Visual Based Filtering on Pixel Grids

Fabian Andres Prada Niño
Advisor: Diego Fernandes Nehab

July 2013

Contents

Introduction

A *pixel* is the unit of representation of color in a digital image. When such digital image is to be reconstructed in a traditional display, e.g., LCD and LED screens, the concept of *pixel* acquires a physical nature. Each *pixel* is now formed by chromatic components with distinct spatial location and geometry. These components are called *subpixels*. Since these components are independently addressed by the graphics hardware, it is more accurate to refer to *subpixels* (and not *pixels*) as the reconstruction units of an image in a screen.

On the last years it has been recognized the importance of taking into account the reconstruction device properties and the Human Visual System characteristics for the filtering process. The goal of these approaches is to attain a larger spatial resolution without produce notorious color artifacts. Our work belongs to this context.

One of the main inspirations to this work are the distinct pixel grids recently being introduced by display manufacturers . These new grids exhibits some of the goals that guide manufacturers designs: increase resolution, reduce power consumption and expand color gamut. The techniques we present provide a framework to validate these properties. Therefore, the aim of our work is assisting pixel grids design.

Contributions

Our main contribution is a formulation and solution of visual-based subpixel filtering in a continuous domain. Previous approaches (to our known) represented subpixels as chromatic impulses in sparse grids. These discrete formulations trade the exact geometry and alignment of subpixels by simplified models. Instead, our continuous domain formulation take advantage of accurate measures of both subpixel geometry and alignment. Our formulation also involves flexible representation of light spectrum and human visual system.

In this work we compute optimal filtering kernels on arbitrary multi-chromatic pixel grids. Previous approaches deal only with RGB grids, or proposed nonlinear filtering reductions on RGBW and RGBY grids[1].We also present simulated perceptual reconstructions on RGB, RGBG, RGBW and RGBY commercial devices to validate the quality of the preceived image reconstruction.

Our optimal reconstruction formulation, define subpixel coefficients as the solution to a least square problem. We describe a convolution-based representation of our optimality condition, and we present its spectral decomposition. This decomposition allows an efficiently parallelized solution computation. We implement a parallel Cholesky solver to this purpose which lead to satisfactory results.

Finally, we present an analysis of how subpixel chromatic dependence and large visual blurring conditions may lead to non-unique optimal subpixel configurations for a same image reconstruction (i.e., singular systems). This kind of discussions is omitted in most of previous research by assuming a fixed observation distance. To compute optimal reconstruction in singular systems we propose a two step optimization framework that first optimizes on quality and then on reconstruction cost term. This reconstruction term can be naturally associated to the energetic consumption of the device.

Document Overview

In Chapter 1 we present a discussion in some relevant topics that motivate our work. First, we describe the gains and risks of naive subpixel filtering: a larger resolution can be attained, but it can produce undesired color artefacts. Then we present our approach to simulate human visual system perception, and how this is important to validate a visual model and measures the quality of an image reconstruction. Finally, we introduce some of the pixel grids designed by manufactures in the last years, and briefly describe the kind of filtering challenges arising in these new grids.

In Chapter 2 we present some of the relevant works on subpixel filtering. Most of these works are specialized on RGB striped displays, and provide

[1]For instance, taking $W = \min\{R, G, B\}$, and compensations $R = R - W$,$G = G - W$,$B = B - W$. These nonlinear reductions seems to be against the purpose of increment spatial resolution.

discrete simplifications of optimal filtering.

The mathematical formulation and analysis of our optimization model is introduced in Chapter 3. The first order condition for optimality and the computation of optimal solution are presented in this Chapter. Cases of singularity of the optimal linear system are analysed in Chapter 4.

In Chapter 5 we discuss some implementation details. Chapter 6 is devoted to evaluate parameter variations of our optimization model, and present optimal linear filters for several pixel grids. Finally in Chapter 7 we present our conclusions and future directions of work.

In Chapter 8 we approach some of the theoretical background that supports our work. We present the basic characteristics of the Human Visual System (HVS), and describe some color representations and color transformations considered by our optimization model. We also include a description of the standard filtering techniques in the context of achromatic and trichromatic pixel grids.

The mathematical notation used in this work is explained in Chapter 9. This is a prerequisite for the lecture of Chapters 1, 3,4, and 8.

Chapter 1

Motivation

1.1 What is a pixel?

The concept of pixel is in the hearth of the digital image framework. Pixels are the units used to transmit, process, store, capture, and reproduce digital images. Due to its fundamental character, it is hard to give a precise definition of pixel. In this case, a simple definition is preferred:

Pixels are the ***units*** *providing full color representation of an image at certain position.*

We will study pixels in the context of display reconstruction elements.

The traditional approach to image filtering is based on the Sampling Theorem 9.3.1: the band of low frequencies, up to half cycle of the sampling rate, can be "preserved and reconstructed". When dealing with image reconstruction at a pixel grid, we can find some different interpretations of the Sampling Theorem. If we assume that pixels are indivisible reconstruction units, then, we should preserve frequencies up to half a cycle per pixel. On the other hand, if we assume that subpixels are independent reconstruction units, then, for a RGB striped display, we should preserve frequencies up to a sixth a of cycle per pixel in the horizontal direction.

The two approaches described above are extrema of the filtering problem: the first taking pixels as indivisible elements, and the second taking subpixels as completely independent elements. Our concern in this work is to explore filtering strategies in between.

In Figure 1.5, we compare reconstructions obtained by filtering with an scaled interpolative bspline of third degree centered at subpixel position. We observe that for a pixel scaled filter, with cut-off frequency at 1/2 cycles per pixels, the result is too blurred. Instead, for a subpixel scaled filter, with cut-off at 3/2 cycles per pixels, the result is plagued of color artefacts. A good trade off between little color fringing and sharp results is attained for a scaled filter which remove frequencies above 3/4 cycles per pixel.

1.2 Human Visual System Simulation

In Figure 1.5 we show two perceptual artefacts that can be produced due to inappropriate filtering. By definition, an observer is in position to discern when an image reconstruction in a display contains too much color fringing[1] or too much blur. But, what about machines? How can we express in machine language that a certain image will be perceived with any of these artefacts? This is a real problem!

First, we should identify the two agents in the perceptual relation: the observer and the reconstruction. In order to determine the presence of perceptual artefacts, we must simulate both agents. The models of observer and reconstruction are mathematically developed in Chapter 3.

Our model of the reconstructed image, is given by a **raw**-reconstruction (3.5) which takes into account the reconstruction coeffcients and the subpixels emission surface. On the other hand, the visual model is based in contrast sensitivity for trichromatic vision. The visual model transform the **raw**-reconstruction to a opponent color space and applies blurring in each channel independently (as in SCIELAB model 8.2.3). The result of this process is called the **visual**-reconstruction (3.7). In practice, the **visual**-reconstruction is transformed back from opponent color space to sRGB for display purposes.

[1]By color fringe, we refer to the reproduction and perception of non-existent color contours around shape edges.

The **visual**-reconstruction takes a central role in our model. It is used for three important tasks:

- Provide an insight in the accuracy of the visual model.
- Simulate image perception in arbitrary displays.
- Define a domain of comparison between the current reconstruction and an a reference "optimal" reconstruction.

Applications of our human visual simulation are provided in Figures 1.6 and 1.7. In Figure 1.6 we simulate how an observer at regular viewing conditions[2] would perceived a detail of the reconstructions provided in Figure 1.5. On the other hand, Figure 1.7 shows the simulated perception of a fixed reconstruction[3] for an observer who is moving away the screen.

1.3 Pixel Grids

The new generations of mobile devices are introducing innovative designs of pixel grids. These new display technologies pursue three specific goals: energy efficiency, resolution enhance, and color gamut expansion.

One of the strategies adopted to reduce energy consumption is to increase subpixel size without harming effective resolution. In displays such as the Nexus One 1.2a and the PenTile RGBW 1.3c, the surface covered by two subpixels is equivalent to the surface of a complete RGB pixel. This reduction in the number of subpixels per area reduce the number of transistors of the display. This not only benefit the energy efficiency but also makes devices thinner. Larger subpixels improve the ratio of transmissive area (i.e., area subpixel/area pixel), which allows reproduction of brighter images.

RGB

In the context of RGB displays, Samsumg proposes a new pixel structure in their Galaxy Note 2 1.1b. Instead of traditional vertical stripes, this

[2] This mean, viewing 20 inches away to a 90ppi screen.

[3]The second from top to bottom in Figure 1.5

display proposes horizontal orientation for the red and green subpixels. Another interesting subpixel geometry is presented by Dell 1905FP 1.1c monitor, which implements arrow shaped subpixels.

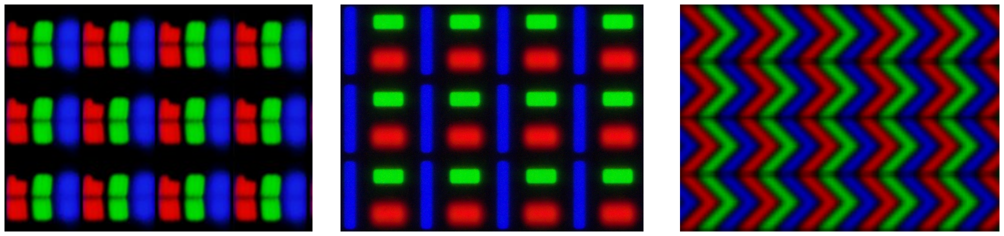

(a) Iphone Retina (b) Galaxy Note 2 (c) Dell 1905FP

Figure 1.1: RGB displays.

RGBG

The purpose of RGBG displays is to mimic light spectrum sensitivity of the human visual system. The distribution of cones in the retina makes the visual system more sensitive to green color than red or blue. Consequently, more luminance resolution is transmitted through the green channel than through the others. This principle is precisely applied in RGBG displays. In the same surface covered by a striped RGB pixel, a RGBG display has one green subpixel and other subpixel which is either red or blue. Since, a RGBG display maintains the same amount of green detail than a RGB display, it closely preserves the effective resolution.

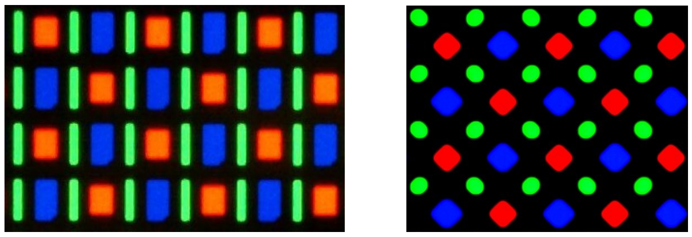

(a) Nexus One (b) Galaxy 4

Figure 1.2: RGBG displays.

RGBW

According to Elliot et al.[2], traditional RGB displays are forced to trade off between high brightness and wide color gamut. The introduction of an unsaturated white primary overcomes this difficulty. A white subpixel in the pixel grid would improve the reproduction of darker saturated color, as well as brighter unsaturated ones. This allows a closer reconstruction of natural images.

Manufacturers of PenTile RGBW 1.3c [11] claim that their eight-subpixel cell allows equal resolution than a striped RGB displays, and reduces the amount of subpixels by a third. The argument exposed is as follows: To reproduce a pattern of black and white vertical lines, Pentile RGBW requires an iterative sequence of two consecutive subpixel columns on and off. This same pattern of black and white vertical lines is reproduced in a RGB display by a sequence of three consecutive subpixel columns on and off.

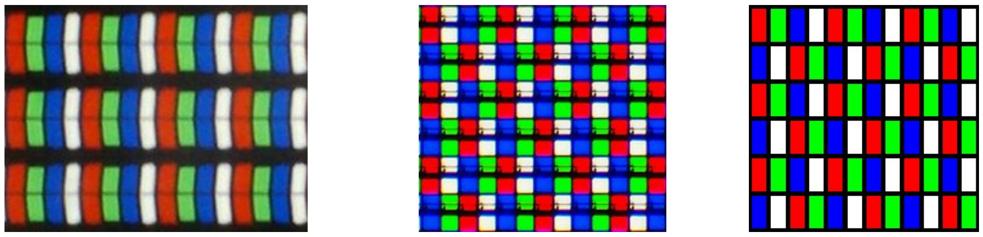

(a) Ricoh GR Digital IV (b) Motoroloa Droid X2 (c) Pentile RGBW

Figure 1.3: RGBW displays.

RGBY

Other approach to expand the color gamut is presented by Sharp through their Quattron RGBY display 1.4. According to the manufacturer, Quattron technology provides smoother colour gradations and brighter pictures [14]. This new technology has received serious critics. Defenderes of current RGB standards consider that the introduction of a yellow primary would not improve image transmision or reproduction [15]. The lack of content specifically designed for the Y primary is another big obstacle that Quattron must face.

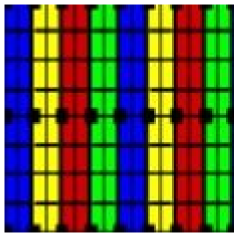


Figure 1.4: Sharp Xgen Quattron RGBY.

1.4 Filtering in General Pixel Grids

The problem of filtering in arbitrary pixel grids is quite interesting and complex. Here we want to provide some insights on it.

1.4.1 Thrichromatic Grids

We say that a pixel grid is thrichromatic when its subpixels correspond to any of three distinct primaries. The traditional RGB striped display is an example of a thrichromatic pixel grid. The RGBG displays manufactured by Samsumg also lay in this category.

The standard linear filtering schemes discussed in section 8.3.1 are valid for any thrichromatic pixel grid. The only condition to care is to equilibrate the maximum intensity of each channel. For instance, in a pixel grid with a 2:1:1 relation between green, red, and blue subpixels, we are required to scale the green channel filters to achieve a white color texture for a DC input. Clearly, this scaling preserves the linearity of a filtering scheme.

1.4.2 Multichromatic Grids

According to the trichromatic vision model, reconstruction in a display with more than three primaries may present “color dependence”. By “color dependence” we mean that the same image reconstruction can be attained for several set of reconstruction coefficients. In section 4.1.2 we will analyse “color dependence” in our optimization problem.

Suppose we are given a Quattron RGBY pixel grid 1.4, and we are asked to reconstruct a texture for which we know its RGB channels. A natural approach in this case is to filter each channel with an antialiasing filter, and then set

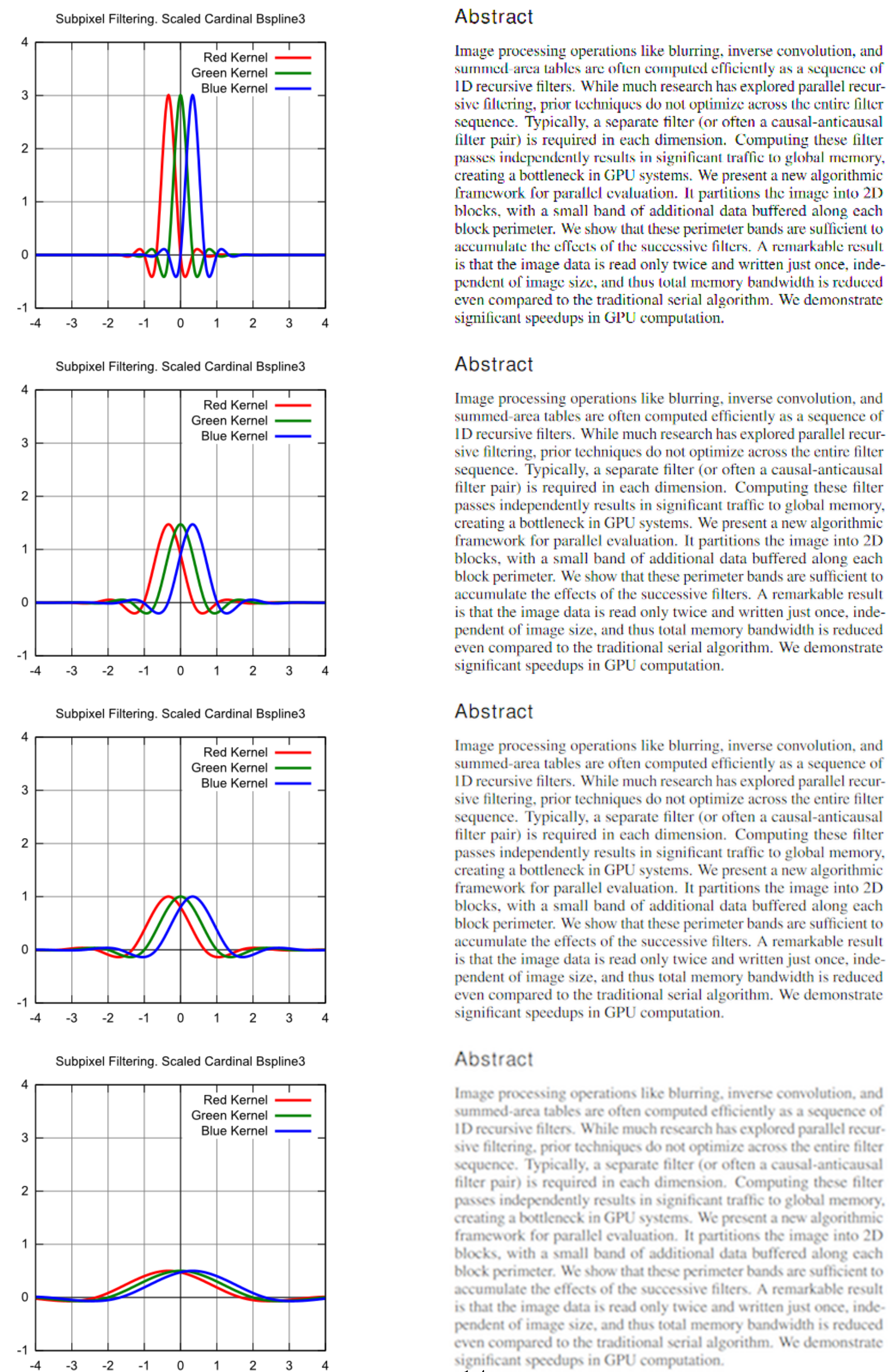
$$c_Y = \min(R, G) \quad c_R = R - c_Y \quad c_G = G - c_Y \tag{1.1}$$

In the process above, the antialiasing step is linear, but the color compensation step is non linear. Therefore, this filtering scheme is a nonlinear process as a whole. Can you imagine a linear filtering scheme to reproduce RGB textures in a RGBY device?

Intuitively, our linear filtering scheme should guarantee that a red-scale texture is only reconstructed using red subpixels, and the same situation should apply for green-scale and blue-scale textures. If we impose these conditions we are inmediately discarding the use of subpixel Y. From a mathematical point of view, we can a reconstruct a red texture using an active Y subpixel: take $R = 1/2$, $Y = 1/2$, $B = 0$ and $G = -1/2$. Clearly, this out of the scope for any phisycal device!

Some of the limitations just described are a consequence of working in the standard RGB space. Instead, if we are required to exploit the space resolution of the reconstruction device (i.e., avoid idle subpixels) and reduce color artefacts, we need to work in a more elaborated model of the human visual system. This justifies our desire of simulating the human visual system as described in Section 1.2. A more detailed description of the visual model we follow is found Section 8.2.3.

In Chapter 6, we present results of our filtering model applied to trichromatic and multichromatic grids. As we show, we can find linear filtering schemes with all active subpixels attaining satisfactory results in multichromatic grids.



Abstract

Image processing operations like blurring, inverse convolution, and summed-area tables are often computed efficiently as a sequence of 1D recursive filters. While much research has explored parallel recursive filtering, prior techniques do not optimize across the entire filter sequence. Typically, a separate filter (or often a causal-anticausal filter pair) is required in each dimension. Computing these filter passes independently results in significant traffic to global memory, creating a bottleneck in GPU systems. We present a new algorithmic framework for parallel evaluation. It partitions the image into 2D blocks, with a small band of additional data buffered along each block perimeter. We show that these perimeter bands are sufficient to accumulate the effects of the successive filters. A remarkable result is that the image data is read only twice and written just once, independent of image size, and thus total memory bandwidth is reduced even compared to the traditional serial algorithm. We demonstrate significant speedups in GPU computation.

Abstract

Image processing operations like blurring, inverse convolution, and summed-area tables are often computed efficiently as a sequence of 1D recursive filters. While much research has explored parallel recursive filtering, prior techniques do not optimize across the entire filter sequence. Typically, a separate filter (or often a causal-anticausal filter pair) is required in each dimension. Computing these filter passes independently results in significant traffic to global memory, creating a bottleneck in GPU systems. We present a new algorithmic framework for parallel evaluation. It partitions the image into 2D blocks, with a small band of additional data buffered along each block perimeter. We show that these perimeter bands are sufficient to accumulate the effects of the successive filters. A remarkable result is that the image data is read only twice and written just once, independent of image size, and thus total memory bandwidth is reduced even compared to the traditional serial algorithm. We demonstrate significant speedups in GPU computation.

Abstract

Image processing operations like blurring, inverse convolution, and summed-area tables are often computed efficiently as a sequence of 1D recursive filters. While much research has explored parallel recursive filtering, prior techniques do not optimize across the entire filter sequence. Typically, a separate filter (or often a causal-anticausal filter pair) is required in each dimension. Computing these filter passes independently results in significant traffic to global memory, creating a bottleneck in GPU systems. We present a new algorithmic framework for parallel evaluation. It partitions the image into 2D blocks, with a small band of additional data buffered along each block perimeter. We show that these perimeter bands are sufficient to accumulate the effects of the successive filters. A remarkable result is that the image data is read only twice and written just once, independent of image size, and thus total memory bandwidth is reduced even compared to the traditional serial algorithm. We demonstrate significant speedups in GPU computation.

Abstract

Image processing operations like blurring, inverse convolution, and summed-area tables are often computed efficiently as a sequence of 1D recursive filters. While much research has explored parallel recursive filtering, prior techniques do not optimize across the entire filter sequence. Typically, a separate filter (or often a causal-anticausal filter pair) is required in each dimension. Computing these filter passes independently results in significant traffic to global memory, creating a bottleneck in GPU systems. We present a new algorithmic framework for parallel evaluation. It partitions the image into 2D blocks, with a small band of additional data buffered along each block perimeter. We show that these perimeter bands are sufficient to accumulate the effects of the successive filters. A remarkable result is that the image data is read only twice and written just once, independent of image size, and thus total memory bandwidth is reduced even compared to the traditional serial algorithm. We demonstrate significant speedups in GPU computation.

