-quality tone-mapped images. Like the FPTMO modified circuit considered in this work, new configurations may place a lighter computational burden on the color processing tasks that

occur after the tone mapping. Depending on the amount of simplification, such color processing tasks might be amenable to focal-plane implementation. Implementing color processing tasks in the focal plane would lead to further reduction in the processing time required for generating color tone-mapped images.

Bibliography

[1] WANDELL, B. "Foundations of Vision". Sinauer Associates, 1995.

[2] REINHARD, E., STARK, M., SHIRLEY, P., et al. "Photographic Tone Reproduction for Digital Images", *ACM Trans. Graph.*, v. 21, n. 3, pp. 267–276, jul. 2002. ISSN: 0730-0301. doi: 10.1145/566654.566575. Disponível em: <http://doi.acm.org/10.1145/566654.566575>.

[3] TUMBLIN, J., RUSHMEIER, H. "Tone reproduction for realistic images", *IEEE Computer Graphics and Applications*, v. 13, n. 6, pp. 42–48, Nov 1993. ISSN: 0272-1716. doi: 10.1109/38.252554.

[4] LARSON, G. W., RUSHMEIER, H., PIATKO, C. "A visibility matching tone reproduction operator for high dynamic range scenes", *IEEE Transactions on Visualization and Computer Graphics*, v. 3, n. 4, pp. 291–306, Oct 1997. ISSN: 1077-2626. doi: 10.1109/2945.646233.

[5] BANTERLE, F., LEDDA, P., DEBATTISTA, K., et al. "Inverse Tone Mapping". In: *Proceedings of the 4th International Conference on Computer Graphics and Interactive Techniques in Australasia and Southeast Asia*, GRAPHITE '06, pp. 349–356, New York, NY, USA, 2006. ACM. ISBN: 1-59593-564-9. doi: 10.1145/1174429.1174489. Disponível em: <http://doi.acm.org/10.1145/1174429.1174489>.

[6] HUO, Y., YANG, F., BROST, V. "A LDR image expansion method for displaying on HDR screen". In: *2013 International Conference on Computational Problem-Solving (ICCP)*, pp. 234–237, Oct 2013. doi: 10.1109/ICCPS.2013.6893570.

[7] KINOSHITA, Y., SIOTA, S., KIYA, H. "An inverse tone mapping operator based on Reinhard's global operator". In: *2016 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS)*, pp. 1–6, Oct 2016. doi: 10.1109/ISPACS.2016.7824712.

[8] VARGAS-SIERRA, S., LIÑÁN-CEMBRANO, G., RODRÍGUEZ-VÁZQUEZ, A. "A 151 dB High Dynamic Range CMOS Image Sensor Chip Architecture With Tone Mapping Compression Embedded In-Pixel", *IEEE Sensors Journal*, v. 15, n. 1, pp. 180–195, January 2015.

[9] MUGHAL, W., CHOUBEY, B. "Fixed pattern noise correction for wide dynamic range CMOS image sensor with Reinhard tone mapping operator". In: *2015 Nordic Circuits and Systems Conference (NORCAS): NORCHIP & International Symposium on System-on-Chip (SoC)*, 2015.

[10] MARTEL, J. N. P., MÜLLER, L. K., CAREY, S. J., et al. "Parallel HDR tone mapping and auto-focus on a cellular processor array vision chip". In: *2016 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2016.

[11] FERNÁNDEZ-BERNI, J., OLIVEIRA, F. D. V. R., CARMONA-GALÁN, R., et al. "Image Sensing Scheme Enabling Fully-Programmable Light Adaptation and Tone Mapping With a Single Exposure", *IEEE Sensors Journal*, v. 16, n. 13, pp. 5121–5122, July 2016. ISSN: 1530-437X. doi: 10.1109/JSEN.2016.2559158.

[12] OHTA, J. "Smart CMOS Image Sensors and Applications". CRC Press, 2007.

[13] BOYLE, W. S., SMITH, G. E. "Charge Coupled Semiconductor Devices", *Bell System Technical Journal*, v. 49, n. 4, pp. 587–593, 1970. ISSN: 1538-7305. doi: 10.1002/j.1538-7305.1970.tb01790.x. Disponível em: <http://dx.doi.org/10.1002/j.1538-7305.1970.tb01790.x>.

[14] HOLST, G. C., LOMHEIM, T. S. "CMOS/CCD Sensors and Camera Systems". SPIE Press Books, 2007.

[15] NAKAMURA, J. "Image Sensors and Signal Processing for Digital Still Cameras". CRC Press, 2006.

[16] REINHARD, E., WARD, G., PATTANAIK, S., et al. *High Dynamic Range Imaging: Acquisition, Display, and Image-Based Lighting (The Morgan Kaufmann Series in Computer Graphics)*. San Francisco, CA, USA, Morgan Kaufmann Publishers Inc., 2005. ISBN: 0125852630.

[17] STEVENS, J. C., STEVENS, S. S. "Brightness Function: Effects of Adaptation*", *J. Opt. Soc. Am.*, v. 53, n. 3, pp. 375–385, Mar 1963. doi: 10.1364/JOSA.53.000375. Disponível em: <http://www.osapublishing.org/abstract.cfm?URI=josa-53-3-375>.

[18] MICHELSON, A. A. "Studies in Optics". University Press, 1927.

[19] CAMPBELL, F. W., ROBSON, J. G. "Application of fourier analysis to the visibility of gratings", *The Journal of Physiology*, v. 197, n. 3, pp. 551–566, 1968. ISSN: 1469-7793. doi: 10.1113/jphysiol.1968.sp008574. Disponível em: <http://dx.doi.org/10.1113/jphysiol.1968.sp008574>.

[20] PELI, E. "Contrast in complex images", *J. Opt. Soc. Am. A*, v. 7, n. 10, pp. 2032–2040, Oct 1990. doi: 10.1364/JOSAA.7.002032. Disponível em: <http://josaa.osa.org/abstract.cfm?URI=josaa-7-10-2032>.

[21] MATKOVIĆ, K., NEUMANN, L., NEUMANN, A., et al. "Global Contrast Factor - a New Approach to Image Contrast". In: *Proceedings of the First Eurographics Conference on Computational Aesthetics in Graphics, Visualization and Imaging*, Computational Aesthetics'05, pp. 159–167, Aire-la-Ville, Switzerland, Switzerland, 2005. Eurographics Association. ISBN: 3-905673-27-4. doi: 10.2312/COMPAESTH/COMPAESTH05/159-167. Disponível em: <http://dx.doi.org/10.2312/COMPAESTH/COMPAESTH05/159-167>.

[22] ČADÍK, M., WIMMER, M., NEUMANN, L., et al. "Evaluation of HDR tone mapping methods using essential perceptual attributes", *Computers & Graphics*, v. 32, n. 3, pp. 330 – 349, 2008. ISSN: 0097-8493. doi: https://doi.org/10.1016/j.cag.2008.04.003. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0097849308000460>.

[23] SCHLICK, C. "Quantization Techniques for Visualization of High Dynamic Range Pictures". pp. 7–20. Springer-Verlag, 1994.

[24] WARD, G. "Graphics Gems IV". Academic Press Professional, Inc., cap. A Contrast-based Scalefactor for Luminance Display, pp. 415–421, San Diego, CA, USA, 1994. ISBN: 0-12-336155-9. Disponível em: <http://dl.acm.org/citation.cfm?id=180895.180934>.

[25] FERWERDA, J. A., PATTANAIK, S. N., SHIRLEY, P., et al. "A Model of Visual Adaptation for Realistic Image Synthesis". In: *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '96, pp. 249–258, New York, NY, USA, 1996. ACM. ISBN: 0-89791-746-4. doi: 10.1145/237170.237262. Disponível em: <http://doi.acm.org/10.1145/237170.237262>.

[26] DRAGO, F., MYSZKOWSKI, K., ANNEN, T., et al. "Adaptive Logarithmic Mapping For Displaying High Contrast Scenes", *Computer Graphics Fo-*

rum, v. 22, n. 3, pp. 419–426, 2003. ISSN: 1467-8659. doi: 10.1111/1467-8659.00689. Disponível em: <http://dx.doi.org/10.1111/1467-8659.00689>.

[27] CHIU, K., HERF, M., SHIRLEY, P., et al. "Spatially nonuniform scaling functions for high contrast images". In: *In Proceedings of Graphics Interface '93*, pp. 245–253, 1993.

[28] LAND, E. H. "The Retinex Theory of Color Vision", *Scientific American*, v. 237, n. 6, pp. 108–129, 1977. ISSN: 00368733, 19467087. Disponível em: <http://www.jstor.org/stable/24953876>.

[29] ASHIKHMIN, M. "A Tone Mapping Algorithm for High Contrast Images". In: *Proceedings of the 13th Eurographics Workshop on Rendering*, EGRW '02, pp. 145–156, Aire-la-Ville, Switzerland, Switzerland, 2002. Eurographics Association. ISBN: 1-58113-534-3. Disponível em: <http://dl.acm.org/citation.cfm?id=581896.581916>.

[30] "Foveon X3 sensor technology". http://www.foveon.com/article.php?a=74. Last accessed 23 November 2017.

[31] BAYER, B. "Color imaging array". jul. 20 1976. Disponível em: <http://www.google.com/patents/US3971065>. US Patent 3,971,065.

[32] GUNTURK, B. K., GLOTZBACH, J., ALTUNBASAK, Y., et al. "Demosaicking: Color Filter Array Interpolation", v. 22, n. 1, January 2005.

[33] MALVAR, H. S., WEI HE, L., CUTLER, R. "High-quality linear interpolation for demosaicing of Bayer-patterned color images". In: *2004 IEEE International Conference on Acoustics, Speech, and Signal Processing*, v. 3, pp. iii–485–8 vol.3, May 2004. doi: 10.1109/ICASSP.2004.1326587.

[34] COFFIN, D. "DCRaw software". https://www.cybercom.net/~dcoffin/dcraw/. Last accessed 01 December 2017.

[35] LUKAC, R. *Single-Sensor Imaging: Methods and Applications for Digital Cameras*. 1 ed. Boca Raton, FL, USA, CRC Press, Inc., 2008. ISBN: 142005452X, 9781420054521.

[36] YAN HUO, J., LIN CHANG, Y., WANG, J., et al. "Robust Automatic White Balance Algorithm using Gray Color Points in Images", v. 52, n. 2, May 2005.

[37] NUNES, G., OLIVEIRA, F., GOMES, J. G., et al. "Color Tone-Mapping Circuit for a Focal-Plane Implementation". In: *2018 IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2018.

[38] INTEL. "Intel® 64 and IA-32 Architectures Optimization Reference Manual". `https://software.intel.com/sites/default/files/managed/9e/bc/64-ia-32-architectures-optimization-manual.pdf`.

[39] OLIVEIRA, F. D. V. R., GOMES, J. G. R. C., FERNÁNDEZ-BERNI, J., et al. "Gaussian Pyramid: Comparative Analysis of Hardware Architectures", v. 64, n. 9, pp. 2308–2321, September 2017.

[40] "ETH Zürich, EMPA Media Technology, HDR Database". `http://empamedia.ethz.ch/hdrdatabase/`. Last accessed 19 July 2017.

[41] HAEBERLI, P. E. "Matrix Operations for Image Processing". `http://graficaobscura.com/matrix/index.html`, November 1993. Last accessed 09 November 2017.

[42] HERWIG, J., SOBCZYK, M., PAULI, J. "Tone Mapping for Single-shot HDR Imaging". In: *2014 IEEE International Conference on Computer Vision Theory and Applications (VISAPP)*, 2014.

[43] YEGANEH, H., WANG, Z. "Objective Quality Assessment of Tone-Mapped Images", *IEEE Transactions on Image Processing*, v. 22, n. 2, pp. 657–667, February 2013.

[44] WANG, Z., BOVIK, A. C., SHEIKH, H. R., et al. "Image Quality Assessment: From Error Visibility to Structural Similarity", *IEEE Transactions on Image Processing*, v. 13, n. 4, pp. 600–612, April 2004.