Figure 1.5: Subpixel filtering with scaled kernels. Cut-off frequency at 3/2 (top), 3/4 (mid-top), 1/2 (mid-bottom) and 1/4 (bottom) cycles per pixel.

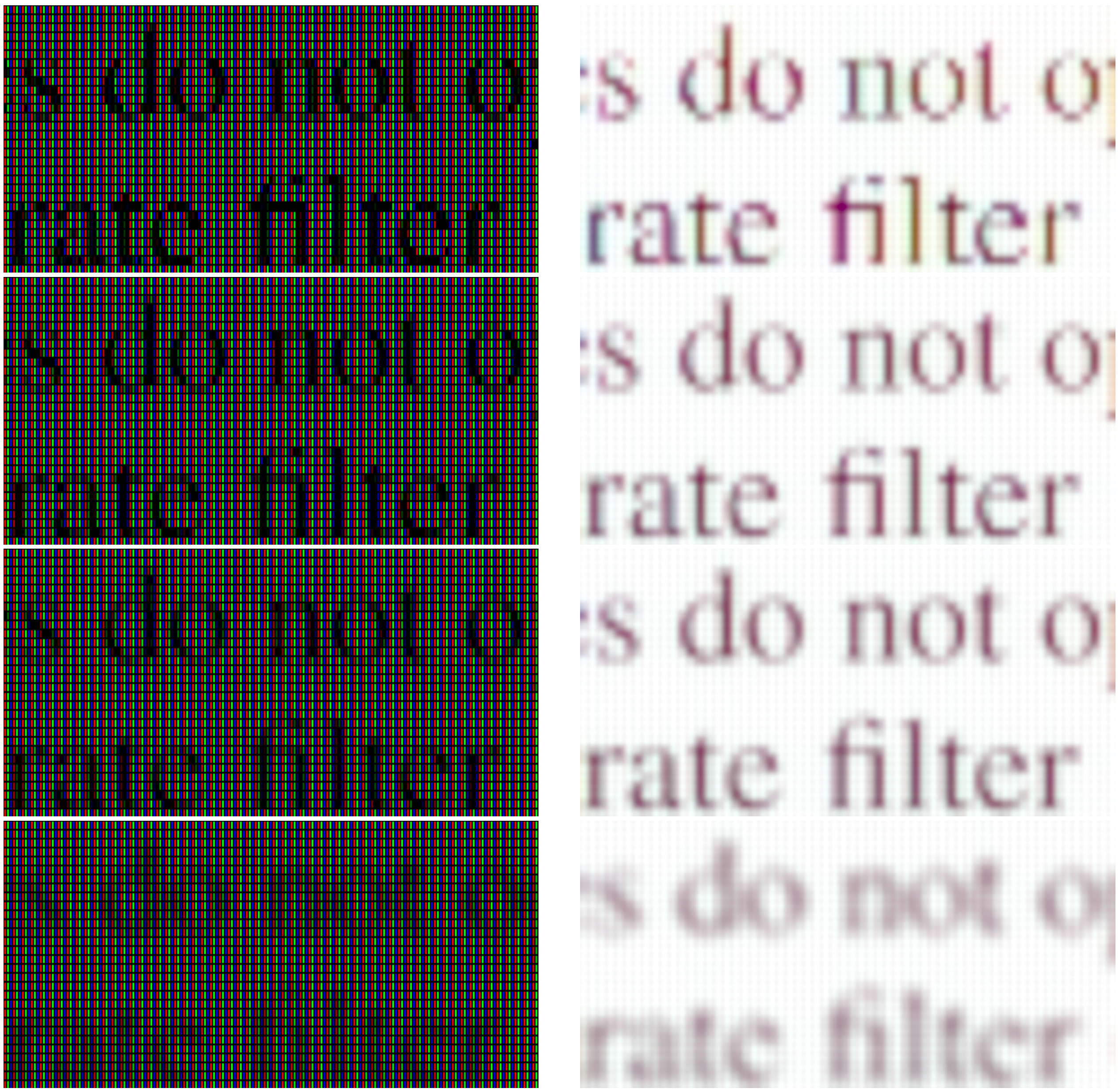
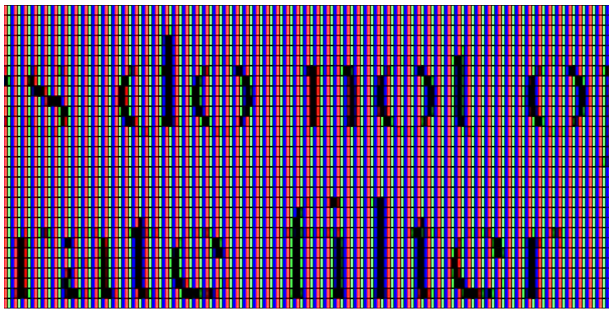
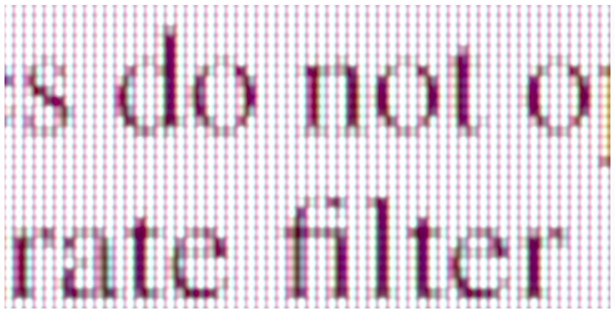


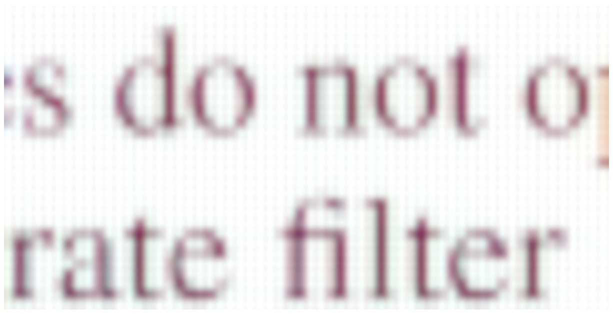
Figure 1.6: Simulation of raw and visual reconstruction of the images in the right column of Figure 1.5.



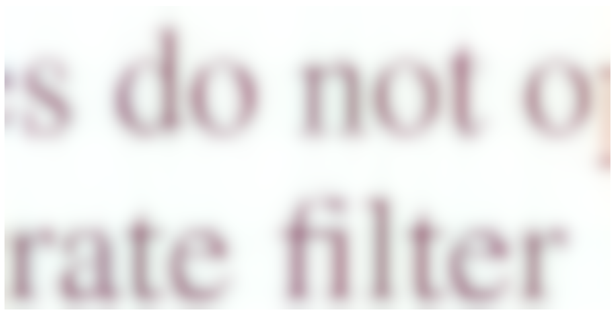
(a) 0.2 inches



(b) 10 inches



(c) 20 inches



(d) 38 inches

Figure 1.7: Simulation of visual blurring for several viewing distances on a striped RGB 90 pixels per inch display.

Chapter 2

Related Works

Kajiya and Ullner (1981)

One of the pioneering works on proposing text filtering based on display attributes and human visual system is due to Kajiya and Ullner [7]. In this work, the authors study text rasterization on CRTs. The authors claim that traditional filtering schemes such as triangular filtering effectively solve the aliasing problem, but are still suboptimal. They observe that these approaches disregard the reconstruction kernel of the display, which in case of a CRT, are Gaussian spots.

Kajiya and Ullner expose the mismatch between filtering with a triangle kernel and reconstructing with a Gaussian spot, and suggest large room for improvement. They propose a reconstruction scheme that minimizes the L_2 error between the original texture and the image reconstructed from the Gaussian spots:

$$J(\ldots, x_{-1}, x_0, x_1, \ldots) = \int_{-\infty}^{\infty} \left| f(\xi) - \sum_{i=-\infty}^{\infty} x_i g_i(\xi) \right|^2 d\xi \qquad (2.1)$$

This unconstrained problem is reduced to a linear problem $Ax = b$. Matrix A corresponds to Gaussian spots autocorrelation, and vector b to cross correlation between Gaussian spots and texture f. Matrix A, called Grammian matrix, is symmetric and block Toeplitz[1].

[1]The mathematical formulation of our optimization problem is an extension of Kajiya and Ullner work. We consider more elaborated display and visual models.

They claim that the solution to the unconstrained optimization is weak, in the sense that it does not guarantee values in the $[0, 1]$ range for the reconstruction coefficients. He notices that truncating the unconstrained solution to $[0, 1]$ could lead to poor results, specially if the quadratic problem is very eccentric (i.e., poorly conditioned).

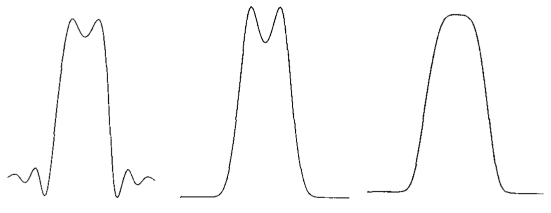


Figure 2.1: Optimal reconstruction of a box from Gaussian kernels. Unconstrained optimum (left), positivity constrained optimum (middle), and range constrained optimum (right). Adapted from [7].

Therefore, the authors also consider constrained optimization for the same energy function 2.1. The solution to the constrained problem is given by Kuhn Tucker conditions, and is computed by iterative methods (gradient descent in Kajiya and Ullner work). This is computationally more expensive than solving the unconstrained problem, but produces a more accurate result. The constrained solution overcomes the notorious ringing artifacts of the unconstrained case.

Kajiya and Ullner acknowledge that computing squared differences between linear scale intensities (8.5a) is suboptimal. Instead, a more accurate visual metric is required. They discuss a single channel model that weights more accurately errors for different luminance levels. Finally, they present multichannel models as a more robust approach. The energy function associated to these visual models are nonlinear. Due to the complexities of optimizing this kind of energy functions, the optimal solutions were not computed.

Platt (2000)

Platt [12] discusses subpixel filtering on RGB striped displays. He observes that traditional filtering schemes treat pixels as indivisible reconstruction elements. Instead, subpixels of a RGB striped display are addressed independently, and some resolution can be gained by filtering at subpixel level.

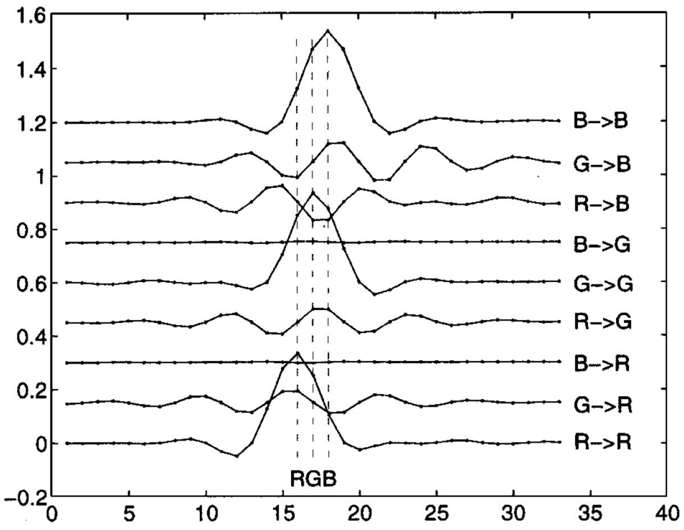


Figure 2.2: Crossed Filters. Compare with our result 6.2.1. Adapted from [12].

Platt proposes an optimization model that seeks resolution gains without producing color artefacts. This model is based on the SCIELAB [17] perceptual metric. Platt analyses and discusses his model in the frequency domain, and computes the optimal solution by solving the associated linear system.

Platt shows that solving the linear system is equivalent to filtering the texture with a 9-filter set[2]: $R \to R$, $R \to G$, $R \to B$, $G \to R$, $G \to G$, $G \to B$, $B \to R$, $B \to G$, and $B \to B$. These filters are illustrated in Figure 2.2. The coefficients of the reconstructed red channel are given by filtering

[2]As we discuss in section 8.3.1, any linear scheme for subpixel filtering can be represented as a set of kernels, one for each pair of subpixel and input channel.

the input red, green and blue channels with $R \to R$, $G \to R$, $B \to R$, respectively, and summing up. The coefficients for the reconstructed green and blue channel are obtained analogously.

The filters $R \to R$, $G \to G$ and $B \to B$ are antialiasing kernels centered in their respective subpixel position. On the other hand, crossed filters are highly oscillatory or null kernels.

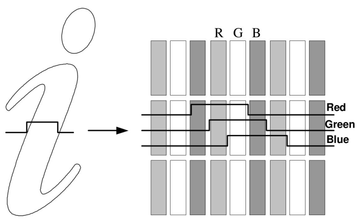


Figure 2.3: Cleartype Filtering. Adapted from [1].

Betrisey et al. (2000)

The work of Platt was seminal for the development of Microsoft Cleartype [10]. Betrisey et al.[1] present details about real time implementation of Microsoft Cleartype. Instead of using the 9 optimal filters described by Platt, they used a technique called RGB decimation. This technique disregards the crossed filters, and simplify the $R \to R$, $G \to G$ and $B \to B$ filters to box kernels. Specifically, Betrisey et al. sample the input channels six times horizontally per pixel, and take a uniform mean of support 1 centered at subpixel position (Figure 2.3). They claim that this approach produce results of sharpness comparable to optimal filtering. Beside its computational speed, this approach is exempt of $[0, 1]$ truncation.

Klompenhauewer and De Haan (2003)

Klompenhauewer and De Haan [8] analyse the gain of resolutions due to subpixel sampling in RGB striped displays. The authors represent each subpixel as the characteristic function of a rectangular region covering one third of the pixel area and shifted to its respective position. No visual features are considered in the reconstruction model.

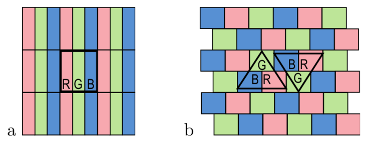


Figure 2.4: RGB striped and Delta Nabla pixels. Adapted from [8].

Klompenhauewer and De Haan compare the frequency spectra of an ideal achromatic texture[3] with the spectra of reconstructions using sampling at pixel center, for one side, and sampling at subpixels positions on the other side. This spectra analysis is done in YUV space. They present a simulated reconstruction of a high frequencies texture (zoneplate) using each sampling scheme. They claim that subpixel sampling produces a shift of aliasing from luminance to chrominance. As they indicate, this is desirable since humans are less sensitive to chrominance errors than to luminace errors.

It is also discussed subpixel sampling for a Delta Nabla (see Figure 2.4) pixel grid, and some simulated reconstructions are exhibited for both text and natural images. Their simulated reconstructions correspond to images in a large scale with pixels activated in one primary color. They apply a low pass filter on their simulated reconstructions to obtain brigther images and induce some effects of human visual system[4].

Messing et al. (2003)

Messing et al. [9] compare the efficiency of certain pixel geometries and sampling schemes on covering the limits of perception of the human visual system. They consider two kind of pixel grids: striped RGB and Pentile II.

Messing et al. develop their analysis by first introducing the visible boundaries of the 2D contrast sensitivity functions for luminance and chrominance channels. They argue that the luminance channel (Y) is independently sampled at R and G subpixels, while the chrominance channels (U and V) are limited by the sampling rate of B and R, respectively. Following this

[3]The spectra of the ideal achromatic texture is assumed to be 1 for frequencies below 1/2 cycles per pixel, and 0 for frequencies above.

[4]This blurring is not done in the context of a visual model, i.e. an opponent color space, as in our case.

idea, they justify that the PentileII display covers all the visible frequencies, but using fewer subpixels than a twice resolution RGB striped display.

They propose a subpixel rendering scheme based in the optimization of a quality energy function similar to the proposed by Platt [12], together with a set of constraints. They assume that each subpixel is a full color unit, and they use the constraints to implicitly specify the color of the subpixel[5]. For instance, the full color subpixel associated to a green subpixel, is constraint to be of intensity 0 in red and blue.

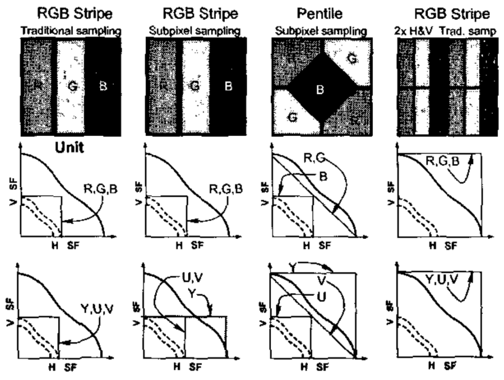


Figure 2.5: Visible frequencies reproducible by pixels and sampling schemes. Adapted from [9].

The optimal solution to their model is given by the solution of a linear system, with Lagrange multipliers associated to each constraint. The Lagrange multipliers indicate how the reconstruction error would be diminished if the respective constraints were lifted. The authors claim that this information provide the gains in resolution by adding subpixels to the current display.

Farrell et al. (2011)

Farrell et al.[6] seeks to measure the results quality for a set of discrete filters. The discrete filters evaluated are all symmetric and support 5 (in subpixel units). As in Cleartype, these discrete filters act independently on

[5]In our model we assume that each type of subpixel has a fixed light spectrum, so this kind of constraints are unnecessary.

sampled R, G, and B channels.

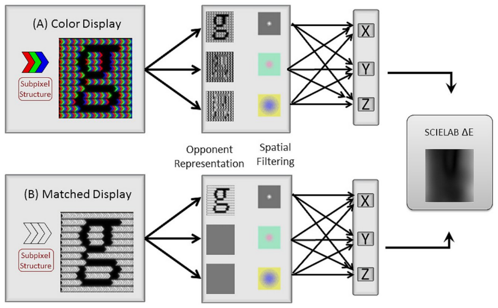


Figure 2.6: Simulated Reconstruction and Matched Display. Comparison by SCIELAB error metric. Adapted from [6].

The first step of their approach is to simulate image reconstruction. This is done by rendering an image with intensity-scaled point spread function for each subpixel[6]. The figure in the top left corner of 2.6 provide an example of this simulation. Then, the authors set as quality reference to a reconstruction in a matched display. As observed from the bottom left of 2.6, a matched display has the same number of subpixels, but these are achromatic instead of monochromatic. Finally, they evaluate the performance of a discrete filter, by computing the perceptual SCIELAB error metric between the quality reference and the reconstruction provided by the filter.

A good compromise between resolution and low chromatic artefacts was obtained by the discrete filter $(-0.1, 0.2, 0.8, 0.2, -0.1)$.[7] This is partially corroborated by a psychophysical test the authors carried out with three observers, three letters, two fonts and 2 type of displays.

Fang et al. (2013)

Fang et al. [5, 4] propose downsampling schemes for traditional RGB displays which extends cutoff frequencies beyond Nyquist limit. They define

[6]This is also the approach we follow for our **raw**-reconstruction.

[7]The presence of negative weights exhibits some relation between this discrete filter and ideal low pass filtering. In Figure 1.5 we show that good results are indeed attained by scaling antialiasing filters.

a luminance function, which is a uniform mean of the R, G, and B image coefficients. The authors claim that high frequencies of different colors tend to be similar. Based on this hypothesis, they are able to move (by applying a linear transform) from R, G, B, to a space Y, C_1, C_2, where C_1, C_2 are low frequency components (see Figure 2.7).

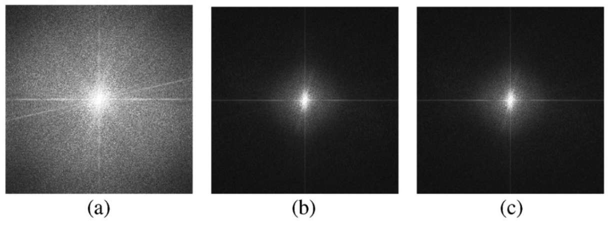


Figure 2.7: (a) Y, (b) C_1, and(c) C_2 spectrum. Adapted from [5].

They explore three downsampling techniques: Direct Pixel Downsampling (DPD), Direct Subpixel Based Downsampling (DSD), and Diagonal DSD (DDSD). They observe that DSD produces $Y, C1, C2$ spectral components to align vertically; since $C1, C2$ are low frequency components, they are able to expand the horizontal cutoff frequency of Y, without introducing alias. In the case of DDSD,the spectral components of Y,$C1$, and $C2$ aligns diagonally, thus allowing a larger spectrum of Y both vertically and horizontally.

Fang et al. propose some measures to compare several subpixel filtering techniques. They propose a Luminace Sharpness Measure (LSM) to identify the presence of high frequencies on the downsampled reconstruction, and a Luminance Aliasing Measure (LAM) that establishes the proximity of the reconstruction to a aliased reference or to a blurred reference. They also propose a Luminance Contrast Measure (LCM), claiming that images with larger contrast are perceptually desirable. Finally it is proposed a Chrominance Distortion Measure (DCM), which compares the U and V component of the reconstructed image, with the respective components of a reference reconstruction with no color distortion.

Engelhardt et al. (2013)

Engelhardt et al.[3] present an approach to construct visual-based kernels for subpixel filtering on regular pixel grids. The approach follow very much Platt[12] works, and its main novelty is the adaptability to more complex pixel grids such as RGBW and RGBG. The purposes and methods discussed in this paper are close to our model, but there are several notorius differences. First, the authors compute optimal visual-based kernel through a discrete domain formulation, then, subpixels are represented as an arrange of impulses on a discrete grid. This representation disregards pixel geometry and forces (or assume) perfect alignment of subpixels to a sparse grid. Also, the construction of the sparse grid is pixel dependent, what implies certain user (or designer) interaction[8]. Instead, our continuous domain formulation overcomes these limitations.

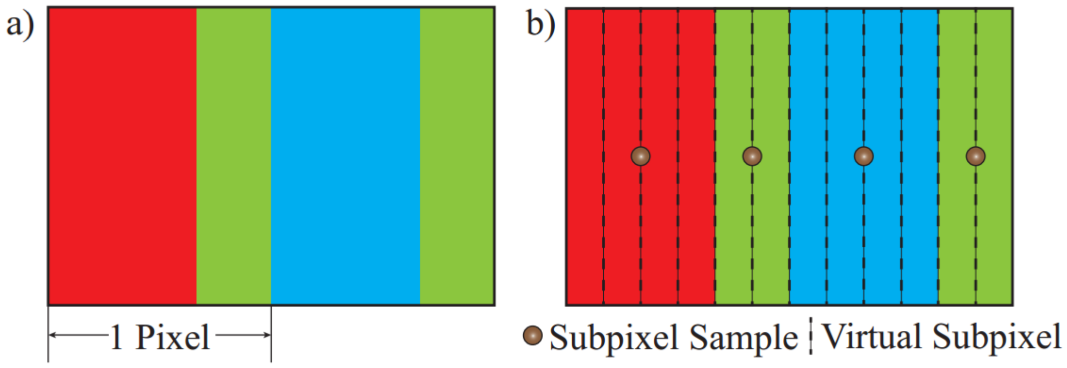


Figure 2.8: Pixel Discretization. Adapted from [3].

Once the problem is solved in the discrete domain, the authors fit a continuous function over the optimal discrete filters. The author looks for parameters a, b, on a functional form $f(x, y) = \operatorname{sinc} a\sqrt{x^2+y^2}\exp(-\frac{x^2+y^2}{b})$, that best fit the model. The selection of this functional form suits the oscilatory and exponential decay of these filters. However, it only allows two degrees of freedom and forces the kernel to be radially symmetric. This last property seems not to be coherent to the asymmetries of the physical pixel

[8]Consider for instance designing a sparse grid to represent a Galaxy Note 6.8 pixel. This would require introducing several idle variables to the system.

grid.

The authors also discus the problem of optimal subpixel filtering in the context of color dependence. Their approach is very different to ours since they use a standard nonlinear mapping: in a RGBW pixel you take $W = \min(R, G, B)$ and subtracts W to each R, G, B. Observe that this approach would reconstruct a grayscale image using W pixels only. This seems a contradiction to the goal of subpixel filtering: increase the spatial resolution[9]. Finally, the authors do not aboard the problem of system singularity introduced by the color dependence. Maybe these kind of problems are not present in the fixed distance visual model of the authors.

[9]Our human visual system simulations show that best results for grayscale images are attianed when all subpixels are active.

Chapter 3

Linear Problem

3.1 Introduction

The work here developed is an approximation to the following question:

What is the most visually accurate reconstruction attainable for a given texture in certain display?

We will approach this problem by proposing an optimization model that considers both the Human Visual System and display attributes.

Our approach to simulate the Human Visual System response will follow the SCIELAB model 8.2.3. This visual model considers the following physiological attributes, described in Chapter 8:

1. *Trichromatic vision.* Due to the presence of three different types of cones in the retina : L, M and S.

2. *Opponent Color Space Decomposition.* Due to the decorrelation of the spectral response of L, M and S cones.

3. *Visual Blur.* Due to the density and distribution of L, M and S cones.

4. *Luminance Perception.* Due to the nonlinear response of cones and rods.

The specific notation used for the problem formulation is explained in 9.2.

3.2 Parameters

For clarity, we will divide our model in three stages:

Raw Level

The **raw**-texture is the object to be reproduced. It is represented by a multichannel decomposition of its light spectrum,

$$f : (\Omega_C) \times \{\boldsymbol{h}\} \to \mathbb{R} \tag{3.1}$$

The display is a grid of pixels with identical subpixel structure. For a pixel with n subpixel elements, we represent the **raw**-pixel by the multichannel vector,

$$\varphi = \begin{pmatrix} \varphi_1 \\ \varphi_2 \\ \vdots \\ \varphi_n \end{pmatrix} \tag{3.2}$$

Each φ_s is a multichannel decomposition of the light spectrum emitted by the subpixel surface. $\varphi_s : (\Omega_C) \times \{\boldsymbol{h}\} \to \mathbb{R}$ is such that the domain origin coincides with the pixel center.

We will write $\varphi_{s,i,j}$ to denote the s^{th} reconstruction element (i.e., subpixel) in the $(i,j)^{\text{th}}$ pixel of the image. This is precisely given by the expression:

$$\varphi_{s,i,j} = \varphi_s(\cdot - i, \cdot - j) \tag{3.3}$$

The reconstruction coefficients indicate the intensity of subpixels at the each pixel position. Reconstruction coefficients will be represented by the discrete channel vector,

$$\boldsymbol{c} = \begin{pmatrix} \boldsymbol{c}_1 \\ \boldsymbol{c}_2 \\ \vdots \\ \boldsymbol{c}_n \end{pmatrix} \tag{3.4}$$

Each $\boldsymbol{c}_s$ is a channel, $\boldsymbol{c}_s : \Omega_D \to \mathbb{R}$, indicating the intensities of the s^{th} subpixel. For a fixed **raw**-pixel φ, we define the **raw**-reconstruction associated to coefficients $\boldsymbol{c}$ by,

$$\begin{aligned} g_{\boldsymbol{c}} &= \sum_{s,i,j} \boldsymbol{c}_{s,i,j} \varphi_{s,i,j} \\ &= \sum_{s} \boldsymbol{c}_s * \varphi_s \\ &= \boldsymbol{c}^\top * \varphi \end{aligned} \tag{3.5}$$

Remark

In equation 3.4 we defined $\boldsymbol{c}$ as a channel vector. In practice, we will overload the symbol $\boldsymbol{c}$ to also denote the multichannel vector formed by its repeated copies in depth. In equation 3.5, $\boldsymbol{c}$ is a multichannel vector of depth h, where $\boldsymbol{c}^1 = \boldsymbol{c}^2 = \cdots = \boldsymbol{c}^h$.

Visual Level

We denote by $A : \mathbb{R}^h \to \mathbb{R}^d$ the linear transformation from the multichannel decomposition of the light spectrum to the opponent color space.

The visual blurring kernels will be denoted by the multichannel,

$$k_A = \left(k_{A_1}; k_{A_2}; \ldots; k_{A_d}\right) \tag{3.6}$$

where each k_{A_i} is a channel, $k_{A_i} : \Omega_D \to \mathbb{R}$, that acts independently on the respective component of the opponent color space.

By blurring the opponent color decomposition of the **raw**-texture and **raw**-reconstruction, we obtain the **visual**-texture and the **visual**-reconstruction, respectively. These are multichannels given by expressions:

$$\check{f} = A(f) * k_A, \qquad \check{g} = A(g_{\boldsymbol{c}}) * k_A \tag{3.7}$$

The expression for the **visual**-reconstruction can be written in the form:

$$\begin{aligned} A(g_{\boldsymbol{c}}) * k_A &= A(\boldsymbol{c}^\top * \varphi) * k_A \\ &= \boldsymbol{c}^\top * (A(\varphi) * k_A) \end{aligned} \tag{3.8}$$

The multichannel vector $\check{\varphi} = A(\varphi) * k_A$ will be called the **visual**-pixel.

Metric Level

Let $B : \mathbb{R}^d \to \mathbb{R}^e$ be the linear transformation from opponent color space to the metric space. By applying this transformation to **visual**-texture and **visual**-reconstruction, we obtain the **metric**-texture and the **metric**-reconstruction, respectively. These are multichannels given by the expressions:

$$\bar{f} = B(A(f) * k_A), \qquad \bar{g}_{\boldsymbol{c}} = B(A(g_{\boldsymbol{c}}) * k_A) \tag{3.9}$$

We can rewrite the **metric**-reconstruction in the form:

$$\begin{aligned} B(A(g_{\boldsymbol{c}}) * k_A) &= B(\boldsymbol{c}^\top * (A(\varphi) * k_A)) \\ &= \boldsymbol{c}^\top * (B(A(\varphi) * k_A)) \\ &= \boldsymbol{c}^\top * \bar{\varphi} \end{aligned} \tag{3.10}$$

The multichannel vector $\bar{\varphi} = B(A(\varphi_s) * k_A)$ will be called the **metric**-pixel.

Remark

We will be specially concerned with the SCIELAB opponent color space and their respective blurring kernels 8.2.3. However others color spaces and blurring kernels could be considered. We apply a metric transform B to obtain a color space where L_2 distance between colors adjust better to perceptual distance. In practice, B could represent a scaling of the opponent color space, or a transformation to another color space.

3.3 Problem Formulation

Problem 3.1. *Let $V_{B,A,\varphi}$ denote the reconstruction space defined by* ***raw****-subpixels φ, visual model A and metric transformation B. Let $\bar{f}$ denote the* ***metric****-texture. Our aim is to find* ***metric****-reconstruction, $\bar{g}_{\boldsymbol{c}} \in V_{B,A,\varphi}$, that minimizes:*

$$||\bar{f} - \bar{g}_{\boldsymbol{c}}||^2 \tag{3.11}$$

Since $\bar{g}_{\boldsymbol{c}}$ is completely characterized by the reconstruction coefficients $\boldsymbol{c}$, we can evidence this parameter, and restate the problem as follows:

Let f be a texture to be reconstructed as an $W \times H$ image. Our objective is to find $\boldsymbol{c} : (\mathbb{Z}_{W\times H} \times \{\boldsymbol{n}\}) \to \mathbb{R}$ that minimizes,

$$J(\boldsymbol{c}) = ||\bar{f} - \boldsymbol{c}^\top * \bar{\varphi}||^2 \tag{3.12}$$

3.4 Problem Analysis

The problem stated above is an unconstrained convex[1] quadratic problem on the reconstruction coefficients $\boldsymbol{c}$. Since the reconstruction space $V_{B,A,\varphi}$, is a finite dimensional vector subspace of L_2, there is a unique $\bar{g}_{\boldsymbol{c}} \in V_{B,A,\varphi}$ that minimizes distance to $\bar{f}$. Whenever the reconstruction elements are linear independent, there is a unique representation $\boldsymbol{c}$ for this optimal.

For the unconstrained problem, first order conditions are sufficient to guarantee optimality. Expressing 3.12 as an inner product, we obtain,

$$\begin{aligned} J(\boldsymbol{c}) &= \langle \bar{f} - \boldsymbol{c}^\top * \bar{\varphi}, \bar{f} - \boldsymbol{c}^\top * \bar{\varphi} \rangle \\ &= \langle \bar{f}, \bar{f} \rangle - 2\langle \bar{f}, \boldsymbol{c}^\top * \bar{\varphi} \rangle + \langle \boldsymbol{c}^\top * \bar{\varphi}, \boldsymbol{c}^\top * \bar{\varphi} \rangle \\ &= \langle \bar{f}, \bar{f} \rangle - 2\langle [\bar{f} * \bar{\varphi}^\vee]^+, \boldsymbol{c} \rangle + \langle [\bar{\varphi}^\vee * \bar{\varphi}^\top]^+ * \boldsymbol{c}, \boldsymbol{c} \rangle \end{aligned} \tag{3.13}$$

The first order condition is given by $\frac{\partial J}{\partial \boldsymbol{c}} = 0$. This corresponds to equation:

$$[\bar{\varphi}^\vee * \bar{\varphi}^\top]^+ * \boldsymbol{c} = [\bar{f} * \bar{\varphi}^\vee]^+ \tag{3.14}$$

In aim of clarity, we will denote,

$$\boldsymbol{m} = [\bar{\varphi} * (\bar{\varphi}^\vee)^\top]^+, \qquad \boldsymbol{b} = [\bar{f} * \bar{\varphi}^\vee]^+ \tag{3.15}$$

$$\boldsymbol{m}_{sr} = [\bar{\varphi}_r * \bar{\varphi}_s^\vee]^+, \qquad \boldsymbol{b}_s = [\bar{f} * \bar{\varphi}_s^\vee]^+ \tag{3.16}$$

$$\boldsymbol{m} = \begin{pmatrix} \boldsymbol{m}_{11} & \boldsymbol{m}_{12} & \cdots & \boldsymbol{m}_{1n} \\ \boldsymbol{m}_{21} & \boldsymbol{m}_{22} & \cdots & \boldsymbol{m}_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{m}_{n1} & \boldsymbol{m}_{n2} & \cdots & \boldsymbol{m}_{nn} \end{pmatrix}, \qquad \boldsymbol{b} = \begin{pmatrix} \boldsymbol{b}_1 \\ \boldsymbol{b}_2 \\ \vdots \\ \boldsymbol{b}_n \end{pmatrix} \tag{3.17}$$

[1]The convexity of the problem follows from lemma 3.1.

Here $\boldsymbol{b}$ is a discrete channel vector of dimensions $n \times 1$, and $\boldsymbol{m}$ is a discrete channel matrix of dimensions $n \times n$. The energy function and optimal condition can now be expressed as:

$$J(\boldsymbol{c}) = \langle \boldsymbol{m} * \boldsymbol{c} - \boldsymbol{b}, \boldsymbol{c} \rangle + \langle \bar{f}, \bar{f} \rangle \tag{3.18}$$

$$\boldsymbol{m} * \boldsymbol{c} = \boldsymbol{b} \tag{3.19}$$

Remarks

Our functional $J(\boldsymbol{c})$ can be written in matrix form as

$$J(c) = c^t M c - 2b^t c + d, \tag{3.20}$$

where c is the vector representation of the reconstructions coefficients, and M is the Hessian of the minimization function. M is called the Gramian matrix and corresponds to subpixels correlations. Indexing vector coordinates by triplets (s, i, j), refering to s^{th} subpixel at $(i, j)^{\text{th}}$ image position, entries in 3.20 are given by:

$$\begin{aligned} M_{sij,rkl} &= \langle \bar{\varphi}_{r,k,l}, \bar{\varphi}_{s,i,j} \rangle & b_{sij} &= \langle \bar{f}, \bar{\varphi}_{s,i,j} \rangle \quad d = \langle \bar{f}, \bar{f} \rangle \\ &= [\bar{\varphi}_r * \bar{\varphi}_s^{\vee}]^{+}_{i-k,j-l} & &= [\bar{f} * \bar{\varphi}_s^{\vee}]^{+}_{i,j} \\ &= \boldsymbol{m}_{sr}(i-k, j-l) & &= \boldsymbol{b}_s(i, j) \end{aligned} \tag{3.21}$$

The Hesssian M is a Hermitian positive semidefinite matrix. This is a corollary of the following lemma:

Lema 3.1. *Let $\{\varphi_1, \varphi_2, \ldots, \varphi_n, w\}$ be a set of functions defined from $\Omega \subset \mathbb{R}^n$ to $\mathbb{R}$, and $w \geq 0$. Let H be the matrix defined by:*

$$H_{ij} = \int_{\Omega} w(x) \langle \varphi_i(x), \varphi_j(x) \rangle dx$$

Then H is Hermitian positive semidefinite.

3.5 Solution via Spectral Decomposition

Applying Discrete Fourier Transform on both sides of 3.19, we deduce that for all $(u, v) \in \{\boldsymbol{W}\} \times \{\boldsymbol{H}\}$, it must be satisfied,

$$\hat{\boldsymbol{m}}(u, v)\hat{\boldsymbol{c}}(u, v) = \hat{\boldsymbol{b}}(u, v) \tag{3.22}$$

In matrix form this condition is written as,

$$\begin{pmatrix} \hat{\boldsymbol{m}}_{11}(u,v) & \hat{\boldsymbol{m}}_{12}(u,v) & \cdots & \hat{\boldsymbol{m}}_{1n}(u,v) \\ \hat{\boldsymbol{m}}_{21}(u,v) & \hat{\boldsymbol{m}}_{22}(u,v) & \cdots & \hat{\boldsymbol{m}}_{2n}(u,v) \\ \vdots & \vdots & \ddots & \vdots \\ \hat{\boldsymbol{m}}_{n1}(u,v) & \hat{\boldsymbol{m}}_{n2}(u,v) & \cdots & \hat{\boldsymbol{m}}_{nn}(u,v) \end{pmatrix} \begin{pmatrix} \hat{\boldsymbol{c}}_1(u,v) \\ \hat{\boldsymbol{c}}_2(u,v) \\ \vdots \\ \hat{\boldsymbol{c}}_n(u,v) \end{pmatrix} = \begin{pmatrix} \hat{\boldsymbol{b}}_1(u,v) \\ \hat{\boldsymbol{b}}_2(u,v) \\ \vdots \\ \hat{\boldsymbol{b}}_n(u,v) \end{pmatrix} \tag{3.23}$$

Observe that $\hat{\boldsymbol{m}}_{ij}, \hat{\boldsymbol{c}}_i$, and $\hat{\boldsymbol{b}}_i$ are the Discrete Fourier Transform of $W \times H$ real arrays, therefore, $\hat{\boldsymbol{m}}_{ij}(u, v), \hat{\boldsymbol{c}}_i(u, v), \hat{\boldsymbol{b}}_i(u, v)$ are complex numbers. By solving each (u, v)-linear system, we get the value of the Discrete Fourier Transform of each $\boldsymbol{c}_i$ for the (u, v)-th frequency, this is,

$$\hat{\boldsymbol{c}}_s(u, v) = \Big(\hat{\boldsymbol{m}}(u, v)^{-1}\hat{\boldsymbol{b}}(u, v)\Big)_s \tag{3.24}$$

Once we have solved the linear system for all the (u, v)-frequencies, we find the value of the reconstruction coefficients by applying Inverse Discrete Fourier Transform:

$$\boldsymbol{c} = \mathrm{IDFT}(\hat{\boldsymbol{m}}^{-1}\hat{\boldsymbol{b}}) \tag{3.25}$$

Remarks

Since $\boldsymbol{m}_{ji} = \boldsymbol{m}_{ij}^{\vee}$, we deduce,

$$\hat{\boldsymbol{m}}_{ji}(u, v) = \overline{\hat{\boldsymbol{m}}_{ij}(u, v)} \tag{3.26}$$

This implies that matrices $\hat{\boldsymbol{m}}(u, v)$ are Hermitian. In the next section we will show that each $\hat{\boldsymbol{m}}(u, v)$ is also positive semidefinite.

3.6 System Spectrum

Denote by $\boldsymbol{\omega}_{u,v}$ the (u, v)-eigenvector of the Discrete Fourier Transform. For any $\boldsymbol{F} : \mathbb{Z}_{W \times H} \to \mathbb{R}$ we have,

$$\boldsymbol{F} * \boldsymbol{\omega}_{u,v} = \langle \boldsymbol{F}, \boldsymbol{\omega}_{u,v} \rangle \boldsymbol{\omega}_{u,v} = \hat{\boldsymbol{F}}(u,v)\boldsymbol{\omega}_{u,v} \tag{3.27}$$

Let $\otimes$ denote the Kronecker product operator. For $\boldsymbol{z} = (z_1, \dots, z_n) \in \mathbb{C}^n$ and $\boldsymbol{\omega}_{u,v}$, we have

$$\boldsymbol{z} \otimes \boldsymbol{\omega}_{u,v} = \begin{pmatrix} z_1\boldsymbol{\omega}_{u,v} \\ z_2\boldsymbol{\omega}_{u,v} \\ \vdots \\ z_n\boldsymbol{\omega}_{u,v} \end{pmatrix} \tag{3.28}$$

By convolving the expression above respect to the channel matrix $\boldsymbol{m}$ we obtain

$$\begin{aligned} \boldsymbol{m} * (\boldsymbol{z} \otimes \boldsymbol{\omega}_{u,v}) = \begin{pmatrix} \sum_{s=1}^{n} (\boldsymbol{m}_{1s} * z_s\boldsymbol{\omega}_{u,v}) \\ \sum_{s=1}^{n} (\boldsymbol{m}_{2s} * z_s\boldsymbol{\omega}_{u,v}) \\ \vdots \\ \sum_{s=1}^{n} (\boldsymbol{m}_{ns} * z_s\boldsymbol{\omega}_{u,v}) \end{pmatrix} &= \begin{pmatrix} \sum_{s=1}^{n} (\hat{\boldsymbol{m}}_{1s}(u,v) z_s\boldsymbol{\omega}_{u,v}) \\ \sum_{s=1}^{n} (\hat{\boldsymbol{m}}_{2s}(u,v) z_s\boldsymbol{\omega}_{u,v}) \\ \vdots \\ \sum_{s=1}^{n} (\hat{\boldsymbol{m}}_{ns}(u,v) z_s\boldsymbol{\omega}_{u,v}) \end{pmatrix} \\ &= (\hat{\boldsymbol{m}}(u,v)\boldsymbol{z}) \otimes \boldsymbol{\omega}_{u,v} \end{aligned} \tag{3.29}$$

Let $Z_{u,v} = \{\boldsymbol{z}^1_{u,v}, \dots, \boldsymbol{z}^n_{u,v}\} \subset \mathbb{C}^n$, $\Lambda_{u,v} = \{\lambda^1_{u,v}, \dots, \lambda^n_{u,v}\} \subset \mathbb{C}^n$ be the set of eigenvectors and eigenvalues of $\hat{\boldsymbol{m}}(u,v)$. By definition,

$$\hat{\boldsymbol{m}}(u,v)\boldsymbol{z}^i_{u,v} = \lambda_i \boldsymbol{z}^i_{u,v} \tag{3.30}$$

From expressions 3.29 and 3.30 we get,

$$\boldsymbol{m} * (\boldsymbol{z}^i_{u,v} \otimes \boldsymbol{\omega}_{u,v}) = \lambda_i (\boldsymbol{z}^i_{u,v} \otimes \boldsymbol{\omega}_{u,v}) \tag{3.31}$$

Therefore we conclude that the sets

$$Z = \{\boldsymbol{z}^i_{u,v} \otimes \boldsymbol{\omega}_{u,v} : i \in \{\boldsymbol{n}\}, (u,v) \in \{\boldsymbol{W}\} \times \{\boldsymbol{H}\}\}, \tag{3.32}$$

$$\Lambda = \{\lambda^i_{u,v} : i \in \{\boldsymbol{n}\}, (u,v) \in \{\boldsymbol{W}\} \times \{\boldsymbol{H}\}\} \tag{3.33}$$

are eigenvectors and eigenvalues of $\boldsymbol{m}$.

Since $\Lambda_{u,v} \subset \Lambda \subset \mathbb{R}_{\geq 0}$, we conclude that $\hat{\boldsymbol{m}}(u,v)$ is positive semidefinite.