[45] WANG, Z., BOVIK, A. C. "Reduced- and No-Reference Image Quality Assessment", *IEEE Signal Processing Magazine*, v. 28, n. 6, pp. 29–40, November 2011.

[46] MANTIUK, R., KIM, K. J., REMPEL, A. G., et al. "HDR-VDP-2: A Calibrated Visual Metric for Visibility and Quality Predictions in All Luminance Conditions", *ACM Trans. Graph.*, v. 30, n. 4, pp. 40:1–40:14, jul. 2011. ISSN: 0730-0301. doi: 10.1145/2010324.1964935. Disponível em: <`http://doi.acm.org/10.1145/2010324.1964935`>.

[47] NARWARIA, M., DA SILVA, M. P., CALLET, P. L. "High Dynamic Rage Visual Quality of Experience Measurement: Challenges and Perspectives". Springer, Cham, 2015. doi: https://doi.org/10.1007/978-3-319-10368-6_5.

[48] JUDD, T., DURAND, F., TORRALBA, A. "A Benchmark of Computational Models of Saliency to Predict Human Fixations". In: *MIT Technical Report*, 2012. Disponível em: <http://hdl.handle.net/1721.1/68590>.

[49] BYLINSKII, Z., JUDD, T., BORJI, A., et al. "MIT Saliency Benchmark". http://saliency.mit.edu/. Last accessed 09 January 2018.

[50] GOFERMAN, S., ZELNIK-MANOR, L., TAL, A. "Context-Aware Saliency Detection", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 34, n. 10, pp. 1915–1926, Oct 2012. ISSN: 0162-8828. doi: 10.1109/TPAMI.2011.272.

[51] SUMMER, R. "Processing RAW Images in MATLAB". https://rcsumner. net/raw_guide/RAWguide.pdf, May 2014. Last accessed 09 January 2018.

Appendix A

RAW Image Treatment

In this appendix, it is described how the raw images are read and pre-preprocessed, before they are made available for the tone-mapping operators. The pre-processing steps used in this work follow the ones defined in the guide "Processing RAW Images in MATLAB", written by Rob Summer [51]. The raw image format is not standardized. Each camera manufacturer has its own proprietary raw format (.CR2 for Canon, .NEF for Nikon, and so on). When using the MATLAB software, attempting to read raw HDR images in these formats into a matrix variable does not yield pixel values in the original dynamic range. Instead, the built-in MATLAB functions assign the raw image to a matrix variable, whose pixel values are quantized using eight bits, thereby varying from 0 to 255. An unknown tone-mapping algorithm is executed on the raw image before it is made available to the user. This is undesirable in this work, since the implemented tone-mapping operators are compared to each other based on their resulting images. The differences observed between the images are assumed to be caused only by the different implemented tone-mapping functions. To make sure that no other processing is affecting the performance of the tone-mapping operators, the input raw image pixel values are expected to be as close as possible to the original sensor values.

To read the data in its original dynamic range, the raw image is first converted to a 16-bit TIFF file (the original dynamic range is assumed to not exceed the 16-bit value range). The conversion is done through an auxiliary open-source software called *DCRaw*, created by Dave Coffin [34]. This software runs in command-line and is suitable for raw image processing. It reads many different raw image formats and contains information from many camera models from different manufacturers. To perform the conversion, the program must run with the options "-4 -D -T", where "-4" is to convert the image using sixteen bits, "-D" is to not apply any white-balance or demosaicing to the input image and "-T" is to convert the input image to the TIFF image format. An example of this operation is shown in Figure A.1. After the image is converted, it can be read in MATLAB into a matrix variable that preserves

```
C:\>dcraw-9.27-ms-64-bit.exe -4 -D -T img01.CR2
```

Figure A.1: DCRaw command for converting an image from a proprietary raw file format to the TIFF file format.

```
C:\>dcraw-9.27-ms-64-bit.exe -v -w -T img01.CR2
Loading Canon EOS-1D Mark IV image from img01.CR2 ...
Scaling with darkness 2048, saturation 15280, and
multipliers 2.115234 1.000000 1.533203 1.000000
AHD interpolation...
Converting to sRGB colorspace...
Writing data to img01.tiff ...
```

Figure A.2: DCRaw command for finding the maximum and minimum raw image pixel values, as well as the white-balance gains calculated by the camera for the corresponding image. Inside the red rectangle are the minimum (darkness) and maximum (saturation) raw pixel values.

the original dynamic range.

The raw pixel values may have an offset and arbitrary scaling, thus occupying a value range between b and $b + s \cdot (2^{n_{bits\_cam}} - 1)$ (where s denotes the scale factor and b denotes the offset value), instead of spaning the value range between 0 and $2^{n_{bits\_cam}} - 1$ (where n_{bits_cam} is the number of bits used by the camera to convert from analog to digital pixel values). In order to avoid problems caused by raw pixel values being in different ranges, the pixel values are normalized to the same range, which is the [0,1] value range. The maximum and minimum raw pixel values can be found by executing the DCRaw software with the command "-v -w -T", where "-v" enables the verbose mode, in which the program outputs information about the image processing, such as image metadata and white-balance gains used, "-w" is to use the white-balance coefficients calculated by the camera for the corresponding image (if they are available) and "-T" is to output the processed image in the TIFF file format. Figure A.2 shows the software execution with this command and the corresponding minimum and maximum raw image pixel values given by the software. Because of sensor noise, the maximum and minimum pixel values might be, respectively, higher and lower than the values returned by the software [51]. As a consequence, after the normalization, any pixel values that lie outside the [0,1] range must be clamped to 0 (if the normalized pixel values are negative) and to 1 (if the normalized pixel values are above unity).

The raw image that has been read is in the Bayer pattern, which means that each pixel represents a different color channel (red, green or blue). For all tone-mapping operators that are applied directly to raw images in the Bayer pattern, such as the FPTMO implementations, this is the final raw image pre-processing stage. The raw image is then made available for these operators. For the digital

```
C:\>dcraw-9.27-ms-64-bit.exe -v -w -T img01.CR2
Loading Canon EOS-1D Mark IV image from img01.CR2 ...
Scaling with darkness 2048, saturation 15280, and
multipliers 2.115234 1.000000 1.533203 1.000000
AHD interpolation...
Converting to sRGB colorspace...
Writing data to img01.tiff ...
```

Figure A.3: DCRaw command for finding the maximum and minimum pixel values of the raw image, as well as the white-balance gains calculated by the camera for the corresponding raw image. Inside the red rectangle are the white-balance gains for the red, green, blue and green pixels, respectively (the second and fourth green white-balance gains correspond to the green pixels next to the red and blue pixels, respectively).

tone-mapping operators, further processing is required, because their input are color RGB images. Each pixel in these images is represented by three values, instead of a single value. Each value corresponds to one particular color channel (red, green or blue). To obtain a color RGB image, the Bayer pattern raw image pixel values are multiplied by the corresponding white-balance gains. The gains are obtained through the DCRaw software, using same command issued previously. The gain values are inside the red rectangle in the image shown in Figure A.3. The first and third numbers correspond to the gain of the red and blue pixels, respectively. The second and fourth numbers correspond to the gain of the green pixels next to the red and blue pixels, respectively.

To avoid color distortions caused by multiplying pixels of the Bayer pattern by incorrect white-balance gains, it is necessary to know which Bayer pattern arrangement the camera sensors use. This can be done by executing the DCRaw software with the command "-i -v". This command identifies the image file and outputs metadata information about the image (without processing the image). Figure A.4 shows the output console messages that are shown when this command is used. After multiplying the pixels by the corresponding white-balance gains, values outside the [0,1] range are clamped to 0 or 1. A demosaicing algorithm is then applied to the resulting raw image, thus yielding a color RGB raw image.

One last step is required, in order to assure that the raw image colors are correctly shown on a display device. The camera sensors have their own responses to different wavelengths of the incoming light (which depend on the response of the color filter used in each sensor and on the sensor quantum efficiency curves). The camera sensors are divided into three types. Each type is more sensitive to a different part of the visible light spectrum. The raw pixel values are then determined according to the incoming light spectral distribution and to the spectral sensitivity curve of the given sensor. When demosaicing is performed, each raw pixel of the resulting

Figure A.4: DCRaw command for showing some metadata of the image file. Inside the red rectangle is the Bayer pattern arrangement used for the sensors of the camera.

image contains three values that represent the sensor responses to its incident light, considering each different spectral sensitivity curve. These curves can be interpreted as the basis of a color space associated with the camera. Each triple-valued raw pixel can be interpreted as a point in this color space (with each point corresponding to a different color).

The LEDs (light-emitting diodes) used to reproduce colors on the display devices usually do not have the same spectral responses as the sensors of a camera. The basis of the display device color space, which is formed by the LED spectral responses, is different from the one of the camera color space. As a consequence, if the resulting images from the demosaicing operation are displayed without correcting their pixel values according to the LED spectral responses of the display device, color distortions may be observed, since the same point in different color spaces may not correspond to the same color. Conventional display devices have a standardized color space, which was cooperatively created by Microsoft and HP. The standard display device color space is called *sRGB*.

The pixel values must then be converted from the camera color space to the standard sRGB color space. In the DCRaw software script, there is a function called "adobe_coeff". This function contains a list of matrix transformations from the XYZ color space, which is standardized by the CIE (Comission Internationale de l'Éclairage), to color spaces of different camera models. The matrix transformations are denoted as $M_{XYZ-to-Cam}$. All matrix values are multiplied by a factor of $s = 10^4$ in the function "adobe_coeff". Therefore, before using such transformations, the matrix values must be normalized by 10^4. To obtain a direct transformation matrix from the sRGB color space to the camera color space ($M_{sRGB-to-Cam}$), the matrix representing the transformation from sRGB to XYZ color space ($M_{sRGB-to-XYZ}$) is

multiplied by the corresponding XYZ-to-Cam matrix $M_{XYZ-to-Cam}$, thus leading to $M_{sRGB-to-Cam} = M_{XYZ-to-Cam} \cdot M_{sRGB-to-XYZ}$. The sRGB-to-XYZ transformation is standardized and uses the CIE D_{65} illuminant as white light reference. This transformation is given below:

$$
\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.412 & 0.358 & 0.180 \\ 0.213 & 0.715 & 0.072 \\ 0.019 & 0.119 & 0.950 \end{bmatrix} \cdot \begin{bmatrix} R \\ G \\ B \end{bmatrix}. \tag{A.1}
$$

White pixels in both camera and sRGB color spaces are represented as the triple-element vector $[1\ 1\ 1]^T$. The color space transformation matrix must then assure that a white pixel in the display device color space corresponds to a white pixel in the camera color space. Equivalently, the triple-element vector $[1\ 1\ 1]^T$ in one color space must be mapped to the same triple-element vector $[1\ 1\ 1]^T$ in the other color space. This is achieved by making the sum of elements in the same row in the sRGB-to-Cam matrix equal to one. Each matrix element must then be normalized by the sum of elements in the corresponding matrix row. The calculated transformation matrix converts from the sRGB to the camera color space. The transformation matrix $M_{Cam-to-sRGB}$ that does the reverse conversion (i.e. from the camera to the sRGB color space) is obtained by inverting the sRGB-to-Cam matrix: $M_{Cam-to-sRGB} = M_{sRGB-to-Cam}^{-1}$. The color raw image yielded after the demosaicing operation is multiplied by the color space transformation matrix $M_{Cam-to-sRGB}$, thereby yielding the final color raw image that serves as input to the digital tone-mapping operators. This completes the raw image pre-processing.

Appendix B

FPTMO with Fixed DCRaw WB Gains For Every Image

In this appendix, the images that result from the FPTMO second and third versions (according to the definitions given in Chapter 4) are shown. These images were generated by fixing, for every image, the DCRaw red and blue white-balance gains. For the FPTMO second version, these gains are used to calculate the final white-balance gains for each pixel in the image. For the FPTMO third version, these gains already correspond to the final white-balance gains. The DCRaw gains were fixed at 2.1610 for the red channel and at 1.5634 for the blue channel. Such values were chosen based on the red and blue white-balance gains calculated by the DCRaw software for each of the ten HDR images used in this work.

(a) Parasol.

(b) Fire Extinguisher.

(c) Flowers.

(d) Sunset.

(e) Courtyard.

(f) Color Chart 1.

(g) Beach.

(h) Palace.
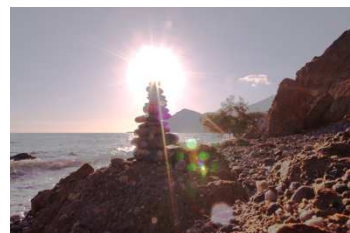
(i) Stone Tower.

(j) Color Chart 2.

Figure B.1: Images resulting from the FPTMO second digital version, in which each pixel integration time is calculated using the own pixel value. The DCRaw white-balance gains were fixed at 2.1610 for the red channel and at 1.5634 for the blue channel.

(a) Parasol.

(b) Fire Extinguisher.

(c) Flowers.
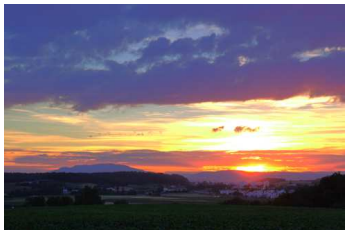
(d) Sunset.

(e) Courtyard.
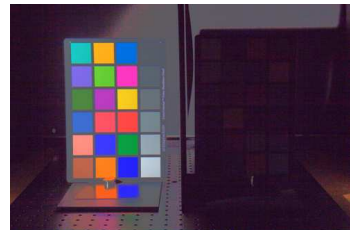
(f) Color Chart 1.

(g) Beach.

(h) Palace.

(i) Stone Tower.

(j) Color Chart 2.

Figure B.2: Images resulting from the FPTMO third digital version, in which the integration time of blue and red pixels is calculated using the value of the neighboring green pixel located above or below them. The DCRaw white-balance gains were fixed at 2.1610 for the red channel and at 1.5634 for the blue channel.

Appendix C

Tone-Mapping Operators Resulting Images

In this appendix, the best results of each tone-mapping operator implemented in this work are presented (the FPTMO implementations follow the definitions given in Chapter 4). The best results were obtained based on the criteria defined in Chapter 4.

(a) Raw.   (b) Logarithm.   (c) Ward contrast-based.

(d) Ferwerda.   (e) Tumblin-Rushmeier.   (f) Exponential.

(g) Drago.   (h) Reinhard Global.   (i) Ward Histogram Adjustment.

(j) Schlick.   (k) Chiu.   (l) Rahman.

(m) Reinhard Local.   (n) Ashikhmin.   (o) FPTMO First Version.

(p) FPTMO Second Version.   (q) FPTMO Third Version.

Figure C.1: Different tone-mapping operators applied to the Parasol image.

(a) Raw.

(b) Logarithm.

(c) Ward contrast-based.

(d) Ferwerda.

(e) Tumblin-Rushmeier.

(f) Exponential.

(g) Drago.

(h) Reinhard Global.

(i) Ward Histogram Adjustment.

(j) Schlick.

(k) Chiu.

(l) Rahman.

(m) Reinhard Local.

(n) Ashikhmin.

(o) FPTMO First Version.

(p) FPTMO Second Version.

(q) FPTMO Third Version.

Figure C.2: Different tone-mapping operators applied to the Fire Extinguisher image.

(a) Raw.   (b) Logarithm.   (c) Ward contrast-based.

(d) Ferwerda.   (e) Tumblin-Rushmeier.   (f) Exponential.

(g) Drago.   (h) Reinhard Global.   (i) Ward Histogram Adjustment.

(j) Schlick.   (k) Chiu.   (l) Rahman.

(m) Reinhard Local.   (n) Ashikhmin.   (o) FPTMO First Version.

(p) FPTMO Second Version.   (q) FPTMO Third Version.

Figure C.3: Different tone-mapping operators applied to the Flowers image.

(a) Raw.

(b) Logarithm.

(c) Ward contrast-based.

(d) Ferwerda.

(e) Tumblin-Rushmeier.

(f) Exponential.

(g) Drago.

(h) Reinhard Global.

(i) Ward Histogram Adjustment.

(j) Schlick.

(k) Chiu.

(l) Rahman.

(m) Reinhard Local.

(n) Ashikhmin.

(o) FPTMO First Version.

(p) FPTMO Second Version.

(q) FPTMO Third Version.

Figure C.4: Different tone-mapping operators applied to the Sunset image.

(a) Raw.

(b) Logarithm.

(c) Ward contrast-based.

(d) Ferwerda.

(e) Tumblin-Rushmeier.

(f) Exponential.

(g) Drago.

(h) Reinhard Global.

(i) Ward Histogram Adjustment.

(j) Schlick.

(k) Chiu.

(l) Rahman.

(m) Reinhard Local.

(n) Ashikhmin.

(o) FPTMO First Version.

(p) FPTMO Second Version.

(q) FPTMO Third Version.

Figure C.5: Different tone-mapping operators applied to the Courtyard image.

154

(a) Raw.

(b) Logarithm.

(c) Ward contrast-based.

(d) Ferwerda.

(e) Tumblin-Rushmeier.

(f) Exponential.

(g) Drago.

(h) Reinhard Global.

(i) Ward Histogram Adjustment.

(j) Schlick.

(k) Chiu.

(l) Rahman.

(m) Reinhard Local.

(n) Ashikhmin.

(o) FPTMO First Version.

(p) FPTMO Second Version.

(q) FPTMO Third Version.
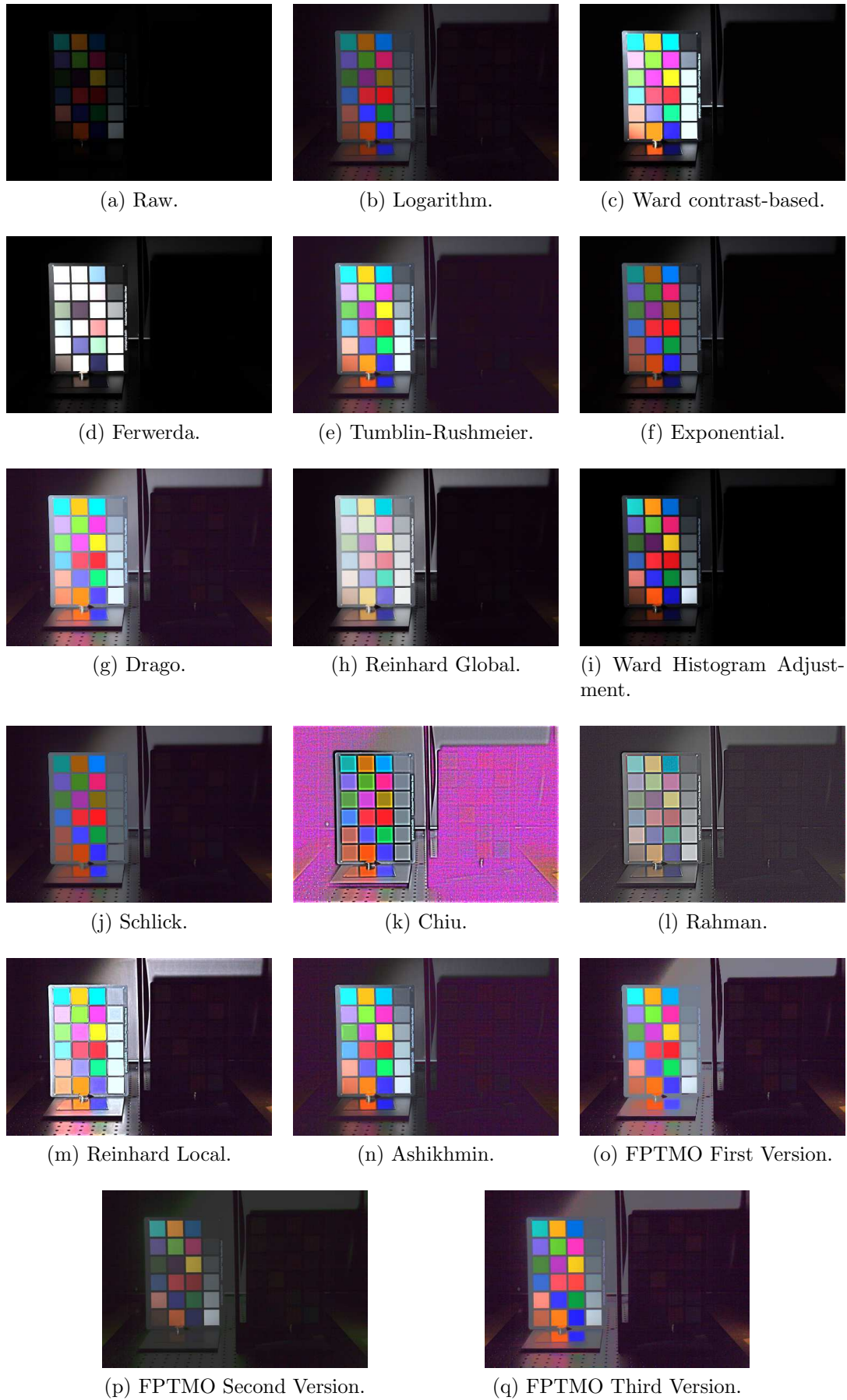
Figure C.6: Different tone-mapping operators applied to the Color Chart 1 image.

(a) Raw.

(b) Logarithm.

(c) Ward contrast-based.

(d) Ferwerda.

(e) Tumblin-Rushmeier.

(f) Exponential.

(g) Drago.

(h) Reinhard Global.

(i) Ward Histogram Adjustment.

(j) Schlick.

(k) Chiu.

(l) Rahman.

(m) Reinhard Local.

(n) Ashikhmin.

(o) FPTMO First Version.

(p) FPTMO Second Version.

(q) FPTMO Third Version.
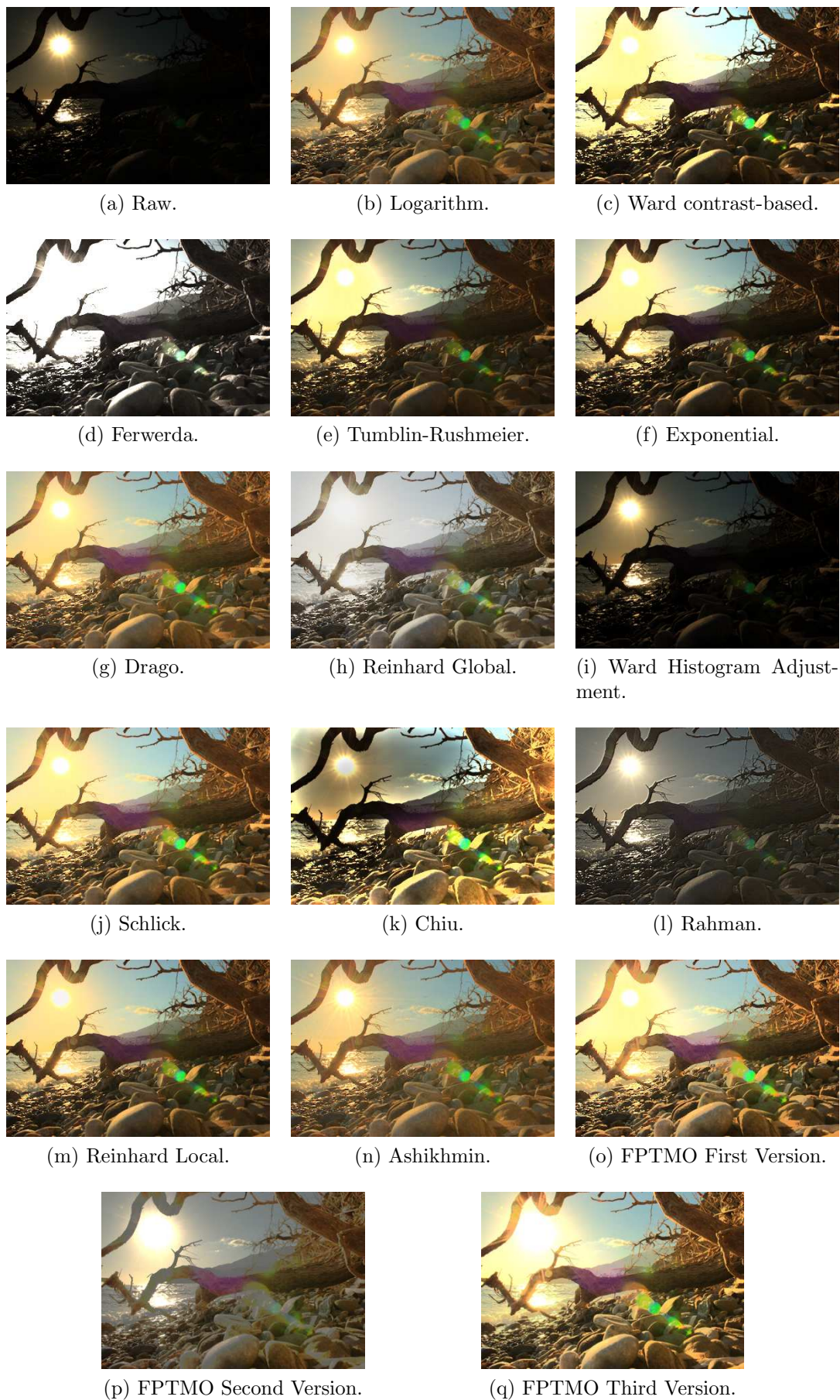
Figure C.7: Different tone-mapping operators applied to the Beach image.