Remark

The set Z defined above correspond to complex eigenvectors of the linear system. Since the linear system is Hermitian we can find an orthonormal set of real eigenvectors. These real eigenvectors are given by the expressions:

$$\boldsymbol{x}^i_{u,v} = (\boldsymbol{z}^i_{u,v} + \overline{\boldsymbol{z}^i_{u,v}})/2 \qquad \boldsymbol{y}^i_{u,v} = (\boldsymbol{z}^i_{u,v} - \overline{\boldsymbol{z}^i_{u,v}})/2 \tag{3.34}$$

Here, we are counting the real eigenvectors twice, since $\boldsymbol{x}^i_{u,v} = \boldsymbol{y}^i_{W-u,H-v}$ and $\boldsymbol{y}^i_{u,v} = \boldsymbol{x}^i_{W-u,H-v}$.

3.7 Filtering Kernels

To compute the matrix coefficients ($\boldsymbol{m}$) and input vectors ($\boldsymbol{b}$) of our linear system, we are required to sample the subpixels correlations and filtered textures. The coefficients of the input vector are given by

$$\begin{aligned}
\boldsymbol{b} &= [\bar{f} * \bar{\varphi}^{\vee}]^{+} \\
&= [\big(B(A(f) * k_A)\big) * \big(B(A(\varphi) * k_A)\big)^{\vee}]^{+} \\
&= [\big(A(f) * k_A\big) * \big(B^t B(A(\varphi) * k_A)\big)^{\vee}]^{+} \\
&= [A(f) * \big(B^t B(A(\varphi) * k_A) * k_A^{\vee}\big)^{\vee}]^{+} \\
&= [A(f) * \tilde{\varphi}_o^{\vee}]^{+}
\end{aligned} \tag{3.35}$$

The multichannel vector $\tilde{\varphi}_o$, defined by,

$$\tilde{\varphi}_o = B^t B(A(\varphi) * k_A) * k_A^{\vee}, \tag{3.36}$$

will be called the **opponent filtering kernel**. On the other hand, the matrix coefficients are given by

$$\begin{aligned}\boldsymbol{m} &= [\bar{\varphi} * (\bar{\varphi}^{\vee})^{\top}]^{+} \\ &= [\big(B(A(\varphi) * k_A)\big)^{\vee} * \big(B(A(\varphi) * k_A)\big)^{\top}]^{+} \\ &= [\big(B(A(\varphi) * k_A)\big)^{\vee} * \big(B(A(\varphi^{\top}) * k_A)]^{+} \\ &= [\big(B^t B(A(\varphi) * k_A)\big)^{\vee} * \big(A(\varphi^{\top}) * k_A)]^{+} \\ &= [\big(B^t B(A(\varphi) * k_A) * k_A^{\vee}\big)^{\vee} * A(\varphi^{\top})]^{+} \\ &= [\tilde{\varphi}_o^{\vee} * A(\varphi^{\top})]^{+}\end{aligned} \tag{3.37}$$

Equations 3.35 and 3.37 illustrate the importance of $\tilde{\varphi}_o$ in the computation of matrix and input vector coefficents. Equation 3.36 gives an explicit formula to compute this kernel based in the **raw**-pixel, visual model, and metric parameters.

Remark

In the most general case, A^t is not computable. For the computable cases, we define

$$\tilde{\varphi}_r = A^t(B^t B(A(\varphi) * k_A) * k_A^{\vee}) \tag{3.38}$$

and call this expression the **raw filtering kernel**. From this definition we obtain:

$$\boldsymbol{b} = [f * \tilde{\varphi}_r^{\vee}]^{+}, \qquad \boldsymbol{m} = [\tilde{\varphi}_r^{\vee} * (\varphi^{\top})]^{+} \tag{3.39}$$

3.8 Dual Filters

Let $\boldsymbol{\delta}$ denote the discrete impulse, and define the discrete channel matrices

$$\boldsymbol{\Delta}_n = \begin{pmatrix} \boldsymbol{\delta} & 0 & \cdots & 0 \\ 0 & \boldsymbol{\delta} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \boldsymbol{\delta} \end{pmatrix} \tag{3.40}$$

$$\boldsymbol{l} = \begin{pmatrix} \boldsymbol{l}_{1,1} & \boldsymbol{l}_{1,2} & \cdots & \boldsymbol{l}_{1,n} \\ \boldsymbol{l}_{2,1} & \boldsymbol{l}_{2,2} & \cdots & \boldsymbol{l}_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{l}_{n,1} & \boldsymbol{l}_{n,2} & \cdots & \boldsymbol{l}_{n,n} \end{pmatrix} \tag{3.41}$$

to be such that $\boldsymbol{m} * \boldsymbol{l} = \boldsymbol{l} * \boldsymbol{m} = \boldsymbol{\Delta}_n$.

From equations 3.35 and 3.19 we obtain

$$\begin{aligned} \boldsymbol{c} &= \boldsymbol{l} * \boldsymbol{b} \\ &= \boldsymbol{l} * [A(f) * \tilde{\varphi}_o^{\vee}]^+ \\ &= [A(f) * \boldsymbol{l} * \tilde{\varphi}_o^{\vee}]^+ \\ &= [A(f) * (\boldsymbol{l}^{\vee} * \tilde{\varphi}_o)^{\vee}]^+ \\ &= [A(f) * (\mathring{\varphi}_o)^{\vee}]^+ \end{aligned} \tag{3.42}$$

The multichannel vector $\mathring{\varphi}_o$, defined by,

$$\mathring{\varphi}_o = \boldsymbol{l}^{\vee} * \tilde{\varphi}_o, \tag{3.43}$$

will be called the **opponent dual kernel**.

Remarks

In equations 3.42 and 3.43 we overload discrete channel matrix $\boldsymbol{l}$ to denote a multichannel matrix with repeated copies of $\boldsymbol{l}$ in depth.

When A^t is computable, we get

$$\boldsymbol{c} = [f * (\mathring{\varphi}_r)^{\vee}]^+ \tag{3.44}$$

The multichannel vector $\mathring{\varphi}_r$, defined by,

$$\mathring{\varphi}_r = \boldsymbol{l}^{\vee} * \tilde{\varphi}_r \tag{3.45}$$

is called the **raw dual kernel**.

Chapter 4

Singular Systems

4.1 Factors of Singularity

The linear system 3.19, which gives optimal condition, is not always invertible. In such cases, a fixed texture can be reconstructed using distinct sets of coefficients. To understand the reasons of singularity, we will study the concept of uniform reconstruction. Then, we will distinguish two scenarios that produce a poorly conditioned system.

4.1.1 Uniform Reconstruction

Intuitively, we say that a reconstruction is uniform when it is perceptually equal in all its points. If you reconstruct a texture of black and white stripes in the screen, and you start moving away, you will eventually perceive a constant gray texture. At the point a constant gray texture is attained (and beyond), we have a uniform reconstruction. Therefore, uniform reconstruction is a consequence of visual blur.

Let us consider the reconstruction problem in an achromatic space. Suppose we have a display with simple achromatic pixels φ (Figure 4.1a), and visual blur in the achromatic space given by k. Assume both φ and k are normalized, this is, $||\varphi||_1 = ||k||_1 = 1$. If we turn all the pixels with equal intensity, we will have a uniform whenever $(\boldsymbol{c}*\varphi)*k = \boldsymbol{c}*(\varphi*k)$ is constant. This is attained if and only if

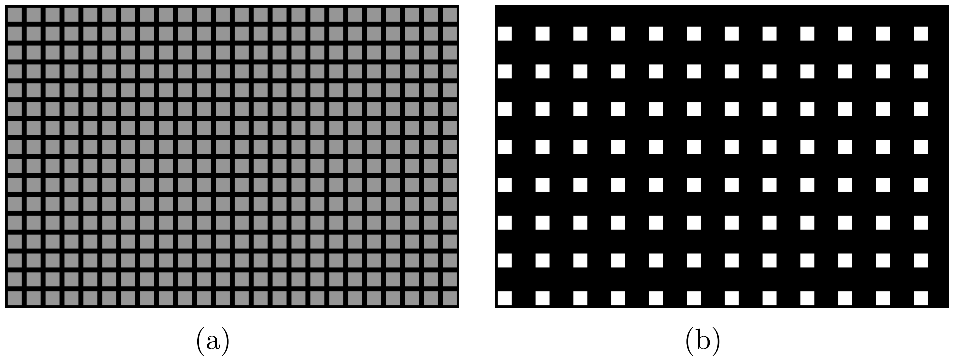
$$\sum_{ij} (\varphi * k)(x+i, y+j) = 1 \qquad \forall (x,y) \in [0,1] \times [0,1] \tag{4.1}$$

The condition above implies $\varphi * k$ satisfies partition of unity (see 9.38). Since φ effective support is only one pixel, in practice, it will not satisfy partition of unity[1]. On the other hand, the visual blurring kernel k will be closer to satisfy partition of unity as far as we move away from the screen. From proposition 9.2, we deduce that partition of unity of k guarantees uniformity in the reconstruction.

Now, consider a reconstruction where only the odd-odd pixels are turned on (Figure 4.1b). In such case, we will have a uniform reconstruction whenever k satisfies 2-step partition of unity. This is

$$\sum_{ij} k(x+2i, y+2j) = \frac{1}{2} \qquad \forall (x,y) \in [0,2] \times [0,2] \tag{4.2}$$

Condition above is equivalent to claim that kernel $k_{1/2}(x) = 2k(2x)$ satisfies partition of unity. If a blurring kernel satisfies partition of unity for certain viewing distance, then, it will satisfies 2-step partition of unity when doubling this distance. In practice, blurring kernels that almost satisfy partition of unity lead to poorly conditioned systems.



(a) (b)

Figure 4.1: Achromatic Display. (a) All pixels at 1/4 of maximum intensity. (b) Odd-Odd pixels at full intensity. Viewing far enough of the screen, both reconstructions may appear identical due to visual blurring.

[1] φ will satisfy partition of unity only if it is a full pixel box kernel.

4.1.2 Color Dependance

In practice, our optimization model represents the light spectrum of each subpixels in a finite dimensional color space. If we have model with n type of subpixels and a color space of dimension m, we will have "color dependance" whenever $n > m$.

Consider a four subpixels RGBW display 1.3a, and suppouse we want to reconstruct a white texture by solving the optimization problem in the three dimensional RGB color space. By projecting the light spectrum of these subpixels to RGB color space we obtain the associations

$$\varphi_R \to \begin{pmatrix}1\\0\\0\end{pmatrix}, \varphi_G \to \begin{pmatrix}0\\1\\0\end{pmatrix}, \varphi_B \to \begin{pmatrix}0\\0\\1\end{pmatrix}, \varphi_W \to \begin{pmatrix}1\\1\\1\end{pmatrix} \tag{4.3}$$

Since the projected light spectrums of $\varphi_W, \varphi_R, \varphi_G, \varphi_B$, form a linear dependent set, we say the subpixels are "color dependent". This isolated condition does not imply singularity of the system. In fact, linear system 3.19 is still invertible for low blurring conditions.

What happens when blurring kernels are large enough to satisfy partition of unity? In that case, we can reconstruct a **uniform** white texture by turning on all RGB subpixels, or by turning on all W subpixels. Both reconstructions are not only perceptually equivalent, but are optimal solutions of the optimization problem.

In conclusion, the linear system 3.19 is singular whenever we have "color dependent" subpixels and blurring kernels satisfying partition of unity.

4.1.3 Blurring Overlapping

Since subpixels of the same type emit identical light spectrum, they are pairwise "color dependent". This dependence harms the system invertibility at large blurring conditions.

Suppose we are interested in reconstructing a $\frac{1}{4}$ intensity red texture in a RGB display. If blurring kernels satisfies two-step partition of unity, we will have optimal reconstruction by turning all red subpixels at $\frac{1}{4}$ intensity, or by turning all odd-odd red subpixels at full intensity, or by turning all even-even red subpixels at full intensity, etc.

No matter if subpixels types are “color independent”, we will have singular systems whenever blurring kernels satisfies two-step partition of unity. Figure 4.1 is an example of this phenomena in an achromatic display.

4.2 Energy Function

4.2.1 Boundedness

As we have previously discussed, systems with large blurring kernels tend to be poorly conditioned and even singular.

In general, a quadratic problem, $F(x) = \frac{1}{2}x^tAx - e^tx$, for A positive semidefinite and singular, is not bounded. If v is an eigenvector of A associated to the eigenvalue 0, and $e^tv < 0$, then v or is direction of infinite descent. This means, $J(\alpha v) \to -\infty$ as $\alpha \to \infty$.

Observe that $e \perp \text{Null}(A)$ is a necesary and sufficient for $F(x)$ to be bounded. This is true for our problem. Our energy function, expressed in convolution form, is given by,

$$J(\boldsymbol{c}) = \langle \boldsymbol{m} * \boldsymbol{c} - \boldsymbol{b}, \boldsymbol{c}\rangle + \langle \bar{f}, \bar{f}\rangle \tag{4.4}$$

Going back in the steps that lead to this equation (see 3.13), we obtain

$$\begin{aligned}
\boldsymbol{c} \in \text{Null}(\boldsymbol{m}) &\iff \boldsymbol{m} * \boldsymbol{c} = 0 \\
&\iff \langle \boldsymbol{m} * \boldsymbol{c}, \boldsymbol{c}\rangle = 0 \\
&\iff \langle \boldsymbol{c}^\top * \bar{\varphi}, \boldsymbol{c}^\top * \bar{\varphi}\rangle = 0 \\
&\iff \langle \omega, \boldsymbol{c}^\top * \bar{\varphi}\rangle = 0 \qquad \forall \omega \in L_2(\Omega_C \times \{\boldsymbol{h}\})
\end{aligned} \tag{4.5}$$

This last condition implies,

$$\begin{aligned}
\langle \bar{f}, \boldsymbol{c}^\top * \bar{\varphi}\rangle = 0 \\
\langle \boldsymbol{b}, \boldsymbol{c}\rangle = 0
\end{aligned} \tag{4.6}$$

We conclude $\boldsymbol{b} \perp \text{Null}(\boldsymbol{m})$. Therefore, energy function 4.4 is indeed bounded.

4.2.2 Range-nullspace Decomposition

Consider our energy function 4.4, expressed in vector form by

$$J(c) = c^t Mc - 2b^t c + d \tag{4.7}$$

Since M is Hermitian, we can express it as $M = QDQ^t$, for Q unitary matrix of eigenvectors, and D diagonal matrix of eigenvalues. Let

$$Q = (Q_R|Q_N), D = \begin{pmatrix} D_R & 0 \\ 0 & 0 \end{pmatrix} \tag{4.8}$$

where Q_R and Q_N are the submatrices of eigenvectors in the range and nullspace of M, respectively. D_R is the diagonal matrix of positive eigenvalues.

By applying the variable change $y = Q^t c$, and writing $y = (y_R, y_N)$, the energy function corresponds to

$$J(c) = y_R^t D_R y_R - 2b^t Q_R y_R + d \tag{4.9}$$

The minimum of the function above is attained for $y_R^* = D_R^{-1} Q_R^t b$. Since 4.9 is independent of y_N, it follows that all vectors $y = (y_R^*, y_N)$ are global minimums. We conclude that the set of optimal solutions to the energy function 4.7 is given by $c = c_R^* + c_N$ where $c_R^* = Q_R y_R^*$ and c_N is any vector in Null(M).

4.3 Minimal Reconstruction Cost Solution

We have already characterized the set of optimal solutions to our energy function 4.7. The question that naturally arise is what solution to choose. The criteria we defined to choose a solution is based in reconstruction costs.

Each subpixel will be assigned a reconstruction cost which is proportional to the square of its intensity. We define the total reconstruction cost by

$$I(c) = c^t D_\gamma c \tag{4.10}$$

where D_γ is a positive diagonal matrix. The values in the diagonal of D_γ are the reconstruction costs associated to each subpixel.

By setting $c_N = Q_N y_N$, we find that the reconstruction cost for all optimal solution to 4.7 is given by,

$$\begin{aligned} I(c) &= (c_R^* + Q_N y_N)^t D_\gamma (c_R^* + Q_N y_N) \\ &= y_N^t (Q_N^t D_\gamma Q_N) y_N + 2 y_N^t (Q_N^t D_\gamma c_R^*) + (c_R^*)^t D_\gamma c_R^* \end{aligned} \tag{4.11}$$

The minimum of this function is attained for

$$y_N^* = -(Q_N^t D_\gamma Q_N)^{-1} (Q_N^t D_\gamma c_R^*) \tag{4.12}$$

By setting $c_N^* = Q_N y_N^*$, we finally get that $c^* = c_R^* + c_N^*$ is the optimal solution with minimal reconstruction cost. From the results above we conclude that

$$\begin{aligned} c^* &= c_R^* + c_N^* \\ &= \big(I - Q_N (Q_N^t D_\gamma Q_N)^{-1} Q_N^t D_\gamma\big) c_R^* \\ &= \big(I - Q_N (Q_N^t D_\gamma Q_N)^{-1} Q_N^t D_\gamma\big)\big(Q_R D_R^{-1} Q_R^t\big) b \end{aligned} \tag{4.13}$$

4.3.1 Linearity of the Minimal Reconstruction Cost Solution

Since b depends linearly on the input (equation 3.21), and c^* depends linearly on b (equation 4.13), we conclude the that the filtering scheme that provide minimal reconstruction cost solution is linear.

If we assume that the reconstruction cost is equal for all the subpixels of the same type, then our optimization problem is also shift invariant (LSI). This results follows from the invariance of the reconstruction cost when the texture is shifted an integer pixel direction.

As a LSI system, this filtering scheme should be written in the form $\boldsymbol{c}_s = \sum_i [f * k_s]^+$ where f is the input texture and k_s is the multichannel filter of the s^{th} subpixel. The analysis of the explicit form of these filters is left for future work. However, we should remark that the computation of these filters can be done in a similar form as done in section 3.8. Let $\boldsymbol{h}_s$ and $\boldsymbol{b}_s$ be discrete channel vectors,

$$\boldsymbol{h}_s = \begin{pmatrix} \boldsymbol{h}_{s1} \\ \vdots \\ \boldsymbol{h}_{ss} \\ \vdots \\ \boldsymbol{h}_{sn} \end{pmatrix}, \qquad \boldsymbol{b}_s = \begin{pmatrix} 0 \\ \vdots \\ \boldsymbol{\delta} \\ \vdots \\ 0 \end{pmatrix} \tag{4.14}$$

such that $\boldsymbol{h}_s$ is the optimal reconstruction coefficients of the minimal cost problem with vector $b = \boldsymbol{b}_s$. Define the channel matrix,

$$\boldsymbol{h} = \begin{pmatrix} \boldsymbol{h}_{1,1} & \boldsymbol{h}_{1,2} & \cdots & \boldsymbol{h}_{1,n} \\ \boldsymbol{h}_{2,1} & \boldsymbol{h}_{2,2} & \cdots & \boldsymbol{h}_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{h}_{n,1} & \boldsymbol{h}_{n,2} & \cdots & \boldsymbol{h}_{n,n} \end{pmatrix} \tag{4.15}$$

Analogous expressions to 3.42 and 3.44 can be formulated for this case. For instance, we may write

$$\boldsymbol{c} = [f * (\mathring{\varphi}_{\gamma r})^{\vee}]^{+} \tag{4.16}$$

where we define,

$$\mathring{\varphi}_{\gamma r} = \boldsymbol{h}^{\vee} * \tilde{\varphi}_r \tag{4.17}$$

The multichannel vector $\mathring{\varphi}_{\gamma r}$ will be called the **minimal cost - raw dual kernel**. This kernel depends in the vector of reconstruction costs γ.

4.3.2 Solution by Spectral Decomposition

Consider a model with n subpixel types, where the reconstruction cost is equal for all the subpixels of the same type. Let γ_s be the reconstruction cost associated to s^{th} subpixel type. Then, the reconstruction cost is given by

$$I(\boldsymbol{c}) = \sum_s \gamma_s ||\boldsymbol{c}_s||^2 \tag{4.18}$$

By Parseval identity, we have,

$$||\boldsymbol{c}_s||^2 = \sum_{(u,v)\in\{\boldsymbol{W}\}\times\{\boldsymbol{H}\}} |\hat{\boldsymbol{c}}_s(u,v)|^2 \tag{4.19}$$

Therefore, reconstruction cost in the frequency domain is given by,

$$I(\boldsymbol{c}) = \sum_{(u,v)\in\{\boldsymbol{W}\}\times\{\boldsymbol{H}\}} \left(\sum_s \gamma_s |\hat{\boldsymbol{c}}_s(u,v)|^2\right) \tag{4.20}$$

On the other hand, optimal solution to the energy function 4.7, is given by conditions,

$$\hat{\boldsymbol{m}}(u,v)\hat{\boldsymbol{c}}(u,v) = \hat{\boldsymbol{b}}(u,v) \qquad \forall (u,v) \in \{\boldsymbol{W}\}\times\{\boldsymbol{H}\} \tag{4.21}$$

From equations 4.20 and 4.21, we deduce that minimal cost solution c^*, is such that $\hat{\boldsymbol{m}}(u,v)\hat{\boldsymbol{c}^*}(u,v) = \hat{\boldsymbol{b}}(u,v)$, and $\sum\limits_s \gamma_s|\hat{\boldsymbol{c}^*}_s(u,v)|^2$ is minimal, for all $(u,v) \in \{\boldsymbol{W}\}\times\{\boldsymbol{H}\}$.

We have transformed the original optimization problem in $W \times H$ smaller subproblems that can be solved in parallel. This approach does not require explicit calculation or storage of matrix Q.

4.3.3 Numerical Issues

Due to the finite arithmetic precision and the numerical methods involved, the computation of eigenvectors and eigenvalues of the optimal linear system 3.19 is not error free. These numerical errors make difficult to do a sharp distinction between zero and non-zero eigenvalues. Therefore, the distinction of null and range spaces may not be done in practice .

We use an energy function to find an approximate solution to the minimal reconstruction cost problem. The new energy function mixes the quality term with the cost term. The cost term is weighted by a scalar factor α,

$$J_\alpha(\boldsymbol{c}) = c^t M c - 2b^t c + d + \alpha c^t D_\gamma c \tag{4.22}$$

As α increases, our system gets better conditioned, but we start loosing quality in favor of cost. Therefore, we must carefully set α to obtain a well conditioned problem with little impact on quality.

The following proposition gives us an insight on the asymptotic behaviour of the energy function J_α as $\alpha \to 0$.

Proposition 4.1. *Let $c^* = c_R^* + c_N^*$ be the optimal-minimal cost solution of equation 4.7. For $\alpha > 0$, let $c^\alpha = c_R^\alpha + c_N^\alpha$ be the optimal solution[2] of equation 4.22. Define $\epsilon_R^\alpha = c_R^* - c_R^\alpha$ and $\epsilon_N^\alpha = c_N^* - c_N^\alpha$. Let m be the minimum positive eigenvalue of M. The following properties hold:*

1. $||\epsilon_R^\alpha|| \leq \alpha(\frac{\max \gamma_s}{m}||c^*||)$.
2. $||\epsilon_N^\alpha|| \leq \frac{\max \gamma_s}{\min \gamma_s}||\epsilon_R^\alpha||$.
3. $\lim_{\alpha \to 0^+} c^\alpha = c^*$.

Remark

The dual filters defined by each system J_α can be computed in analogous way to section 3.8. In this case we must add the reconstruction cost to the subpixel correlation matrix. Using the notation of section 3.8, this corresponds to adding $(\alpha\gamma_s)\boldsymbol{\delta}$ to each $\boldsymbol{m}_{ss}$. For a future work we may study the convergence of the dual filters of J_α to the dual filter of the minimal reconstruction cost problem discussed in section 4.3.

[2] c^α is unique since $M + \alpha D_\gamma$ is positive definite.

Chapter 5

Implementation

In the previous Chapters we presented the formulation and analytic solution to our linear problem. Here, we will discuss some implementation details.

5.1 Function Representations

Textures, visual kernels, and subpixels, are all represented as functions on continuous domains. We will distinguish two type of function representations:

- Exact: Those are functions associated to a formula or an algorithm which provide their exact value at any point.

- Sampled: Those are functions whose exact values we know just in a finite set of samples. When the samples are located in a regular grid, we will call this function an image. From the set of samples S, we extend the function to the rest of the domain by using a reconstruction kernel ψ. Therefore, we get a function approximation given by $S * \psi$. In general ψ is taking interpolative (i.e.,$[\psi] = \delta$), in order to preserve the texture values at the sampled points.

5.2 Filtering Kernels Construction

In order to calculate filtering kernels $\tilde{\varphi}$ of equations 3.36 and 3.38, it is required to compute a sequence of convolutions between the subpixel repre-

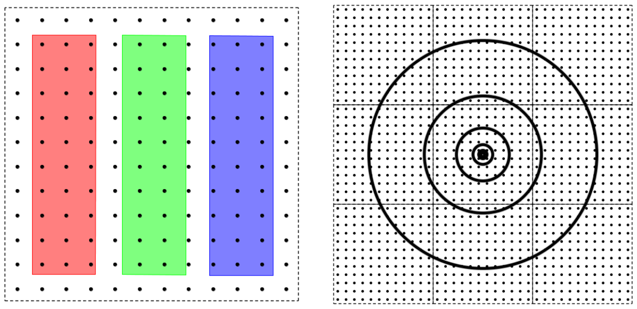

(a) Pixel (b) Kernel

Figure 5.1: Sampling Scheme.

sentation φ and the visual blurring kernels k_A. To compute this convolution, we sample regularly the subpixel geometry and the visual kernel impulse response[1] (Figure 5.1). We associate to each subpixel and visual kernel a continuous representation, based in their samples (S_φ and S_{k_A}) and a common reconstruction kernel ψ:

$$\varphi = S_\varphi * \psi \qquad k_A = S_{k_A} * \psi \tag{5.1}$$

From this representation we can easily compute convolutions by dividing in discrete and continuous parts. For instance, $\varphi * k_A * k_A^\vee = (S_\varphi * S_{k_A} * S_{k_A^\vee}) * (\psi * \psi * \psi^\vee)$. The previous expression is similar to the filtering kernels expressions (3.36,3.38) up to the linear transformations involved in our model.

The discrete convolution between the samples can be quickly computed using FFT. On the other hand, we use simple interpolation kernels for reconstruction. By taking $\psi = b_1$, the bspline of order 1 (i.e., a hat), then, the final reconstruction kernel given by expression $\psi * \psi * \psi^\vee$ is bspline of order 5.

Since a bspline of order 5 is a piecewise polynomial function of large degree and support (5th degree and support 6), it is relatively expensive to evaluate the filtering kernel $\tilde{\varphi}$ using this representation. In practice, we sample densely the subpixel geometry and visual kernels (300×300 samples per pixel for S_φ and S_{k_A}) and we use a hat as final reconstruction kernel. The large sampling density amortizes the error produced by the simplification of

[1]In practice our sampling grid is 300×300 samples per pixel.

the final reconstruction kernel.

5.3 Coefficient Computations

Subpixel correlation coefficients ($\boldsymbol{m}$) and filtered texture coefficients ($\boldsymbol{b}$) are computed by Monte Carlo Integration. Observe that expressions in 3.35 and 3.37 do not require an explicit convolution between its terms, but integration at sampling positions. Monte Carlo approximate an integral by summing over a set of samples:

$$\int_a^b f(x)dx \approx \frac{b-a}{n} \sum_{i=1}^{n} f(x_i) \tag{5.2}$$

Here, x_i's are variates uniformly distributed in $[a, b]$. In our case, the integration functions are $A(\varphi^\top)\tilde{\varphi}(\cdot - \boldsymbol{i})$ for subpixel correlation , and $A(f)\tilde{\varphi}(\cdot - \boldsymbol{i})$ for filtered texture.

We can use a sample, not just for integration in a single pixel position, but in a neighborhood of the pixel grid. Using a sample repeatedly improve our implementation performance by reducing kernels and texture evaluations.

For instance, consider subpixel correlations $[\tilde{\varphi}_o^\vee * A(\varphi^\top)]^+$. A sample of the color transformed subpixel, $A(\varphi)$, can be used repeatedly for Monte Carlo integration at each pixel center. On the other hand, samples of the filtering kernel, $\tilde{\varphi}_o$, can only be used for Monte Carlo integration at its closest pixel center.

Consider the filtered texture, $[\tilde{\varphi}_o^\vee * A(f)]^+$. Samples of the filtering kernel, $\tilde{\varphi}_o$, can now be repeatedly used for integration at each pixel center. On the other hand, samples of the color transformed texture, $A(f)$, can be repeatedly used for integration in a neighbourhood of pixels covered by the filtering kernel support.

This last strategy of reusing texture samples is called "sample sharing", and it is very effective for filtering with large support kernels (as is our case). "Sample sharing" may be memory intensive. In such cases, a rolling buffer can be adapted to reduce memory requirements. By using a rolling buffer, we do not require storage for samples of the complete image, but for samples in the lines covered by the filtering kernel support.

5.4 System Solver

In order to solve each linear system $\hat{A}(u,v)$ we take advantage of the Hermiticty. We factorize and solve the Hermitian system using a variant of Cholesky decomposition. The traditional Cholesky decomposition of a positive semidefinite Hermitian operator M, is given by $M = LL^*$ where L is lower triangular matrix. The variant we implemented factorises $M = LDL^*$, where L is lower triangular with 1's diagonal, and D is real diagonal matrix. Factorization LDL^* can be applied to general hermtian systems, i.e., the positivity condition is not necessary. This is worth for ill-conditioned problems, where the rounding errors could introduce negative eigenvalues to the system.

We preferred LDL^* since it avoids square roots calculations on the diagonal elements of the matrix. On other aspects, LL^* and LDL^* factorizations are quite similar. Both have equal memory demand, and use equal amount of float operations during the variables substitution phase.

Figure 5.2 shows performance of the parallel LDL^* Cholesky solver implemented in GPU. To compute the optimal reconstruction of a 1024×1024 image in a RGB striped display, it is required to solve 1048576 Hermitian linear systems of size 3×3. In a GTX 560 this was done in approximately 1.28 miliseconds.

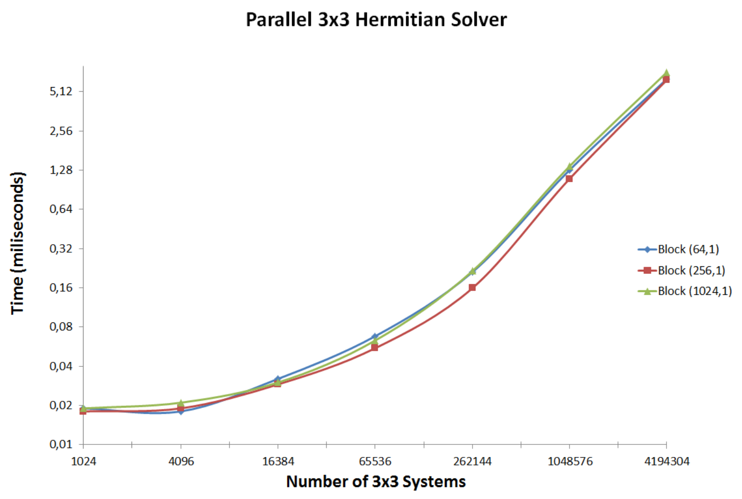


Figure 5.2: GPU performance.

Chapter 6

Results

6.1 Variations

In this section we present some experiments showing optimal reconstruction for some set of parameters. The objective of each experiment is to identify the impact of certain parameter in the optimization model. This is done by varying certain parameter while the others remain fixed. We present a brief result discussion at the end of this section. The following are the experiments considered:

Experiment 1: Subpixel Variations

- Pixel: Striped RGB.
- Opponent Color Transform: RGB to SCIELAB
- Metric Transform: Uniform scaling[1].
- Blurring Kernel: SCIELAB luminance + bspline3 + bspline4.[2]
- **Description**: We compare two variations of subpixel geometric scale and their optimal reconstruction kernels.

[1]Uniform scaling maps (1 1 1) in RGB to (1 1 1) in $O_1O_2O_3$. This ensures that luminance and chrominance have equal weight in the energy function.

[2]SCIELAB blurring kernels for chrominance channels decreases slowly (see 6.5). In these experiments we use bsplines as the blurring kernels for chrominance channels in our optimization model.

Experiment 2: Metric Variations

- Pixel: Striped RGB.
- Opponent Color Transform: RGB to SCIELAB.
- Blurring Kernel: SCIELAB luminance + bspline3 + bspline4.
- **Description**: We compare two variations of the transformation from the opponent color space to the metric space. One of the transformations is a uniform scaling, and the other a non-uniform scaling[3].

Experiment 3: Blurring Kernel Variations

- Pixel: Striped RGB.
- Opponent Color Transform: RGB to SCIELAB
- Metric Transform: Uniform scaling.
- **Description**: We compare two set of scaled blurring kernels which lead to optimal reconstruction kernels for observers at different distances to the screen. The blurring kernels are a 0.7-scaled and a 1.15-scaled (SCIELAB luminance + bspline3 + bspline4) kernels.

Experiment 4: Reconstruction Cost Variations

- Pixel: Ricoh RGBW.
- Opponent Color Transform: RGB to SCIELAB
- Metric Transform: Uniform scaling.
- Blurring Kernel: bspline3 + bspline5 + bspline7.
- **Description**: We compare optimal reconstruction kernels for multichromatic pixel grids with distinct subpixel costs. The cost vectors we evaluate are $(0, 0, 0, 0.1), (0.05, 0.05, 0.05, 0.05)$ and $(0.1, 0.1, 0.1, 0.0)$. The coordinates of these vectors give the cost value of R, G, B and W subpixels.

[3]Non-uniform scaling maps (1 1 1) in RGB to (10 1 1) in $O_1O_2O_3$. Luminance term is predominant. This may impairs the conditioning of the linear system.

The simulation of visual reconstruction is generated using SCIELAB kernels. For all the experiments we assume that 1 visual degree covers 40 pixels of the display. Dual filters are computed as indicated by 3.45. Dual filters are 2-dimensional scalar functions, for comparison and clear visualization we just plot their central horizontal slide.

6.1.1 Experiment 1: Subpixel Variations

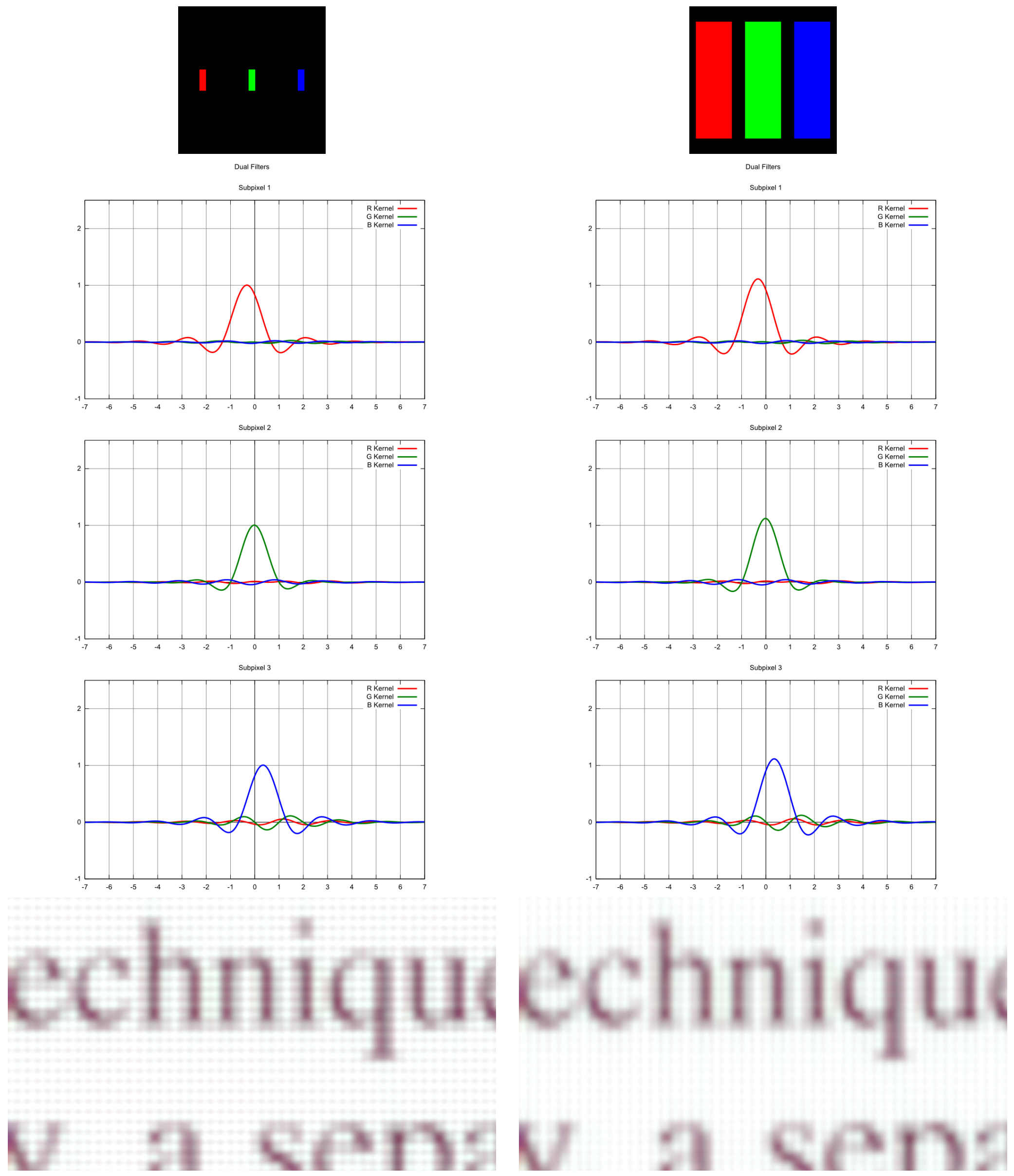
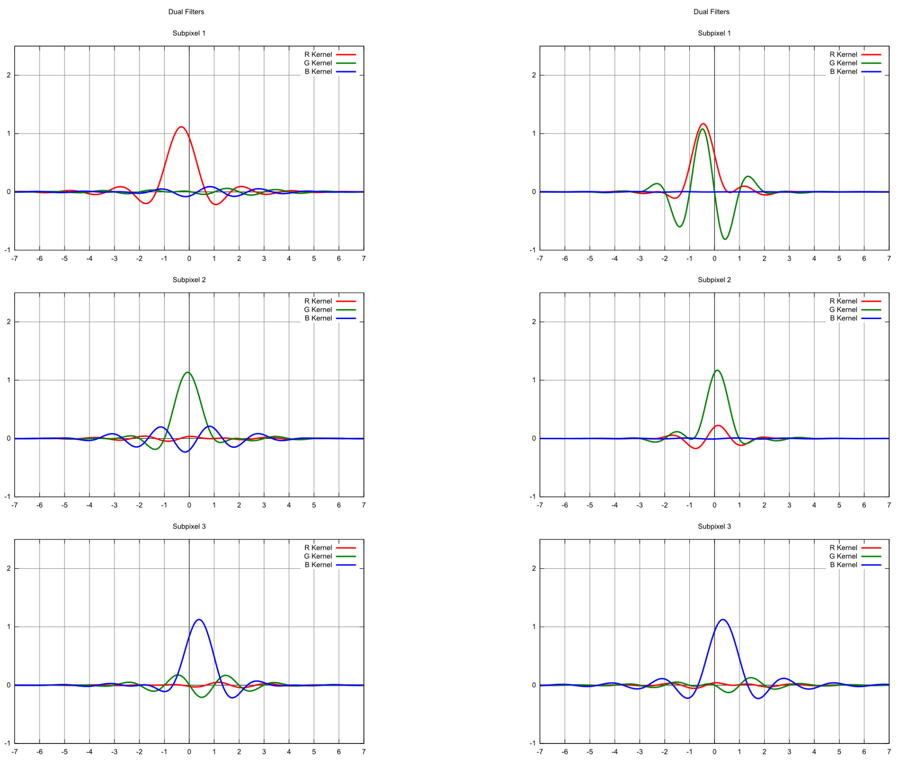


Figure 6.1: Subpixel kernels (top), dual filters (middle) and visual reconstruction (bottom).

6.1.2 Experiment 2: Metric Variations

$$\begin{pmatrix} 0.306 & 0.693 & 0.001 \\ 2.313 & -1.062 & -0.251 \\ -2.143 & -8.846 & 11.99 \end{pmatrix} \qquad \begin{pmatrix} 3.059 & 6.928 & 0.013 \\ 2.313 & -1.062 & -0.251 \\ -2.143 & -8.846 & 11.99 \end{pmatrix}$$



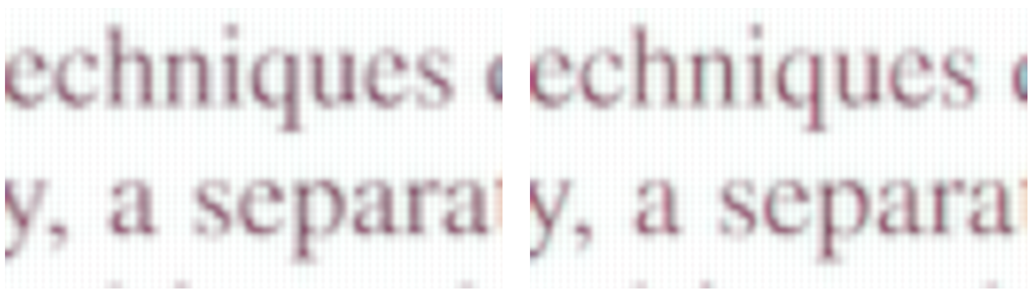


Figure 6.2: Scaled opponent color transformations (top), dual filters (middle) and visual reconstruction (bottom).

6.1.3 Experiment 3: Blurring Kernel Variations

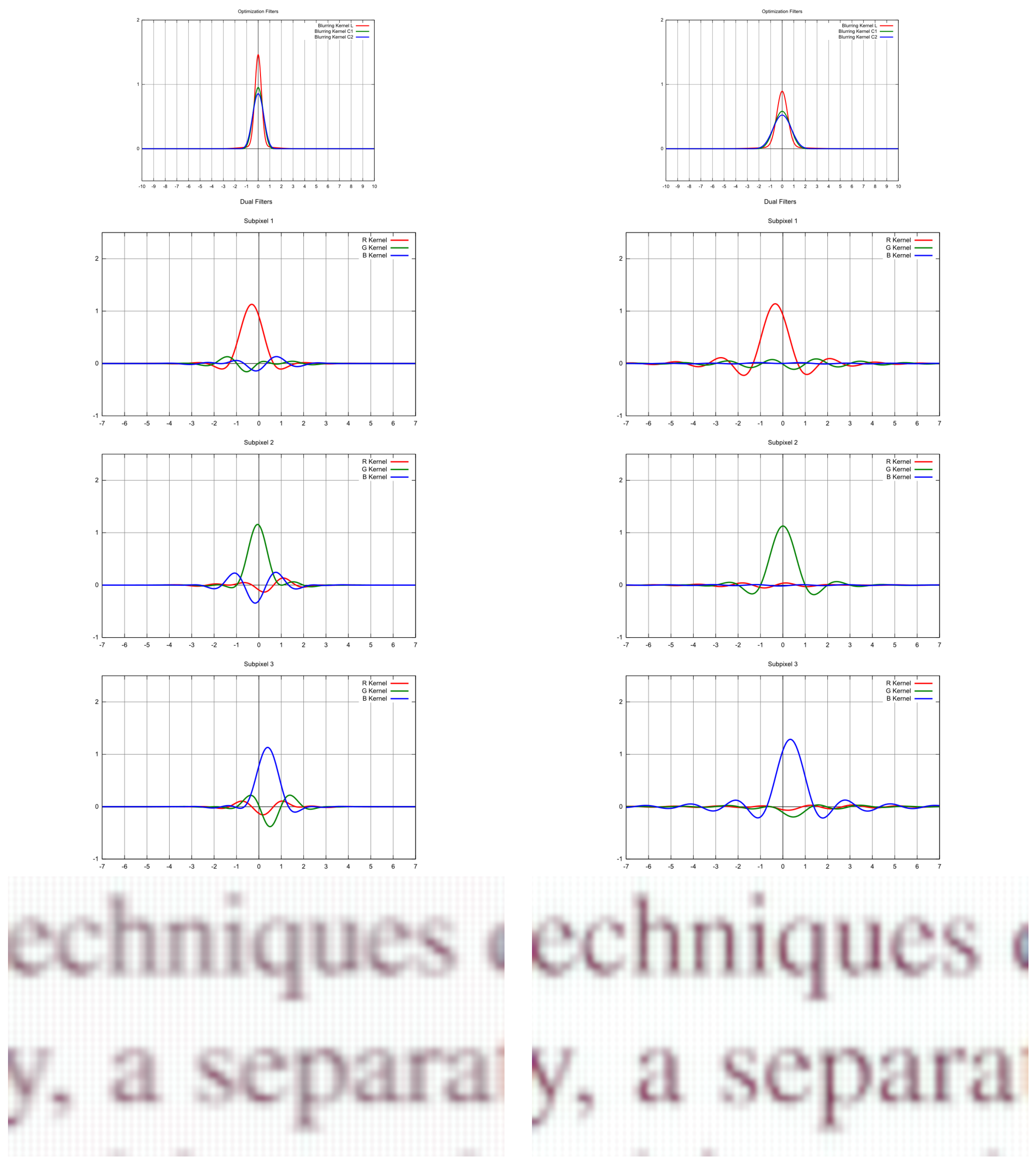


Figure 6.3: Blurring kernels (top), dual filters (middle) and visual reconstruction (bottom).