(a) Raw.

(b) Logarithm.

(c) Ward contrast-based.

(d) Ferwerda.

(e) Tumblin-Rushmeier.

(f) Exponential.

(g) Drago.

(h) Reinhard Global.

(i) Ward Histogram Adjustment.

(j) Schlick.

(k) Chiu.

(l) Rahman.

(m) Reinhard Local.

(n) Ashikhmin.

(o) FPTMO First Version.

(p) FPTMO Second Version.

(q) FPTMO Third Version.

Figure C.8: Different tone-mapping operators applied to the Palace image.

(a) Raw.

(b) Logarithm.

(c) Ward contrast-based.

(d) Ferwerda.

(e) Tumblin-Rushmeier.

(f) Exponential.

(g) Drago.

(h) Reinhard Global.

(i) Ward Histogram Adjustment.

(j) Schlick.

(k) Chiu.

(l) Rahman.

(m) Reinhard Local.

(n) Ashikhmin.

(o) FPTMO First Version.

(p) FPTMO Second Version.

(q) FPTMO Third Version.
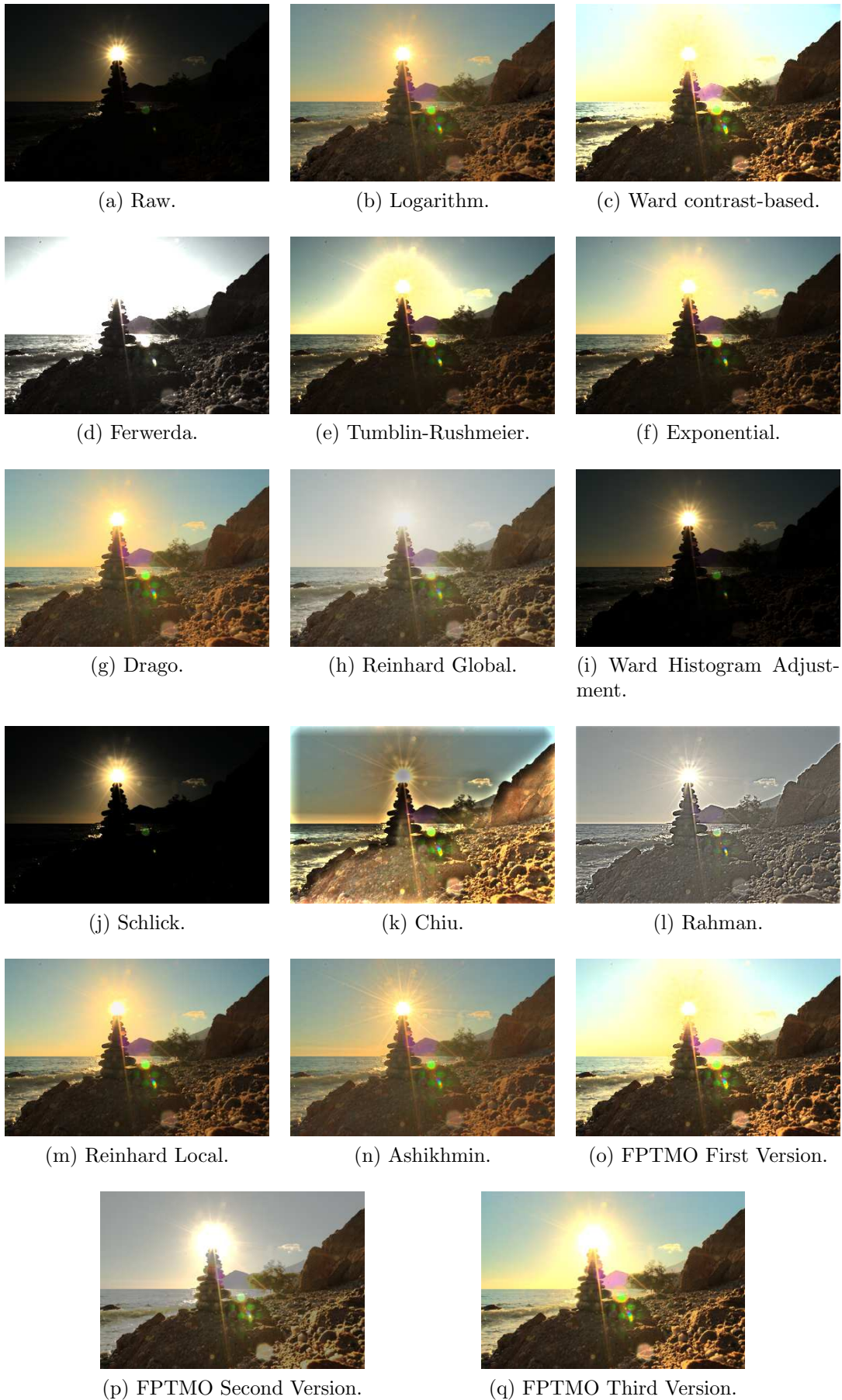
Figure C.9: Different tone-mapping operators applied to the Stone Tower image.

158
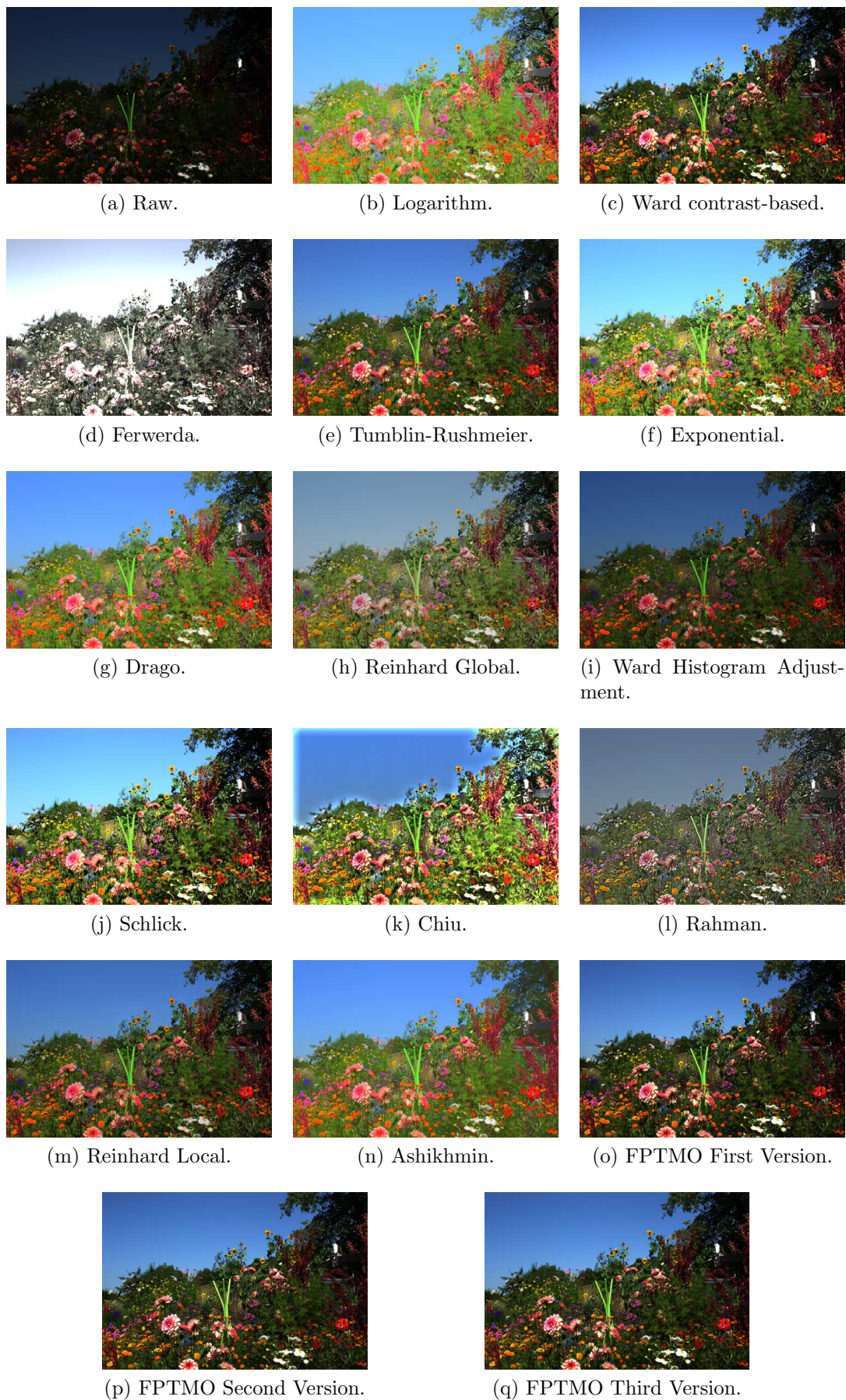
(a) Raw.

(b) Logarithm.

(c) Ward contrast-based.

(d) Ferwerda.

(e) Tumblin-Rushmeier.
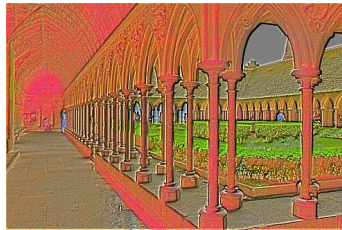
(f) Exponential.

(g) Drago.

(h) Reinhard Global.

(i) Ward Histogram Adjustment.

(j) Schlick.

(k) Chiu.

(l) Rahman.

(m) Reinhard Local.

(n) Ashikhmin.

(o) FPTMO First Version.

(p) FPTMO Second Version.