6.1.4 Experiment 4: Reconstruction Cost Variations

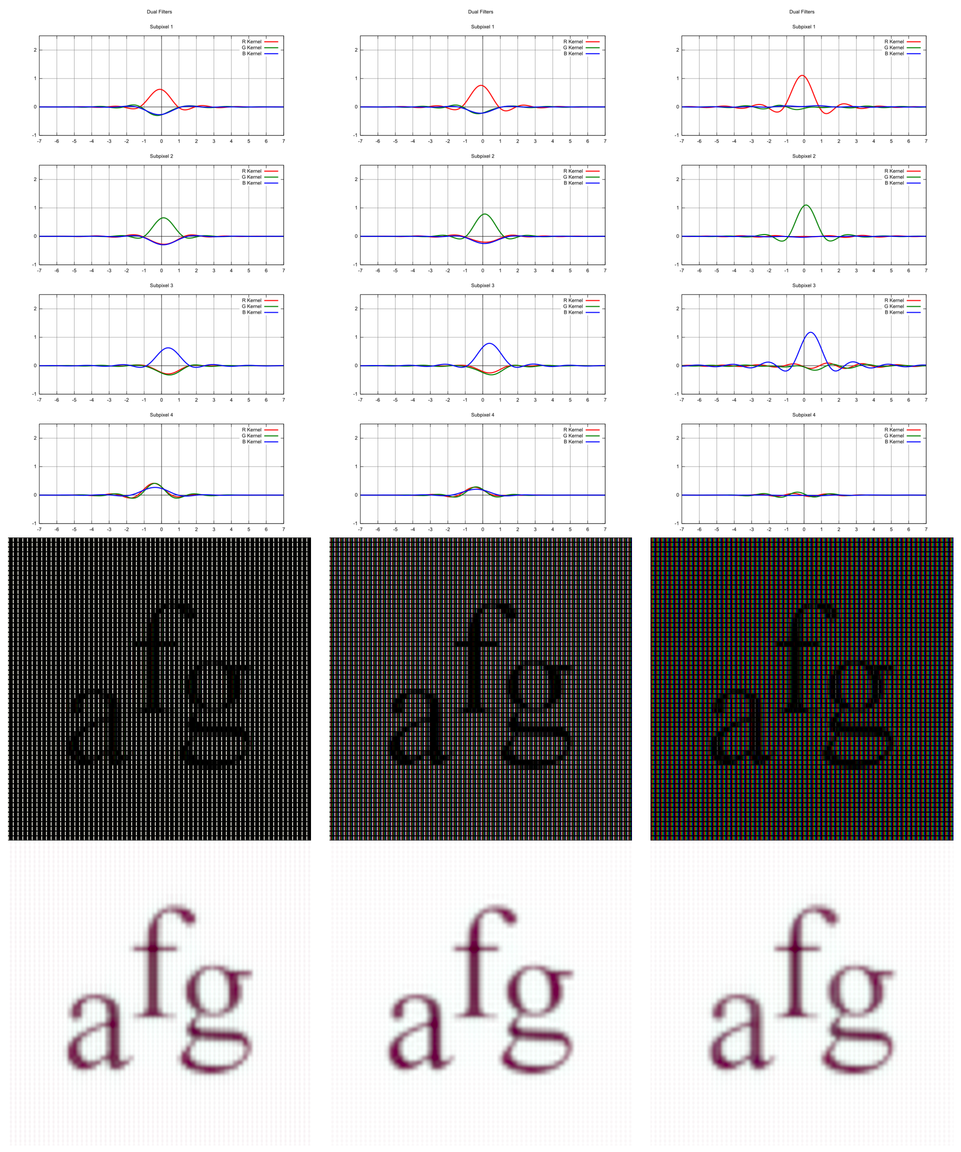


Figure 6.4: Dual filters(top), raw reconstruction (middle), and visual reconstruction (bottom).

6.1.5 Result Discussion

The filter plots in the previous experiments give us insight on how the optimal dual kernels change due to parameter variation. In general, these filter plots resemblance Platt [12] results: for any subpixel, the kernel of its respective color is an interpolatory low pass filtering; instead, kernels for the crossed colors are band pass (or null) filters.

We should notice that a relative variation of the blurring kernel has a larger effect on the overall system than a relative variation of the pixel reconstruction kernel. This is an expected result since the blurring kernel support is in general much larger than the pixel reconstruction kernel[4]. The variation of pixel size produced a slight change on the optimal kernels and a noticeable change in the simulated visual reconstruction. As expected, larger subpixels provide a more uniform reconstruction of a flat white texture.

In the blurring kernel case, the variations on scale produced notorious changes in optimal kernels and simulated visual reconstructions. Stretch blurring kernels does not preserve zero frequency, implying that a constant texture (for instance a white background texture) is not reproduced with the expected intensity. The larger blurring kernels are closer to satisfy partition of unity and this condition immediately guarantees zero frequency invariance.

The experiment on metric variations shows the strong impact of this parameter on the model. For the scenario with an unbalanced metric, we notice that the dual filters present some odd behaviours (high oscillations). This is may be due to the poor conditioning of these systems.

Finally we observe very interesting results on the reconstruction costs test. As expected, adding a relative high cost to certain subpixel inhibits its activation. In the case of subpixels R, G, and B being highly penalized, we observe how the dual filter associated to the supixel color (which is a positive low pass filter) is cancelled by the other colors filter (i.e., the W pixel in the RGBW case). From the simulated visual reconstructions, we can also confirm that a larger resolution is attained by allowing activation of the largest amount of subpixels.

[4]For instance, a blurring kernel with effective support of one visual degree may cover 30 pixels for an observer at 50 cm from a regular LCD display.

6.2 Pixel Grids

In this section we compare reconstruction in several pixel grids. The common parameters are the following:

- Opponent Color Transform: RGB to SCIELAB
- Metric Transform: Uniform scaling.

The blurring kernels are given by the SCIELAB model for both luminance and chrominance channels. According to the relation between blurring kernels and pixel scale we divided in two set of pixels grids:

- Set 1: Striped RGB, Iphone, Galaxy Note, Quattron and Ricoh.
- Set 2: PenTile-RGBW, Nexus and Galaxy S4.

For pixel grids in set 1, a visual degree covers 40 pixels. For pixel grids in set 2, a visual degree covers 20 pixels. The blurring kernels for set 1 and 2 are shown in Figure 6.5.

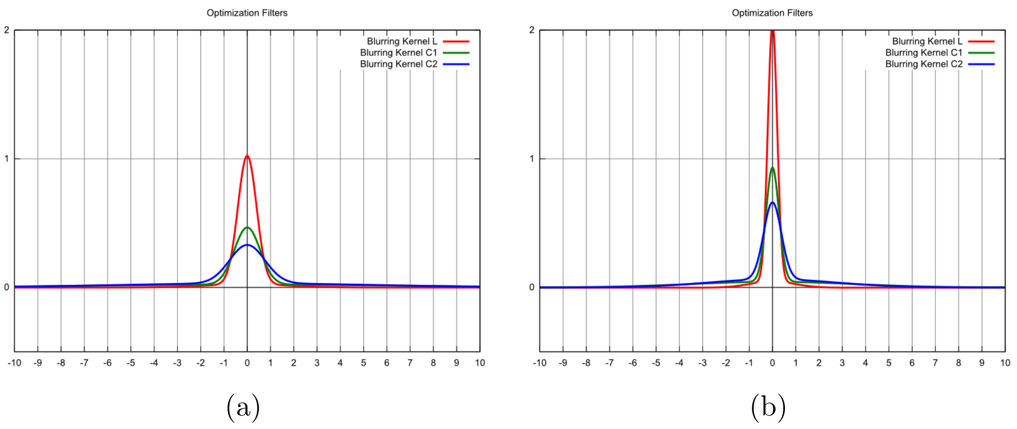


Figure 6.5: SCIELAB blurring kernels scaled for (a) set 1 and (b) set 2.

Simulation of visual reconstruction is also generated using SCIELAB kernels. They are accordingly scaled for pixel grids in set 1 and set 2. Dual filters are computed as indicated by 3.45. For visualization clearness we just plot the central horizontal slide of the dual filters. Result discussion is presented in the end of the section.

6.2.1 Striped RGB

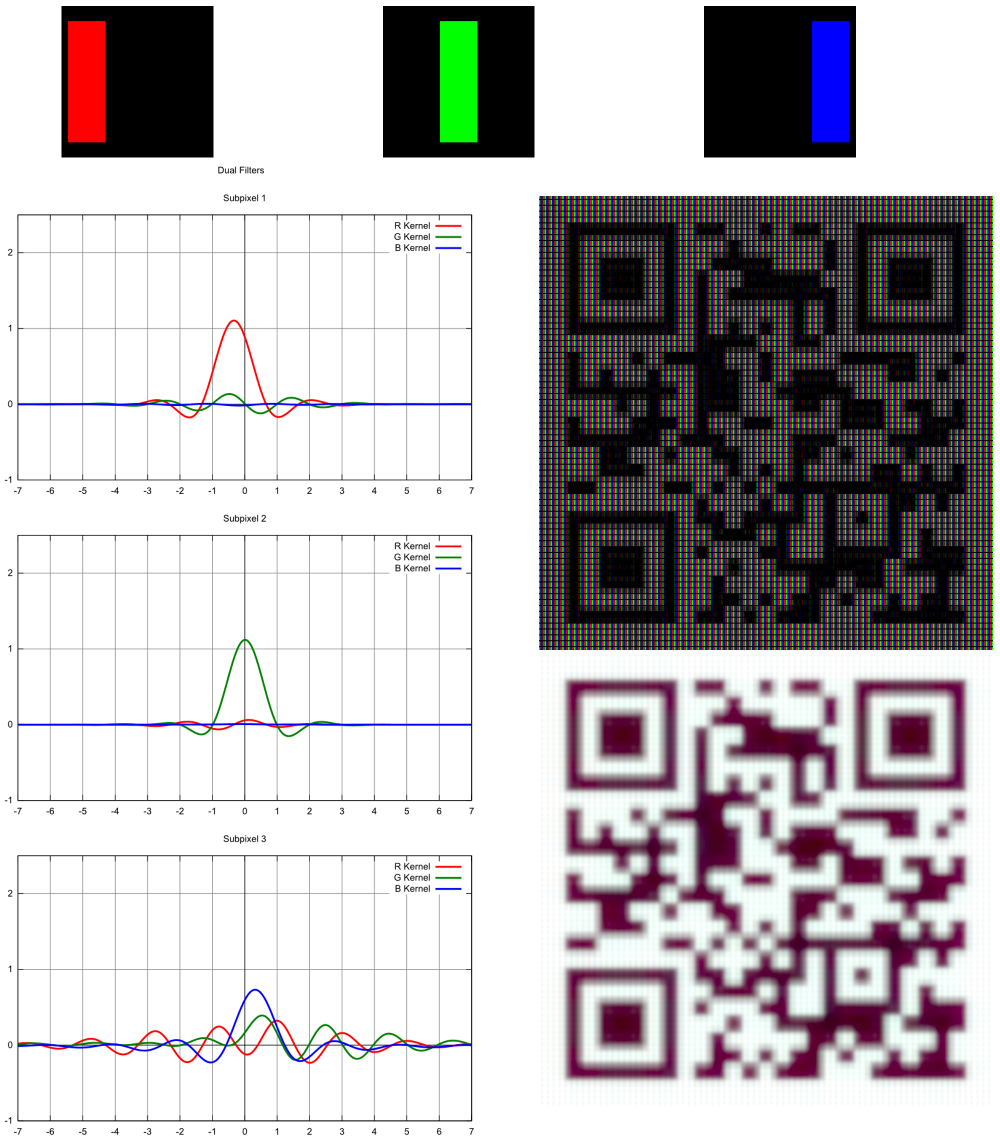


Figure 6.6: Subpixel kernels (top), dual filters (bottom left), raw reconstruction (middle right), and visual reconstruction (bottom right).

6.2.2 Iphone

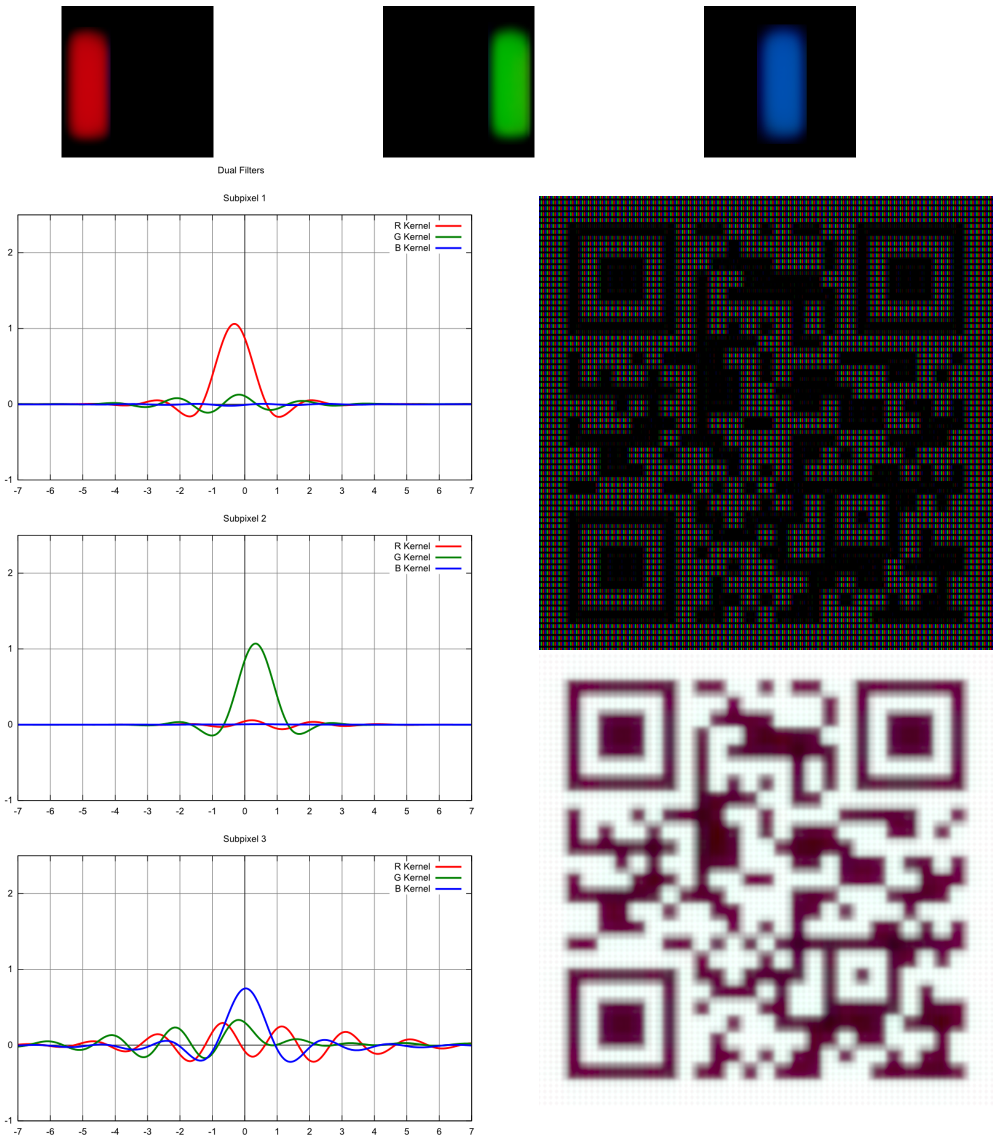


Figure 6.7: Subpixel kernels (top), dual filters (bottom left), raw reconstruction (middle right), and visual reconstruction (bottom right).

6.2.3 Galaxy Note

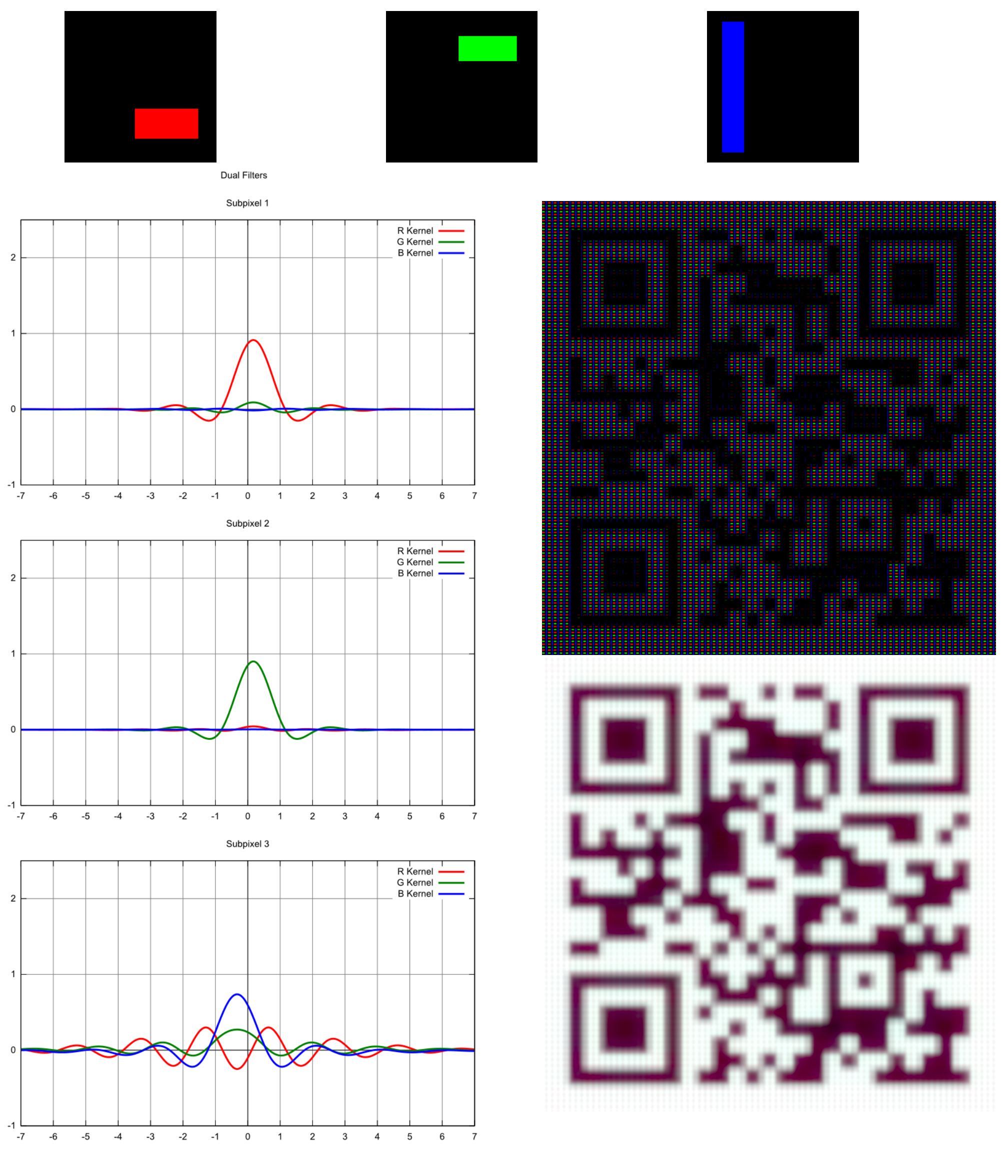


Figure 6.8: Subpixel kernels (top), dual filters (bottom left), raw reconstruction (middle right), and visual reconstruction (bottom right).

6.2.4 Quattron

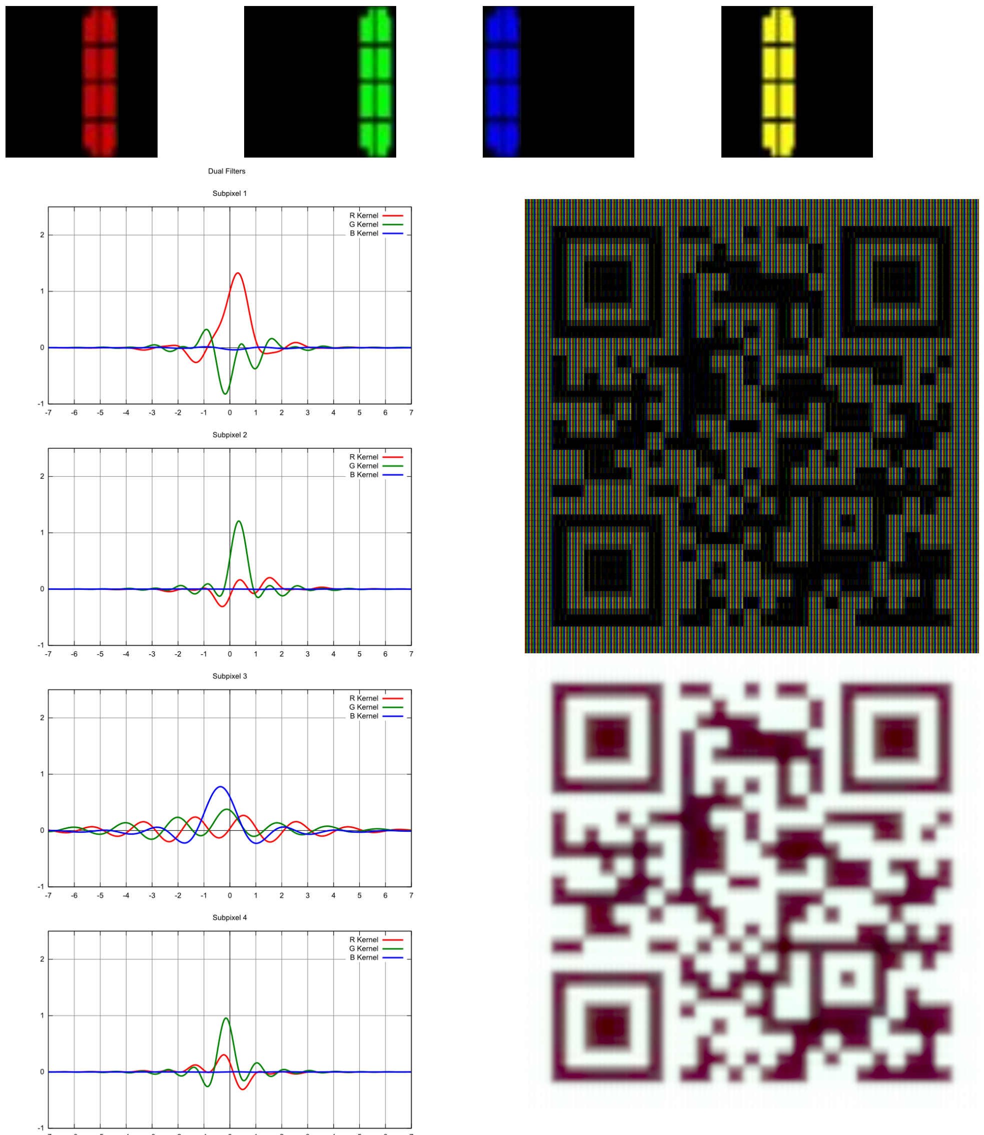


Figure 6.9: Subpixel kernels (top), dual filters (bottom left), raw reconstruction (middle right), and visual reconstruction (bottom right).