(q) FPTMO Third Version.
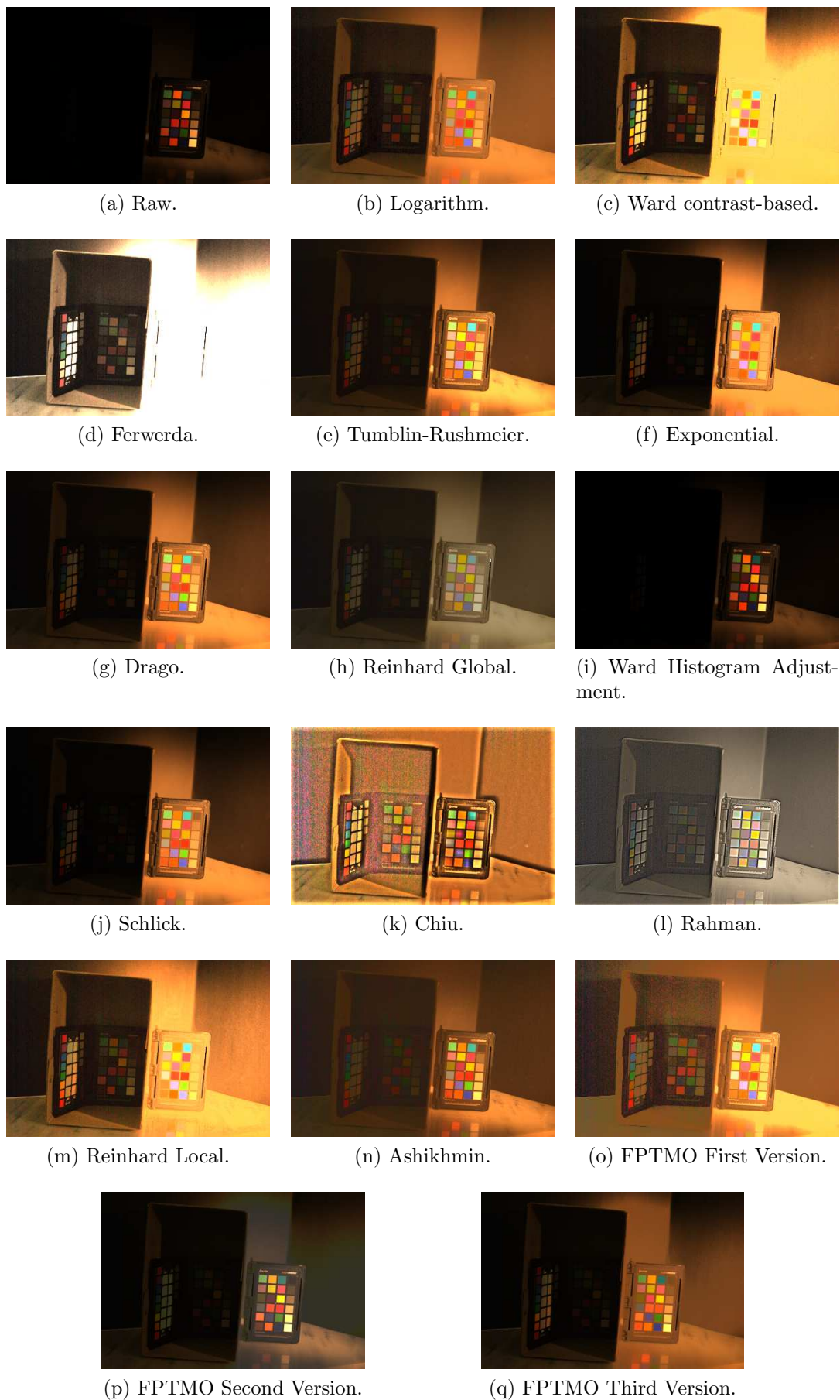
Figure C.10: Different tone-mapping operators applied to the Color Chart 2 image.

Appendix D

Saliency Maps of the Raw and Tone-Mapped Images

In this appendix, the saliency maps of the raw and all tone-mapped versions of every HDR image considered in this work are presented. The saliency maps were obtained using the "Context-Aware Saliency Detection" algorithm [50]. The FPTMO implementations follow the definitions given in Chapter 4.

(a) Raw.

(b) Logarithm.

(c) Ward contrast-based.

(d) Ferwerda.

(e) Tumblin-Rushmeier.

(f) Exponential.

(g) Drago.

(h) Reinhard Global.

(i) Ward Histogram Adjustment.

(j) Schlick.

(k) Chiu.

(l) Rahman.

(m) Reinhard Local.

(n) Ashikhmin.

(o) FPTMO First Version.

(p) FPTMO Second Version.

(q) FPTMO Third Version.
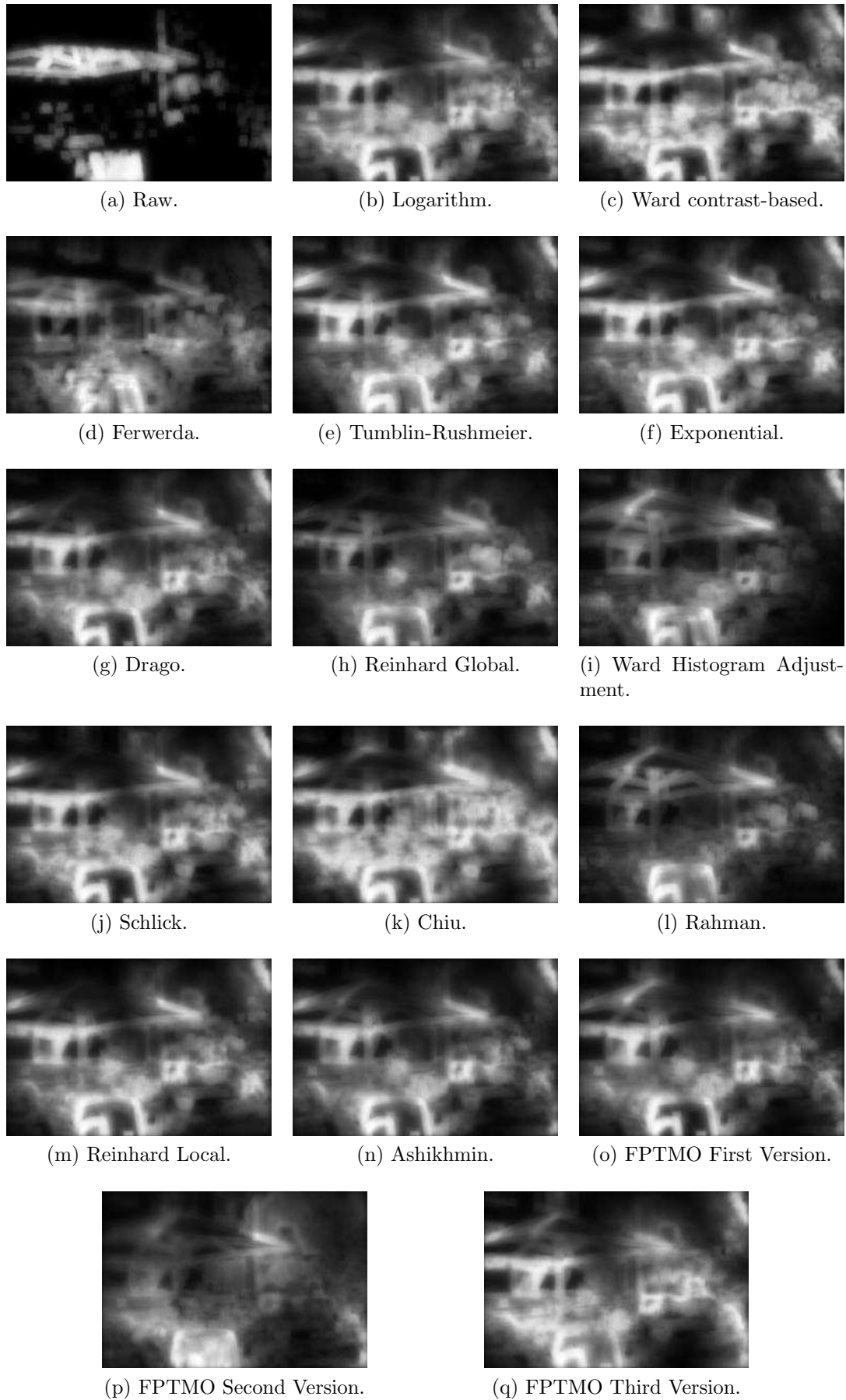
Figure D.1: Saliency maps of the raw and tone-mapped versions of the Parasol image.
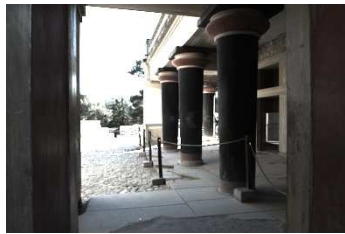
(a) Raw.

(b) Logarithm.

(c) Ward contrast-based.

(d) Ferwerda.

(e) Tumblin-Rushmeier.

(f) Exponential.

(g) Drago.

(h) Reinhard Global.

(i) Ward Histogram Adjustment.

(j) Schlick.

(k) Chiu.

(l) Rahman.

(m) Reinhard Local.

(n) Ashikhmin.

(o) FPTMO First Version.

(p) FPTMO Second Version.

(q) FPTMO Third Version.
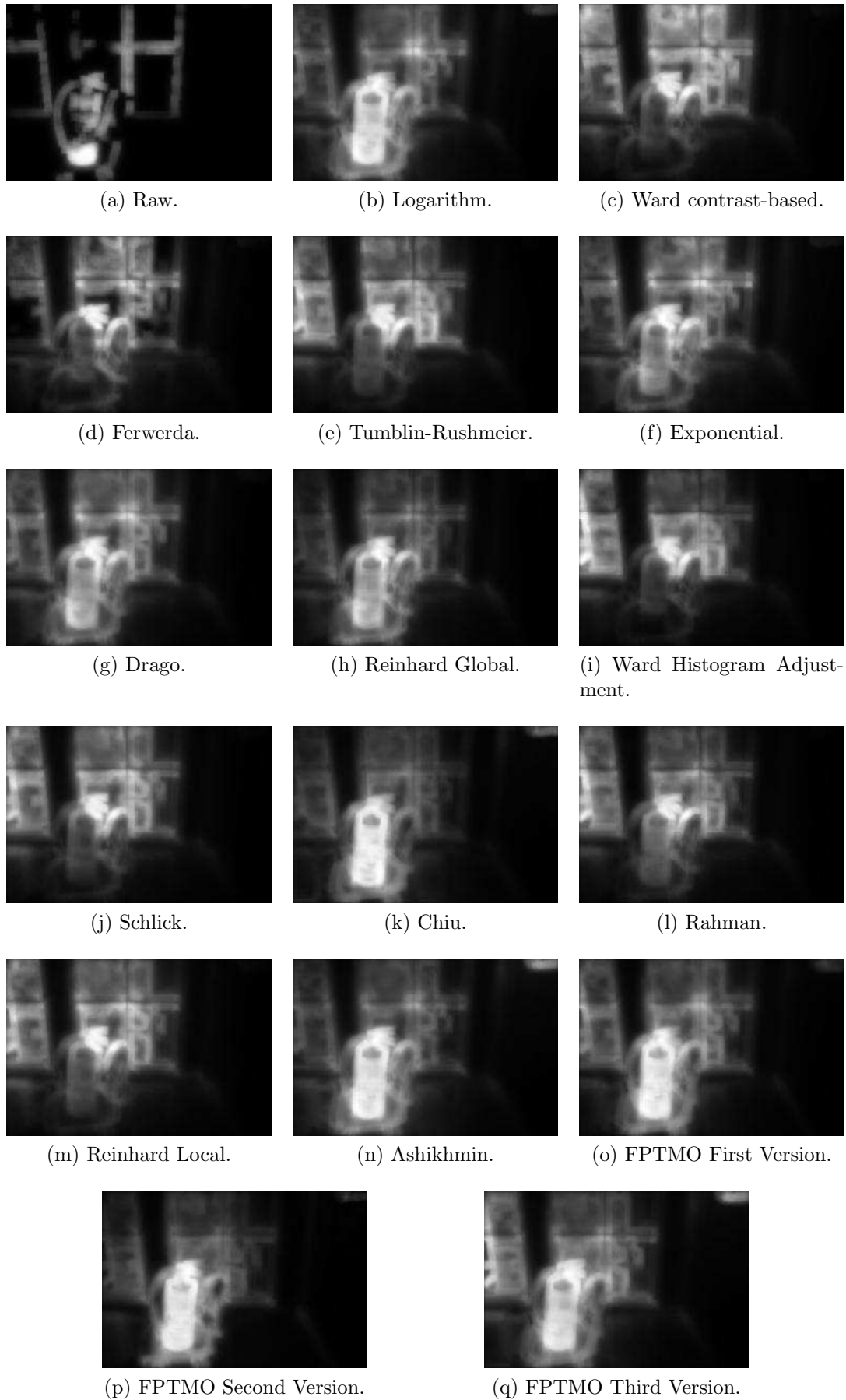
Figure D.2: Saliency maps of the raw and tone-mapped versions of the Fire Extinguisher image.

(a) Raw.

(b) Logarithm.

(c) Ward contrast-based.

(d) Ferwerda.

(e) Tumblin-Rushmeier.

(f) Exponential.

(g) Drago.

(h) Reinhard Global.

(i) Ward Histogram Adjustment.

(j) Schlick.

(k) Chiu.

(l) Rahman.

(m) Reinhard Local.

(n) Ashikhmin.

(o) FPTMO First Version.

(p) FPTMO Second Version.

(q) FPTMO Third Version.
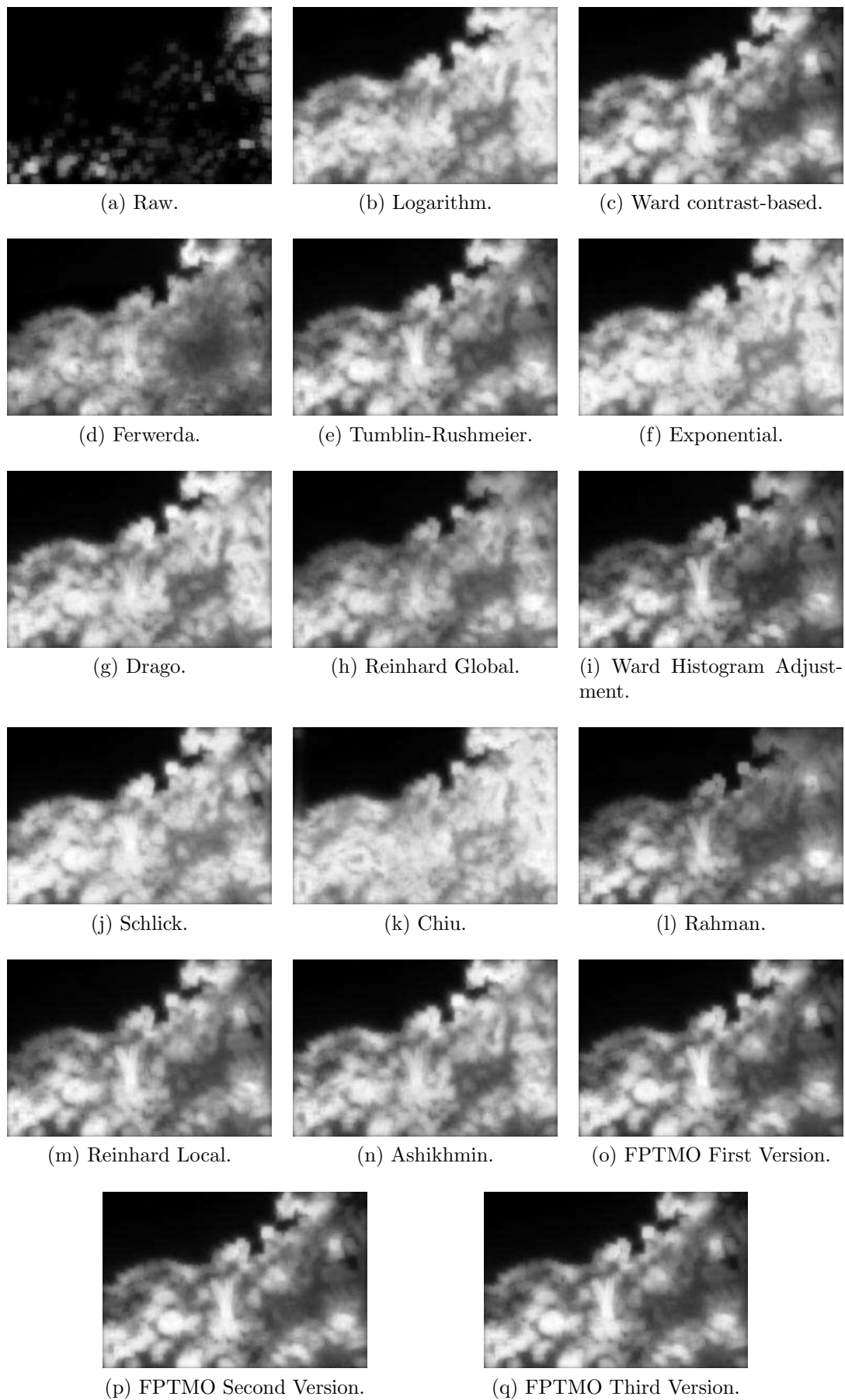
Figure D.3: Saliency maps of the raw and tone-mapped versions of the Flowers image.

(a) Raw.

(b) Logarithm.

(c) Ward contrast-based.

(d) Ferwerda.

(e) Tumblin-Rushmeier.

(f) Exponential.

(g) Drago.

(h) Reinhard Global.

(i) Ward Histogram Adjustment.

(j) Schlick.

(k) Chiu.

(l) Rahman.

(m) Reinhard Local.

(n) Ashikhmin.

(o) FPTMO First Version.

(p) FPTMO Second Version.

(q) FPTMO Third Version.
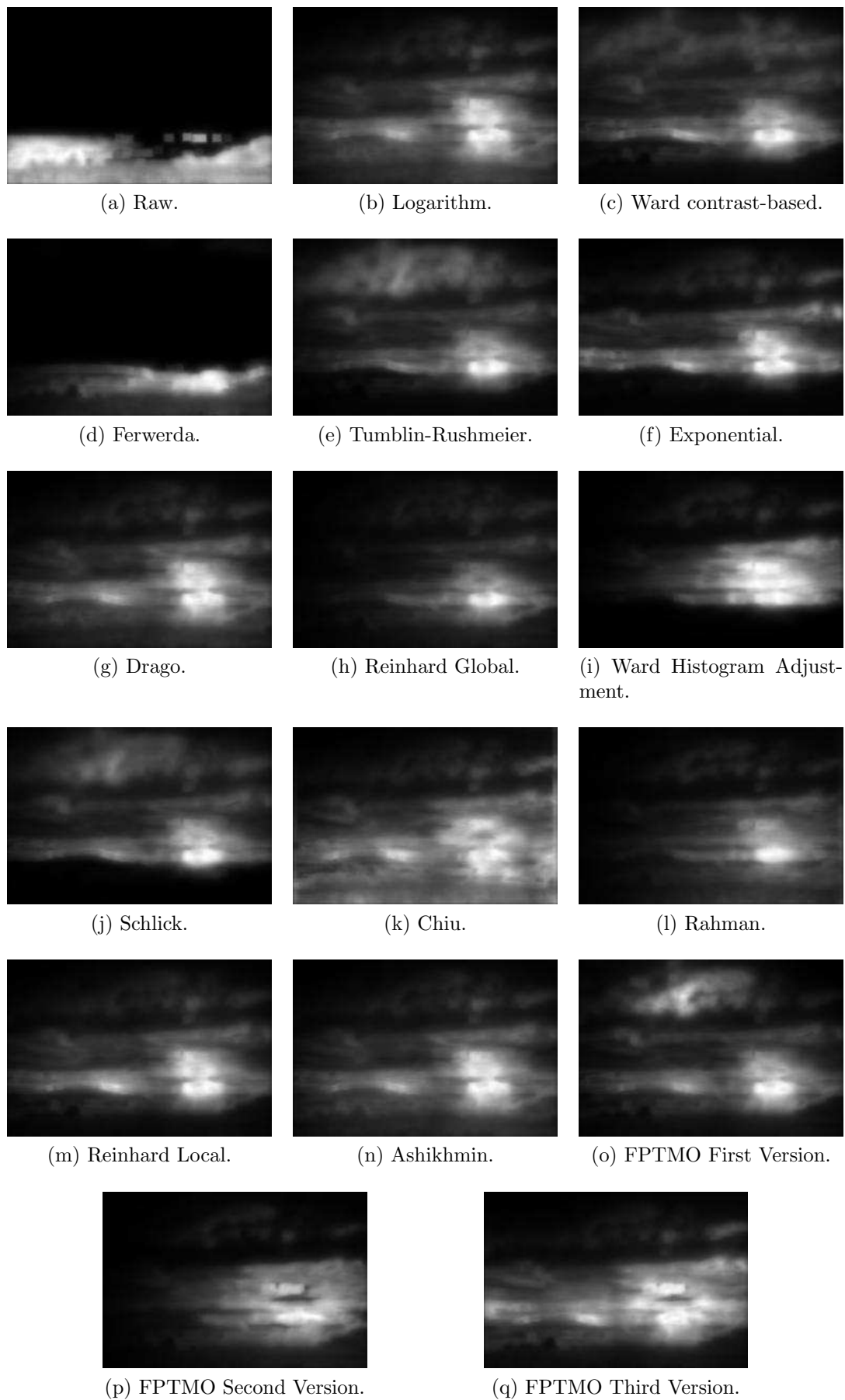
Figure D.4: Saliency maps of the raw and tone-mapped versions of the Sunset image.

164

(a) Raw.

(b) Logarithm.

(c) Ward contrast-based.

(d) Ferwerda.

(e) Tumblin-Rushmeier.

(f) Exponential.

(g) Drago.

(h) Reinhard Global.

(i) Ward Histogram Adjustment.

(j) Schlick.

(k) Chiu.

(l) Rahman.

(m) Reinhard Local.

(n) Ashikhmin.

(o) FPTMO First Version.

(p) FPTMO Second Version.

(q) FPTMO Third Version.
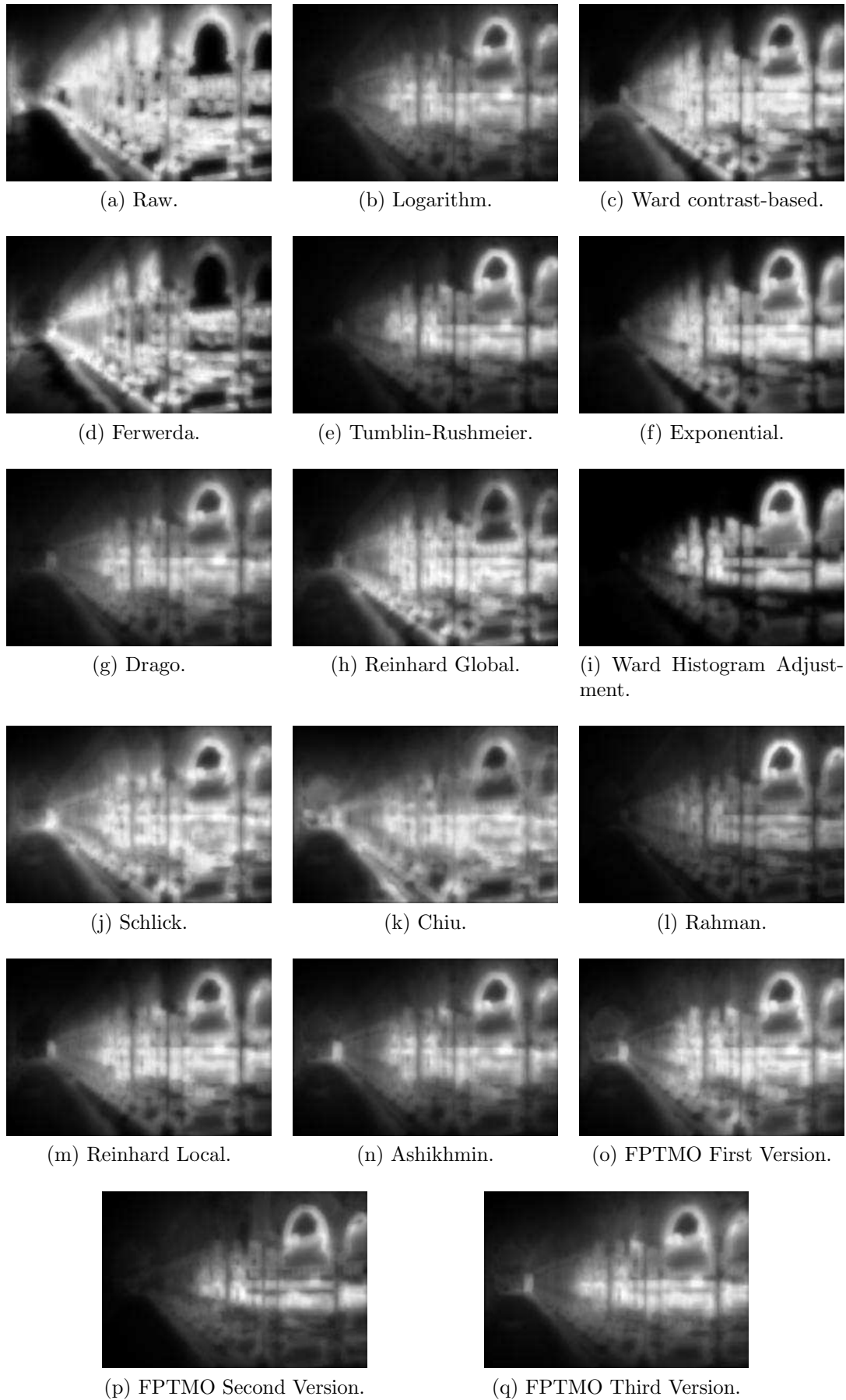
Figure D.5: Saliency maps of the raw and tone-mapped versions of the Courtyard image.