6.2.5 Ricoh

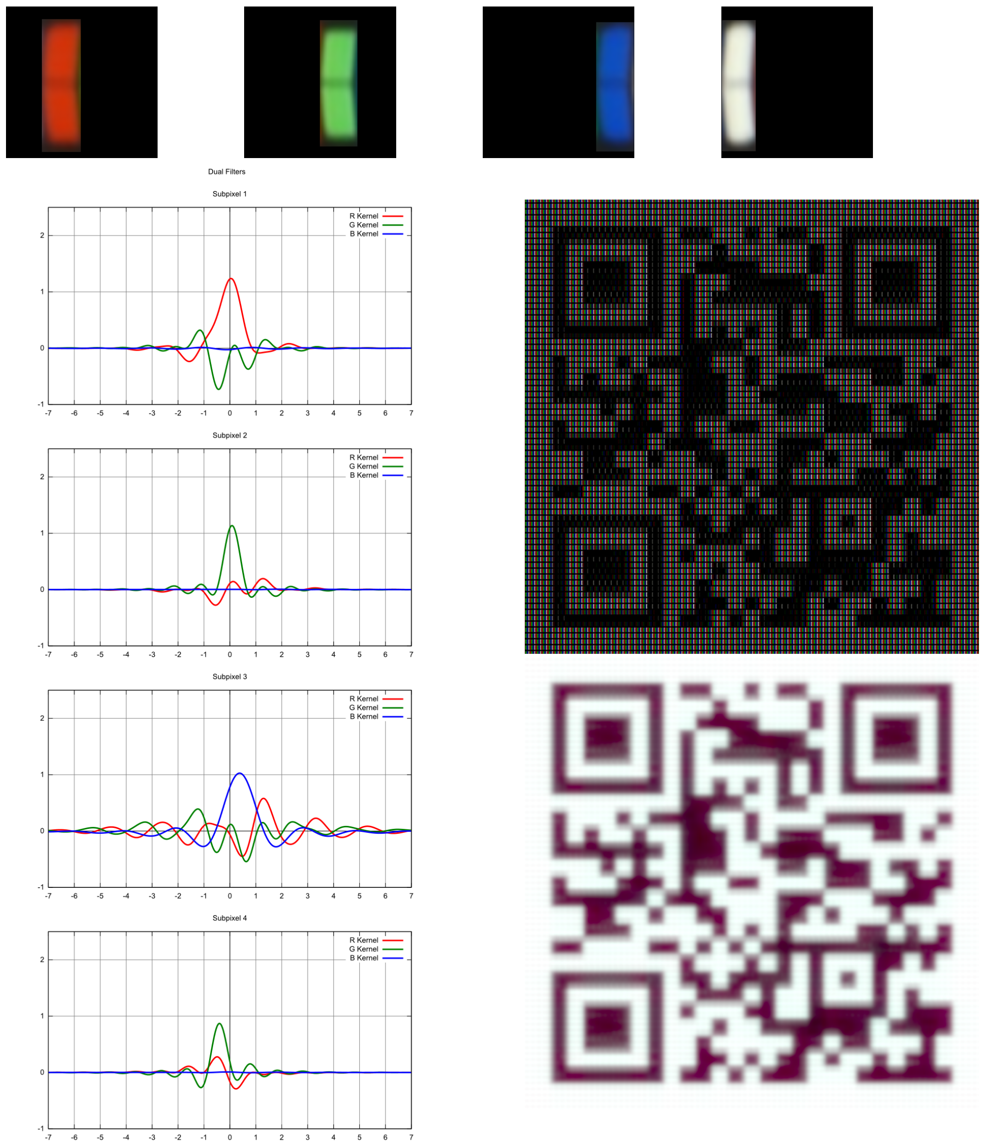
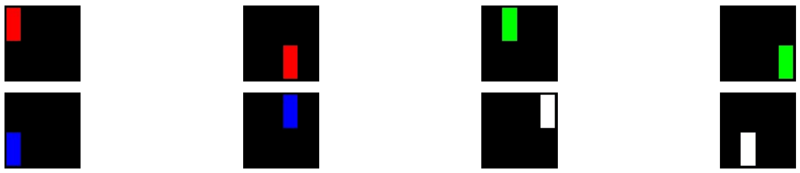
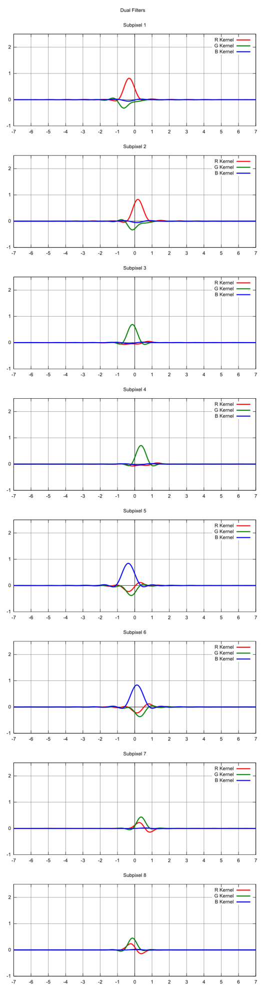
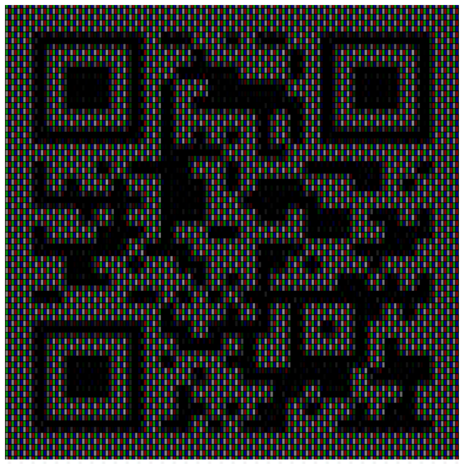


Figure 6.10: Subpixel kernels (top), dual filters (bottom left), raw reconstruction (middle right), and visual reconstruction (bottom right).

6.2.6 Pentile-RGBW







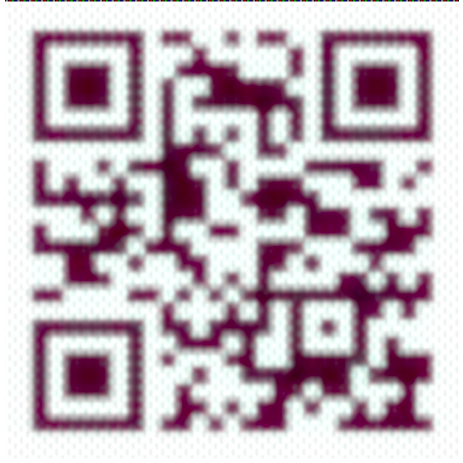


Figure 6.11: Subpixel kernels (top), dual filters (bottom left), raw reconstruction (middle right), and visual reconstruction (bottom right).

6.2.7 Nexus

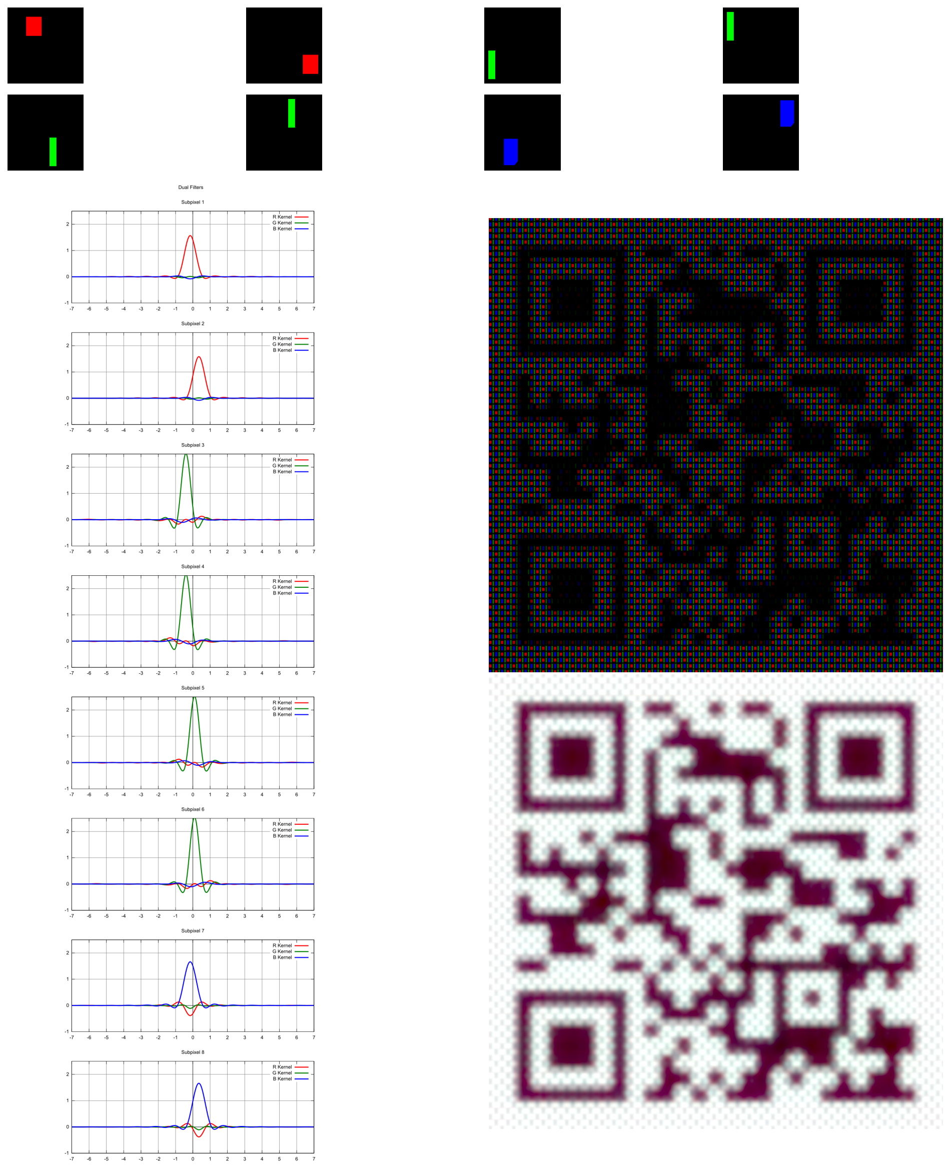


Figure 6.12: Subpixel kernels (top), dual filters (bottom left), raw reconstruction (middle right), and visual reconstruction (bottom right).

6.2.8 Galaxy S4

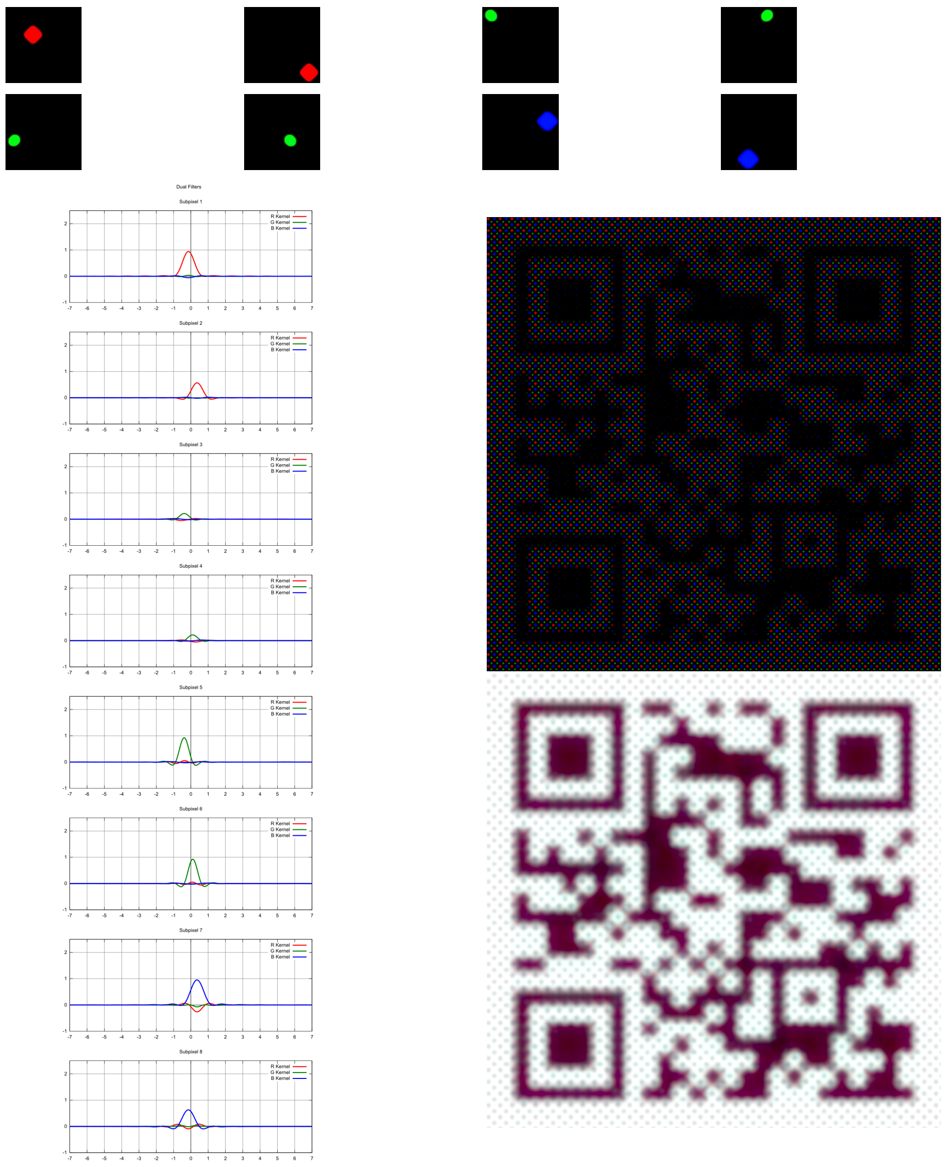


Figure 6.13: Subpixel kernels (top), dual filters (bottom left), raw reconstruction (middle right), and visual reconstruction (bottom right).

6.2.9 Results Discussion

In this section we have presented optimal linear filters for different pixel grids. Some general observations can be drawn from the previous results. First, we can check that for any R, G and B subpixel, the kernel in its respective channel is still an interpolative low pass filter centered at subpixel position. This property was previously stated in the context of RGB stripe displays, and now we observe that it also hold in the remain pixel grids.

We also get some ringing patterns in some of the reconstructions, specially in multichromatic pixel grids. This is probably a consequence of poor conditioned systems due to color dependance. The previous results were obtained without a reconstruction cost terms. As shown in Section , more reliable filter set are obtained by adding a reconstruction cost term.

In the case of the striped RGB pixel,the set of dual filters resembles the results obtained by Platt (see Figure 2.2). Crossed filters for red and green subpixels are almost null, instead, crossed filters for the blue subpixel are highly oscillatory. Similar kernels were obtained for the Galaxy Note pixel, up to its respective shifts to subpixel positions.

In the case of the G subpixels of Galaxy Nexus and Galaxy S4, the period of interpolation is not a complete pixel cycle but half of it. This is coherent with the pattern of these two pixel grids: the green channel is horizontally sampled and reconstructed twice per pixel. From the simulated reconstructions of the Galaxy Nexus and Galaxy S4 we observe that the RGBG pattern almost attain the same level of resolution than a RGB pixel, but reducing the number of subpixels by a third. This supports the efficiency claims regarding this kind of grid patterns.

Interestingly, the yellow subpixel of Quattron, and the white subpixels in the Ricoh and PenTile-RGBW grids, have active kernels only for the red and green channels. This is coherent to our visual model where green and red components are predominant in the luminance channel.

Chapter 7

Conclusions and Future Work

The main contribution of this work is a continuous domain formulation of subpixel filtering that overcomes some limitations of discrete approaches. The main gain of the continuous formulation is a flexible and precise representation of the model parameters: subpixel geometry, light spectrum, metric color spaces, and visual blurring. The continuous formulation provides an appropiate scenario to accurately identify the effect of these parameters on the optimal kernel.

Another major contribution of this work is an strategy for simulating human visual response to an image reconstructed at certain display. This technique, despite its simplicity, was not explored (to our known) on previous works in the area. Previous approaches evaluate reconstruction quality from metric models that hardly captures (or identify) the visual properties incident on the overall result. Instead, our visual simulation approach allows a clear capture of fundamental properties such as sharpness and color fringe. Therefore, our visual simulation could guide the design of new pixel patterns and its respective filters.

From our experimental results we observe a larger dependence of the model on visual blurring kernel than on subpixel reconstruction kernels. This result is coherent with the difference of scale between these kind of kernels. Subpixel kernels determine the central position and interpolatory period of the optimal filtering kernels. Blurring kernels influence the oscilation amplitude of the optimal filtering kernels, and are the main reason of poor conditioning of the optimal linear system.

In this work, we were only concerned with tridimensional color metric spaces (coherently to thrichromatic vision), however, our model formulation and analysis is valid for color spaces of arbitrary dimension. This allows a direct integration with more ellaborated human visual system models, input signals with larger gamut, and multichromatic subpixel grids.

The color metric spaces and subpixels chroma also have a large impact on the optimal kernels and system conditioning. In the case of trichromatic grids (RGB), a satisfactory reconstruction was obtained for color metric spaces with uniformly weighted components. Instead, giving considerably more weight to one component (say luminance) introduces color artefacts and is a source of poor conditioning. In the case of multichromatic grids (e.g. RGBW, RGBY) system singularity may appear on large blurring conditions. This is an expected consequence of the linear dependence (in a vector space sense) of the subpixels chromatic values.

The system singularities introduced by the blurring conditions and color spaces were resolved using a two step optimization procedure. Our approach introduces a cost associated to the reconstruction, and finds the reconstruction with minimal cost that attains the maximum quality. This reconstruction cost can be naturally identified with the energy consumption of the device. Our current implementation is based in a unified energy function that is numerical robust and provides an approximation to the optimal kernel.

7.1 Future Work

Due to the generality of the approached problem, and flexibility of our current model there are several aspects that remain to be explored. Also, we have some considerations in the formulation and implementation of our current approach that may lead to more robust results.

First, our current computation of filtering kernels and optimal linear system coefficients are done using a regular sampling basis. Despite the apparent large density of the initial sampling[1] the simple numerical methods we used for convolution and integration may not provide enough accuracy. The

[1] 300×300 samples per pixel for the blurring and subpixel kernels.

regular sampling approach is sufficient to easily model distinct subpixel geometries, however, it seems unappropiate for sampling blurring kernels with large variance. For a future work we propose doing computation in analytic (i.e., closed) form for both subpixel reconstruction kernels and visual blurring kernels. This may be done, for instance, using a gaussian basis representation for our set of kernels.

A second topic of interest is on improving performance of filtering strategies; this is currently the bottleneck of our approach. Since the filtering kernels inherit the large variance of blurring kernels, accurate integration becomes a computationally expensive task. We have explored texture samples reuse by overalapping filters, and CPU parallelization implementations that improved performance. For a future work, we may explore variance reduction sampling strategies (biased sampling), filtering decomposition (e.g., separable filters), and GPU parallelization.

Our third concern is on exploring chromatic models with an augmented number of components. The results on this work follow a trichromatic model of human vision. This is in accordance to the rough classification of photoreceptor cells (cones) in the eye. However, our formulation does not constrain the number of chromatic components. In a future work we may explore models with several chromatic components. This kind of models could provide a more accurate description of the HVS light perception, and the reproducible gamut by certain display. Using chromatic models with several components may also improve the conditioning of the optimal linear systems.

Providing a quantitative measure of the real resolution and gamut of a pixel grid is also an interesting topic to further explore. Currently, our model reproduce the visual stimulation produced by different pixel grids in terms of perceived resolution and color. In a future work we may try to classifying and quantitavely assess this stimulation in order to measure pixel grid capabilities. This may require of supervised learning models trainned from user experiences. The ability of quantitatively measuring pixel grid capabilities is fundamental to guide pixel design.

Finally, for a future work we may consider the nonlinear behavior of human light perception. The visual model followed in this work is a linear simplification that allows efficient computation. However, it does not capture the metric distortions produced by nonlinear luminance and chromatic responses. This situation was initially identified by Kajiya and Ullner [7]. A nonlinear approach is computationally intensive but could lead to notorius improvement on the results quality.

Chapter 8

Theoretical Background

In this chapter we introduce some of the basic principles on light perception by the human visual system. We present some models for chromatic representation, some of them based in physiological attributes, and other on computational convenience. Finally, we provide brief description of the foundations of standard filtering strategies on achromatic and trichromatic pixel grids.