(a) Raw.

(b) Logarithm.

(c) Ward contrast-based.

(d) Ferwerda.

(e) Tumblin-Rushmeier.

(f) Exponential.

(g) Drago.

(h) Reinhard Global.

(i) Ward Histogram Adjustment.

(j) Schlick.

(k) Chiu.

(l) Rahman.

(m) Reinhard Local.

(n) Ashikhmin.

(o) FPTMO First Version.

(p) FPTMO Second Version.

(q) FPTMO Third Version.
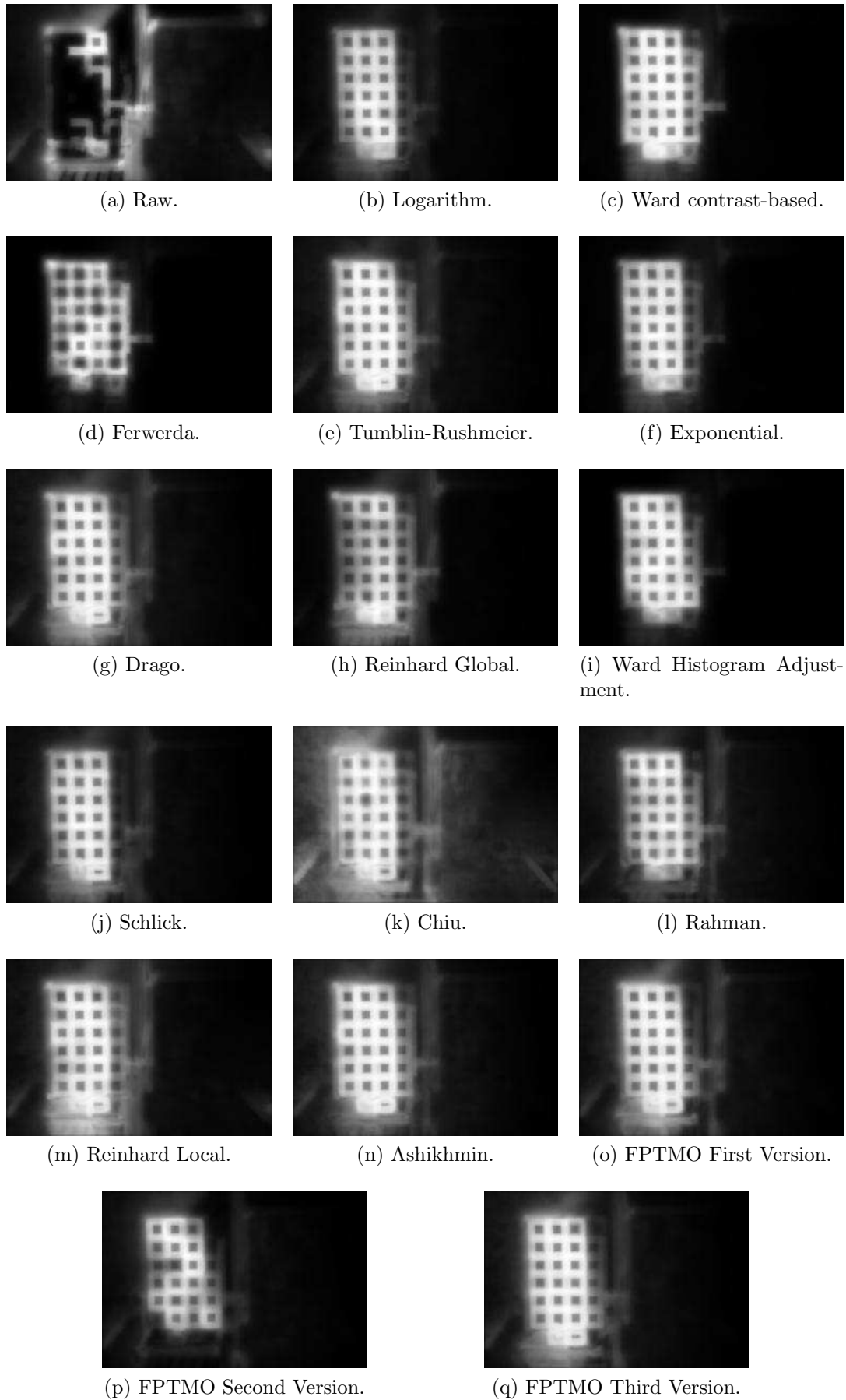
Figure D.6: Saliency maps of the raw and tone-mapped versions of the Color Chart 1 image.

(a) Raw.

(b) Logarithm.

(c) Ward contrast-based.

(d) Ferwerda.

(e) Tumblin-Rushmeier.

(f) Exponential.

(g) Drago.

(h) Reinhard Global.

(i) Ward Histogram Adjustment.

(j) Schlick.

(k) Chiu.

(l) Rahman.

(m) Reinhard Local.

(n) Ashikhmin.

(o) FPTMO First Version.

(p) FPTMO Second Version.

(q) FPTMO Third Version.
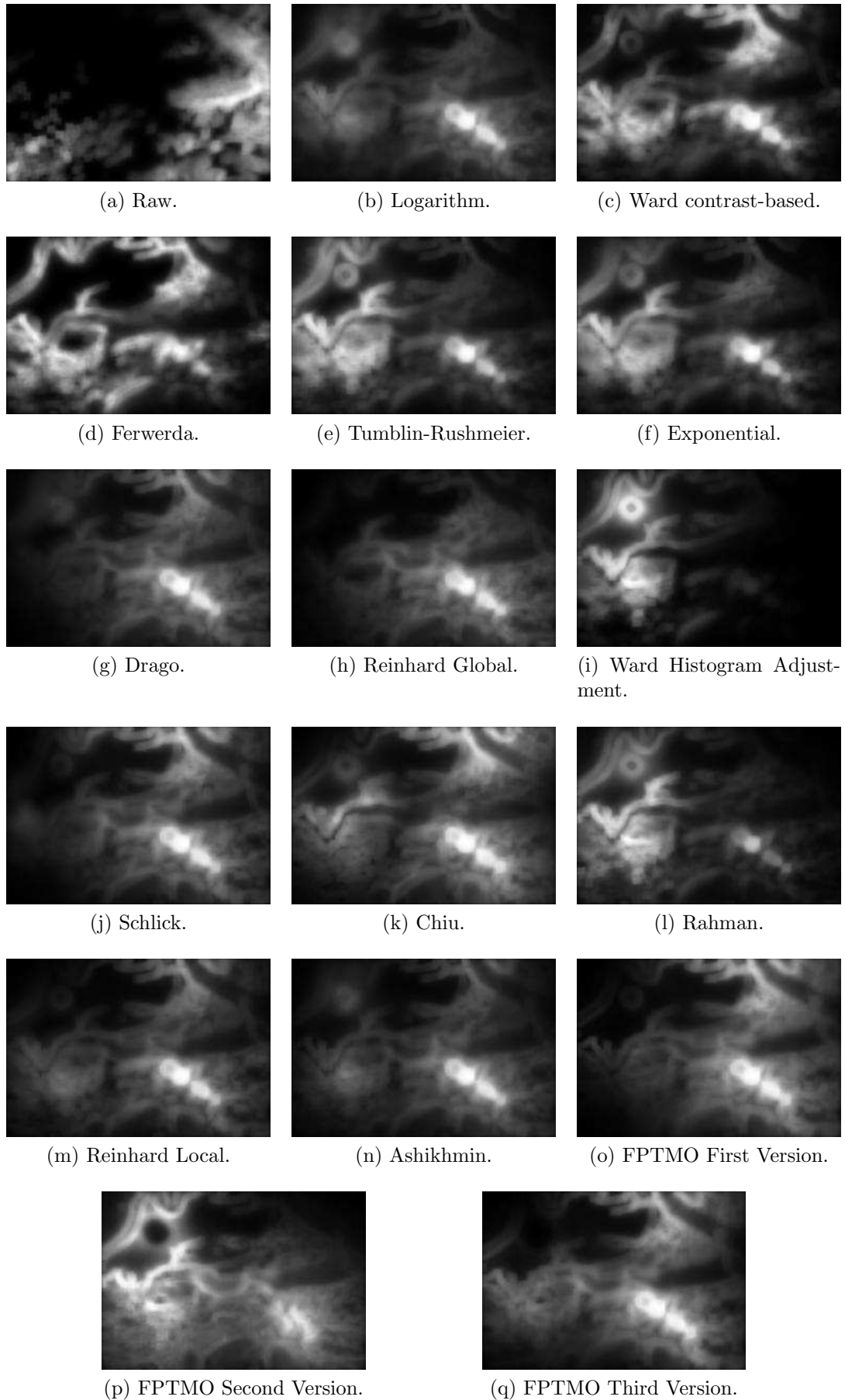
Figure D.7: Saliency maps of the raw and tone-mapped versions of the Beach image.

167

(a) Raw.

(b) Logarithm.

(c) Ward contrast-based.

(d) Ferwerda.

(e) Tumblin-Rushmeier.

(f) Exponential.

(g) Drago.

(h) Reinhard Global.

(i) Ward Histogram Adjustment.

(j) Schlick.

(k) Chiu.

(l) Rahman.

(m) Reinhard Local.

(n) Ashikhmin.

(o) FPTMO First Version.

(p) FPTMO Second Version.

(q) FPTMO Third Version.
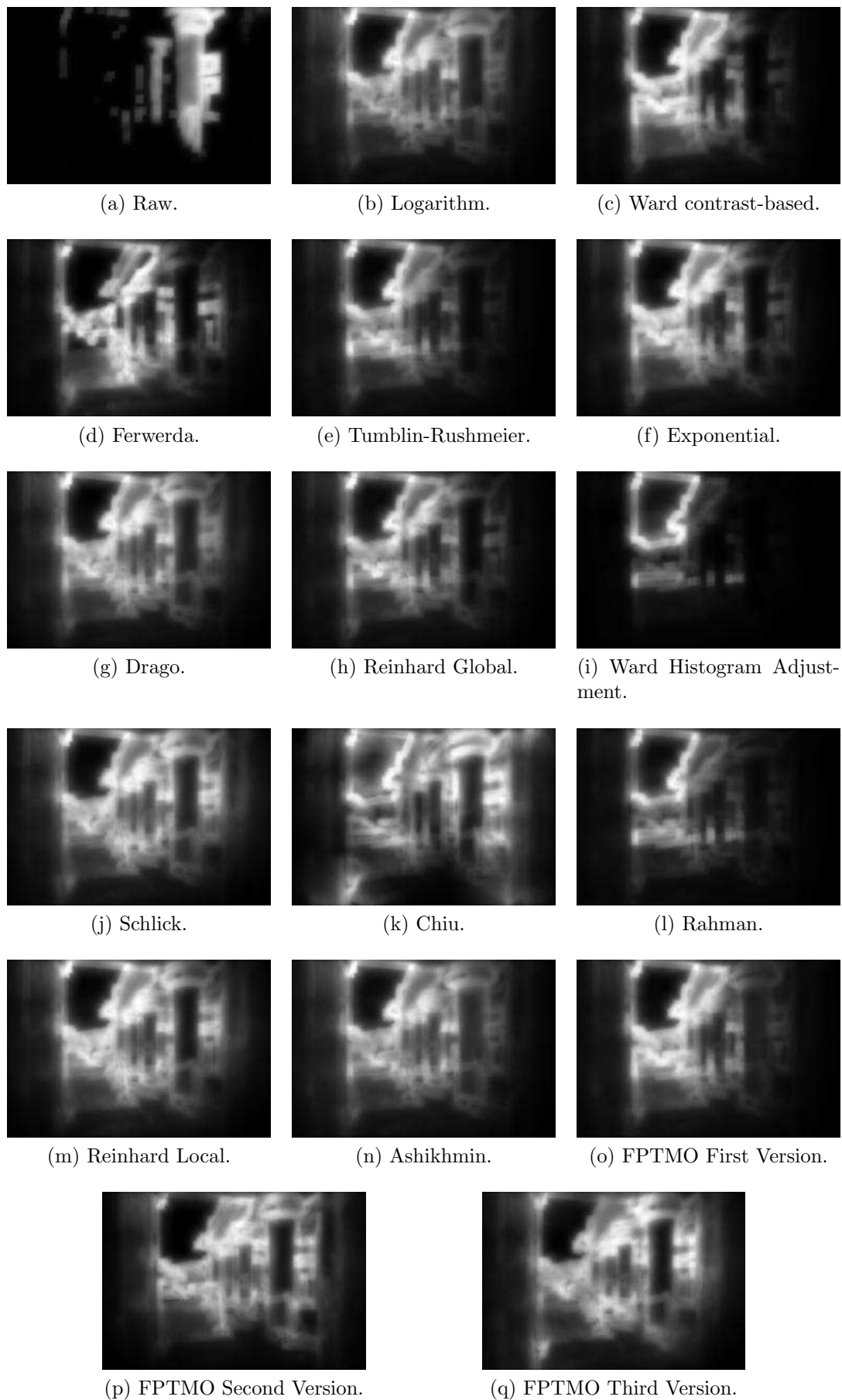
Figure D.8: Saliency maps of the raw and tone-mapped versions of the Palace image.

(a) Raw.

(b) Logarithm.

(c) Ward contrast-based.

(d) Ferwerda.

(e) Tumblin-Rushmeier.

(f) Exponential.

(g) Drago.

(h) Reinhard Global.

(i) Ward Histogram Adjustment.

(j) Schlick.

(k) Chiu.

(l) Rahman.

(m) Reinhard Local.

(n) Ashikhmin.

(o) FPTMO First Version.

(p) FPTMO Second Version.

(q) FPTMO Third Version.
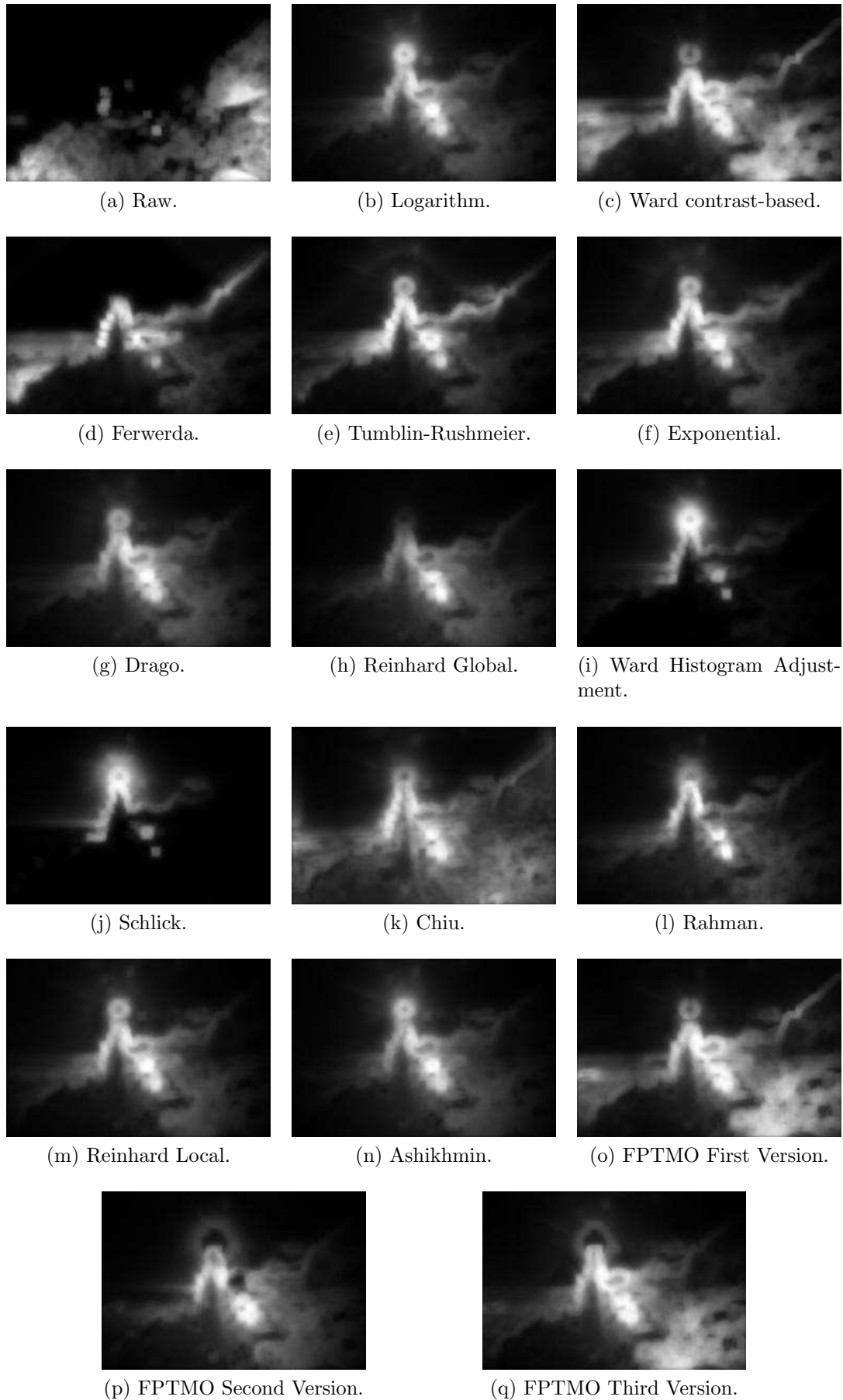
Figure D.9: Saliency maps of the raw and tone-mapped versions of the Stone Tower image.

(a) Raw.

(b) Logarithm.

(c) Ward contrast-based.

(d) Ferwerda.

(e) Tumblin-Rushmeier.

(f) Exponential.

(g) Drago.

(h) Reinhard Global.

(i) Ward Histogram Adjustment.

(j) Schlick.

(k) Chiu.

(l) Rahman.

(m) Reinhard Local.

(n) Ashikhmin.

(o) FPTMO First Version.

(p) FPTMO Second Version.
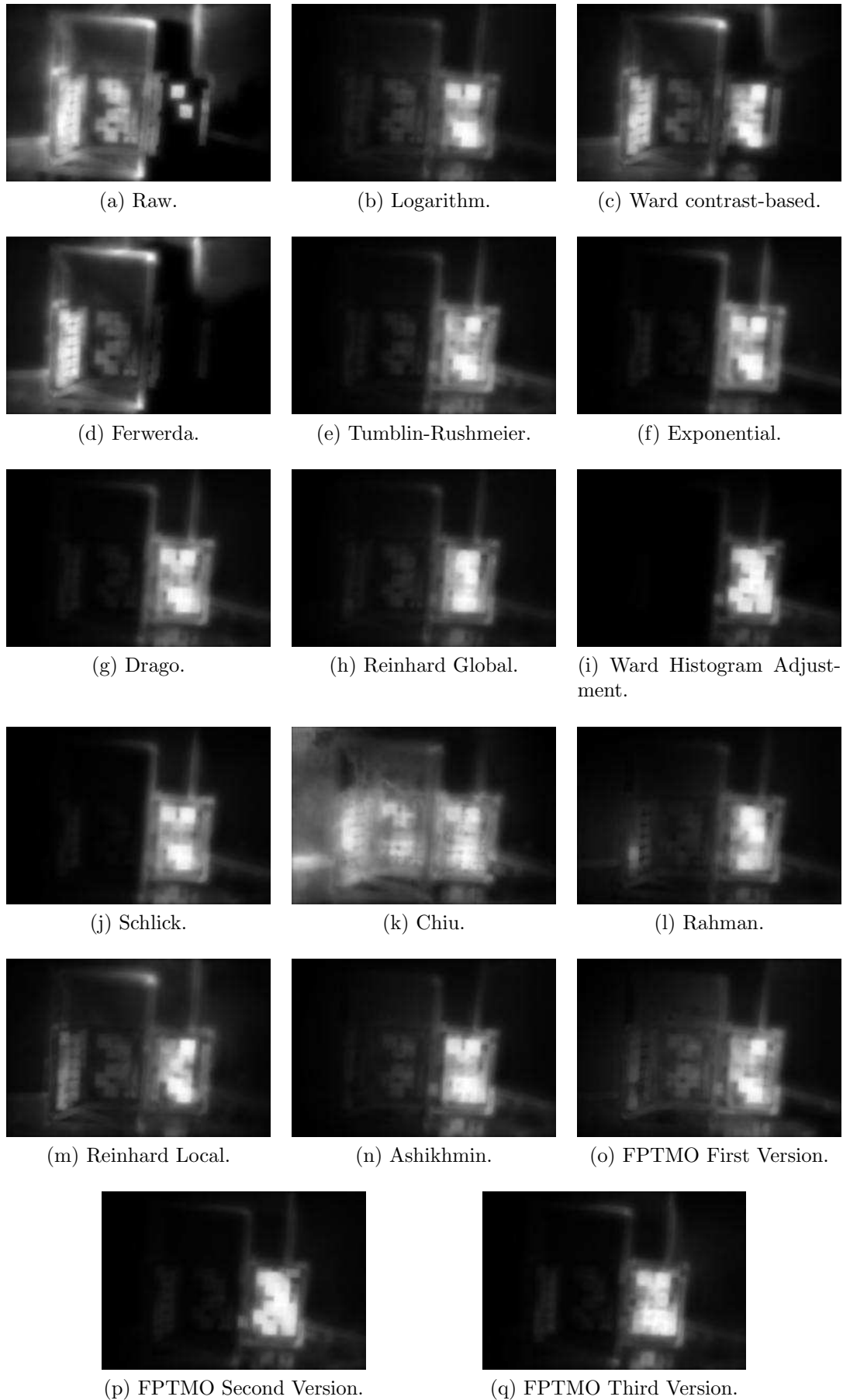
(q) FPTMO Third Version.

Figure D.10: Saliency maps of the raw and tone-mapped versions of the Color Chart 2 image.