8.1 Human Visual System

8.1.1 Visible Light

The visible light is a band of the electromagnetic spectrum covering wavelengths from 300nm to 700nm. Wavelengths less than 400nm are called ultraviolets, and those greater than 700nm are called infra-red. Both ultraviolet and infrared are imperceptible electromagnetic radiation for the human visual system.

The spectral distribution of a light source indicates the energy it transports for each wavelength. The color we perceive from an specific light source is encoded by its spectral distribution. The traditional identification of color with an specific wavelength in the visible spectrum frequently misleads the comprehension of how we actually perceive colors. Light with intensity at a unique wavelength in the visible spectrum is called "pure color". The sensorial response to a "pure color" match closely with the colors observed in

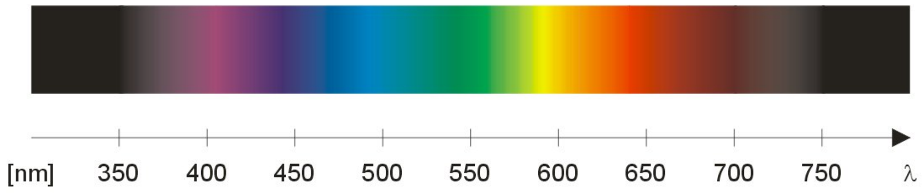


Figure 8.1: Visble Spectrum. Adapted from `http://www.giangrandi.ch/optics/spectrum/spectrum.shtml`.

Figure 8.1. For instance, light with intensity at wavelength 400nm is actually perceived blue, at 500nm is perceived green and at 700nm is perceived red. However "pure colors" are closer to laboratory constructions, than to natural phenomena.

Traditional light sources emit electromagnetic radiation at several wavelength and intensities. In Figure 8.2 we observe the light spectrum of a phosphor-based white LED.

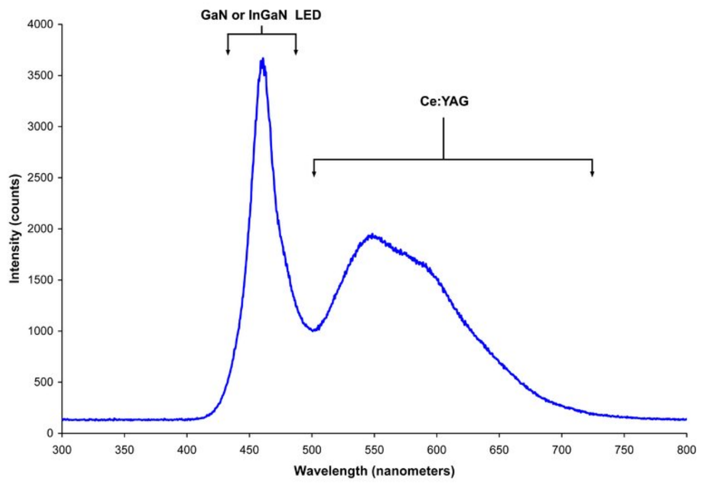


Figure 8.2: Phosphor based white LED spectrum. Adapted from `https://en.wikipedia.org/wiki/Light-emitting_diode`.

Visible light satisfies the *superposition principle*. When lights L_1 and

L_2, with spectral distributions $\rho_{L_1}(\lambda)$ and $\rho_{L_2}(\lambda)$, are added together, the resultant light will have spectral distribution,

$$\rho_{L_1+L_2}(\lambda) = \rho_{L_1}(\lambda) + \rho_{L_2}(\lambda) \tag{8.1}$$

The superposition principle plays a fundamental role in the study of color. It allows a characterization of color perception by using a simple linear model. This model will be discussed in the next section. Figure 8.2 also evidences how the superposition principle can be used in practice: the perceived white color is the superposition of a blue LED and yellow phosphor spectrum.

8.1.2 Color Vision

Human color perception is due to the stimulation of specialized photoreceptors called cones. The cones are classified as L-cones, M-cones or S-cones, according to their specific light spectrum response. The names L, M and S refers precisely to the kind of wavelengths the cone is most sensitive to. For instance, L-cones are the most sensitive to long wavelengths, M-cones to medium wavelength, and S-cones to short wavelengths.

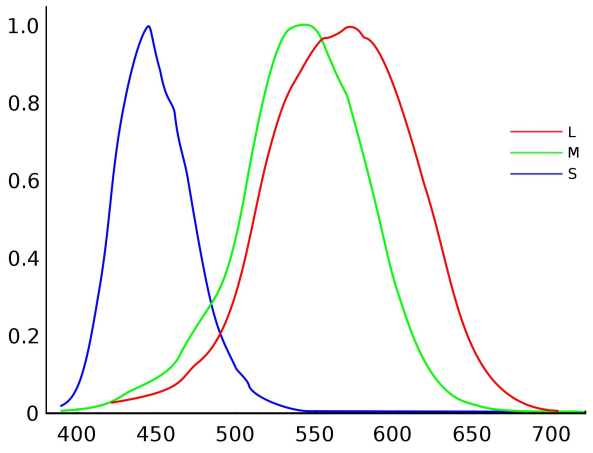


Figure 8.3: Cones Spectral Response. Adapted from `http://www.embedded-vision.com`.

The stimulation a cone experiments under certain light, is calculated by weighting the light spectral distribution with the cone spectrum response (Figure 8.3). This is:

$$R_X(L) = \int_{400}^{700} \rho_L(\lambda) C_X(\lambda) d\lambda \qquad X \in \{L, M, S\} \tag{8.2}$$

From the previous characterization we can deduce that a cone can have the same stimulation for lights with different spectral distribution. In other words, lights L_1 and L_2, will produce the same stimulation in a X-cone as long as,

$$\int_{400}^{700} \rho_{L_1}(\lambda) C_X(\lambda) d\lambda = \int_{400}^{700} \rho_{L_2}(\lambda) C_X(\lambda) d\lambda \tag{8.3}$$

The triplet of cones stimulations defines the color perception. Two lights produce an identical color perception whenever they lead to the same triplet (R_L, R_M, R_S). Any set of lights with different spectral distributions but producing a same color sensation are called metamerisms.

The L, M and S cones also differ in their distribution at the retina. The amount of cells of each type, can vary a lot from a person to other according to factors such as gender and race. In general, L-cones and M-cones are densely distributed in the fovea (central retina) in similar proportions, while S-cones occupies the external part of the retina in a considerable smaller proportion.

Due to such cells distribution we can comprehend why human visual system has a better perception of green and red colors than blue. If you slowly move Figure 8.4 apart from your eyes, eventually, black lines in the blue pattern will disappear, while black lines in the green and red pattern are still visible. This phenomena called visual antialiasing is due to the larger sampling density of L-cones and M-cones compared to S-cones.

8.1.3 Luminance Response

Luminance is non-linearly perceived. As can be observed from Figure 8.5a, an achromatic light source with mid intensity produce a visual sensation closer to white than black. Therefore, a larger range of perceptually different tones can be identified at low levels than at high level of intensity. Due to this phenomena, it is common to apply a non-linear transformation to attain

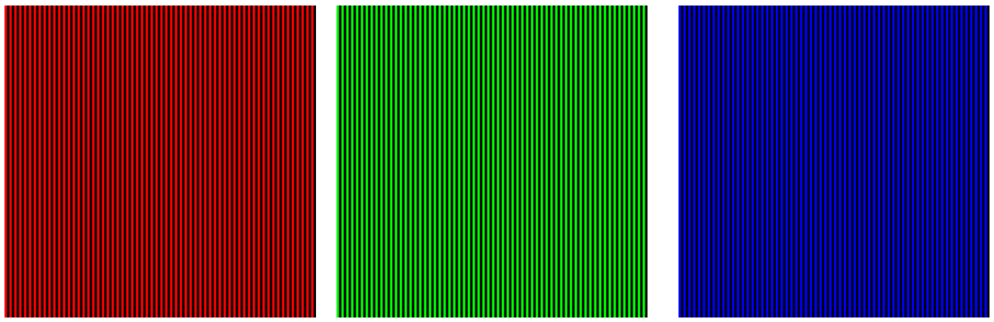
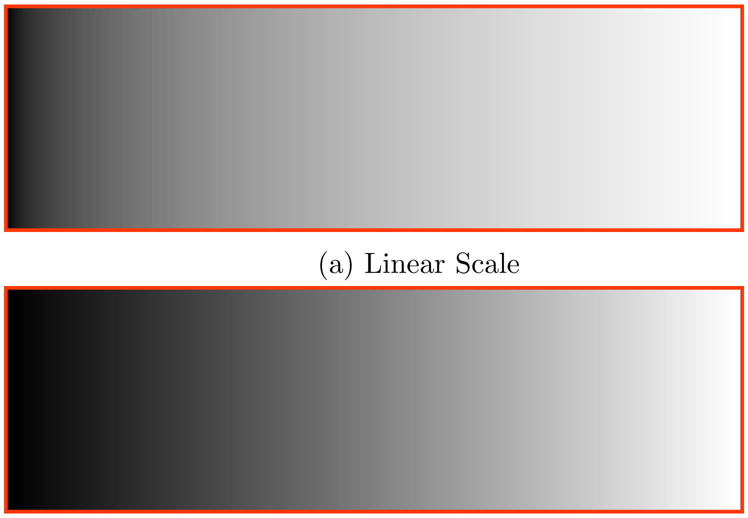


Figure 8.4: Color Patterns

a more compact and perceptually correct intensity curve. This is precisely done in Figure 8.5b, where a gamma transform was applied to the linear scale intensity.



(a) Linear Scale

(b) Perceptual Scale

Figure 8.5: Luminance Response

8.2 Color Spaces

Human color perception can be well approximated on a finite dimensional space. Due to the simplicity and accuracy of the trichromatic model, color space usually requires of just three coordinates to represent any perceived color. This dimensional reduction allow efficient manipulation, transmission and storage of color information.

8.2.1 Linear Color Spaces

This spaces are characterized by a linear transformation, $T : (\mathbb{R} \to \mathbb{R}) \to \mathbb{R}^n$, mapping the light spectrum ρ_L to a finite dimensional space. Linear color spaces preserve light superposition principle: given spectrums $\rho_{L_1}, \rho_{L_2} : \mathbb{R} \to \mathbb{R}$, and $\alpha \in \mathbb{R}$, we have $T(\alpha\rho_{L_1} + \rho_{L_2}) = \alpha T(\rho_{L_1}) + T(\rho_{L_2})$.

The linear transformation associated to a color space can be represented as a set of kernels $T = (T_1, T_2, \ldots, T_n)$, such that, $T(\rho_L) = (T_1(\rho_L), \ldots, T_n(\rho_L))$ and,

$$T_i(\rho) = \int \rho(\lambda) T_i(\lambda) d\lambda \tag{8.4}$$

CIE RGB

The CIE RGB color space has its foundation in the Wright and Guild color-matching experiments. In this experiment, the observer must reconstruct a test "pure color" by adjusting the intensities of three reference primaries. The test "pure color" varies over all the visible spectrum, while the reference primaries are maintained fixed. At the end, we obtain three curves which indicate the intensity of each primary in the reconstruction of "pure colors". These curves are called the color-matching functions.

The CIE comission fixed as reference primaries monochromatic lights at 435.8, 546.1 and 700 nm. The color matching functions of these primaries correspond to the kernels of CIE RGB, illustrated in Figure 8.6.

CIE XYZ

The limitation of the CIE RGB to represent visible spectrum as positive linear combinations of its primaries is one of the motivations to the creation

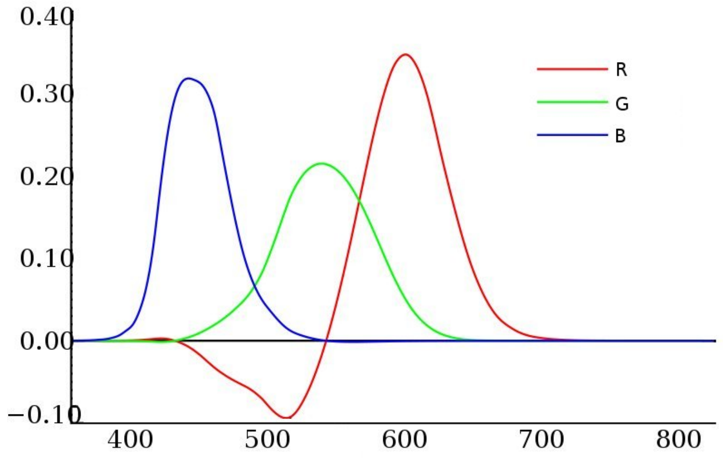


Figure 8.6: CIE RGB kernels. Adapted from `http://en.wikipedia.org/wiki/CIE_1931_color_space`.

of CIE XYZ. To generate a positive cone that contains all the visible spectrum, the CIE XYZ takes its primaries outside the visible domain.

The kernels of CIE XYZ are defined as linear combinations of CIE RGB kernels. The kernel Y approximates the photopic luminance response function. Primaries Z and X where taken such that white point corresponds to $X = Y = Z = 1$. The linear transformation from CIE RGB to CIE XYZ is given by:

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} 0.4900 & 0.3100 & 0.2000 \\ 0.17697 & 0.81240 & 0.01063 \\ 0.0000 & 0.0100 & 0.9900 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} \tag{8.5}$$

The original CIE XYZ color space was proposed in 1931, and is also known as the CIE 1931 standard observer or CIE 2° observer. The term 2° is because color matching experiments were done for two degrees of visual angle. Therfore, light was incident only at the fovea, and no rod intervention considered. The kernels of CIE 1931 are included in Figure 8.7.

The CIE XYZ has been reviewed and adjusted. One of the main additions is the 1964 CIE supplementary standard observer, or 10° standard. This modification fits better for visual angles larger than four degrees.

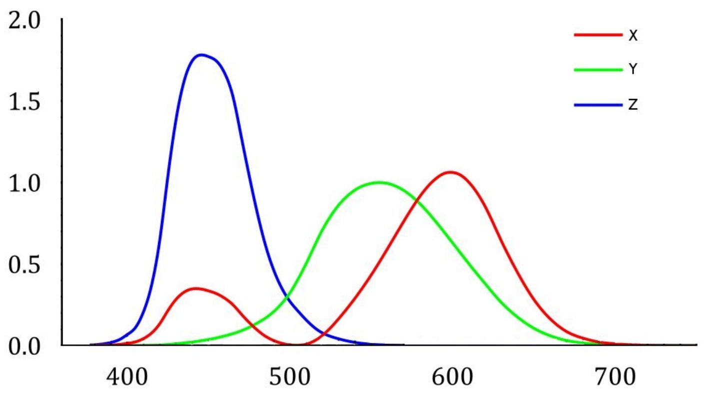


Figure 8.7: CIE XYZ kernels. Adapted from `http://en.wikipedia.org/wiki/CIE_1931_color_space`.

LMS

The LMS color space measures the response of L, M and S cones to each wavelength of the visible spectrum. The kernels of this color space are presented in Figure 8.3.

Since CIE XYZ and LMS kernels are not linearly related, there is not an exact coordinates conversion between both color spaces. An approximation used in practice to transform from LMS coordinates to CIE XYZ is due to Hunt-Pointer-Estevez:

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} 1.9102 & -1.1121 & 0.2019 \\ 0.3710 & 0.6291 & 0.0000 \\ 0.0000 & 0.0000 & 1.0000 \end{pmatrix} \begin{pmatrix} L \\ M \\ S \end{pmatrix} \tag{8.6}$$

8.2.2 Nonlinear Color Spaces

Despite linear color spaces allow an efficient treatment of "color arithmetic", they do not provide a good metric of color perception. Nonlinear spaces are designed to achieve a closer measure of preceived color distances and a more compact transmision of color information.

sRGB

sRGB is a nonlinear color space used for the storage of color information. It is widely used in digital devices due to the efficient color quantization that it provides. The sRGB color space is specified by the ITU_R BT.709 primaries, together with a nonlinear encoding. The transformation from ITU_R BT.709 primaries to CIE XYZ is given by:

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} 0.4124 & 0.3576 & 0.1805 \\ 0.2126 & 0.7151 & 0.0722 \\ 0.0193 & 0.1192 & 0.9505 \end{pmatrix} \begin{pmatrix} R_L \\ G_L \\ B_L \end{pmatrix} \tag{8.7}$$

Since human visual system is more sensitive to luminance variations at low intensity levels, a concave nonlinear encoding improves the usage of bits to store color information. The nonlinear transform used in sRGB is given by:

$$X_{sRGB} = \begin{cases} 1.055(X_L)^{0.416} - 0.055 & \text{if } X_L > 0.0031308 \\ 12.92 X_L & \text{if } X_L \leq 0.0031308 \end{cases} \qquad X \in \{R, G, B\} \tag{8.8}$$

Besides sRGB, there are several standards for digital storage and transmission of color information. Other popular standards are Adobe RGB, PAL, HDTV and NTSC.

CIELAB

CIELAB provides an approximate uniform measure of color distances. The *Lab* representation of a color is computed from its CIE XYZ representation and a reference white point $X_n Y_n Z_n$:

$$\begin{pmatrix} L \\ a \\ b \end{pmatrix} = \begin{pmatrix} 0 & 116 & 0 & -16 \\ 500 & -500 & 0 & 0 \\ 0 & -200 & 200 & 0 \end{pmatrix} \begin{pmatrix} f(\frac{X}{X_n}) \\ f(\frac{Y}{Y_n}) \\ f(\frac{Z}{Z_n}) \\ 1 \end{pmatrix} \tag{8.9}$$

$$f(r) = \begin{cases} \sqrt[3]{r} & \text{if } r > 0.008856 \\ 7.787r + \frac{16}{116} & \text{if } r \leq 0.008856 \end{cases} \tag{8.10}$$

The L component represent the lightness of the color, while a and b are chrominance components. The Lab distance between a pair of colors C_1, C_2 is computed as a euclidean distance of its components:

$$d_{Lab}(C_1, C_2) = |L(C_1) - L(C_2)|^2 + |a(C_1) - a(C_2)|^2 + |b(C_1) - b(C_2)|^2 \quad (8.11)$$

8.2.3 SCIELAB

Point by point color comparison do not always provide an appropriate measure of image appearance differences. Consider images in Figure 8.8. These images are one the inverse of the other. By comparing these images pointwise we would get that they are quite different. However, for an observer who see both images far enough, these images will seem identical. SCIELAB is a model proposed to overcome this problem.

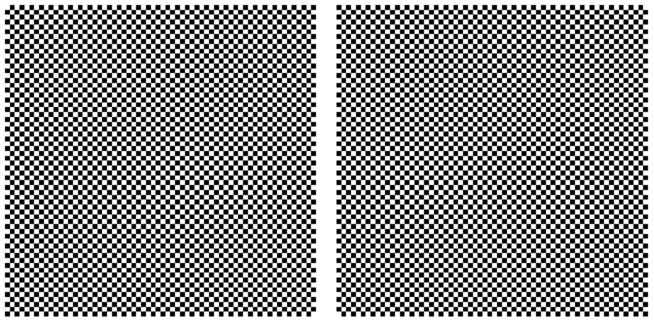


Figure 8.8: Image Appereance

Due to the large overlapping of cones spectral responses (Figure 8.3), human neural system decorrelates these signals to attain more efficient information transmission [16]. This decorrelation process can be interpreted as a linear transformation from the LMS responses to a color space formed by a luminance component and two chromatic opponent components.

The first step of SCIELAB model is precisely the representation of the input images in the opponent color space $O_1O_2O_3$ proposed by Poirson and Wandell [13]. Traditionally, the opponent channels are referred as Black-

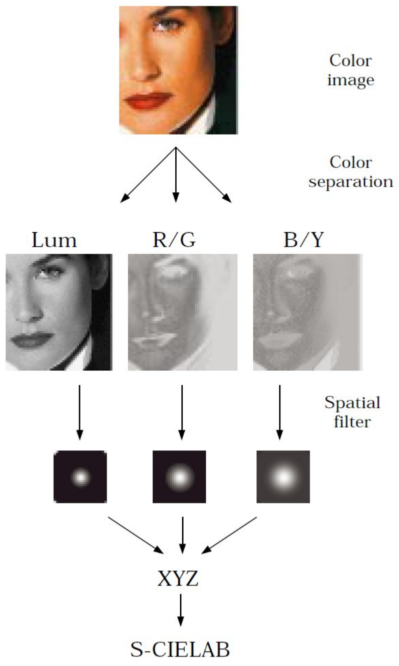


Figure 8.9: SCIELAB. Adapted from [17].

White, Red-Green, and Yellow-Blue. The transformation from CIE XYZ to $O_1O_2O_3$ is given by

$$\begin{pmatrix} O_1 \\ O_2 \\ O_3 \end{pmatrix} = \begin{pmatrix} 0.279 & 0.729 & -0.107 \\ -0.449 & 0.290 & 0.077 \\ 0.086 & 0.590 & -0.501 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} \tag{8.12}$$

The second step of SCIELAB is simulate visual blur on each channel of the opponent space. Blurring kernels are represented as weighted sums of Gaussian functions. The spread of the Gaussian are given in half visual degree units. The values of weights and spreads for each of the blurring kernels are presented in the table below:

Channel	Weight (ω_i)	Spread (σ_i)
Black-White	0.921	0.0283
	0.105	0.133
	-0.108	4.336
Red-Green	0.531	0.0392
	0.330	0.494
Blue-Yellow	0.488	0.0536
	0.371	0.386

The third step of SCIELAB consist in transforming the burred signals from the opponent space $O_1O_2O_3$ to CIELAB. As previoisly discussed, CIELAB space provides a more uniform metric to compute color differences. The final step is to compute mean *Lab* distance (given by equation 8.11) over all the points of the compared images.

8.3 Linear and Nonlinear Filtering

8.3.1 Linear Filters

The linearity attribute indicates wether the output reconstruction just depends linearly on the input texture. By denoting $F(I)$ the reconstruction of filter F applied to texture I, linearity can simply be stated as,

$$F(\alpha I_1 + I_2) = \alpha F(I_1) + F(I_2) \qquad \forall I_1, I_2 \tag{8.13}$$

An special subset of the linear filters are the ones that satisfies the shift invariant property. Let $T_{\boldsymbol{s}}$ represent an integer pixel translation in direction $\boldsymbol{s} = (s_w, s_h)$. We say that filter F is linear shift invariant (LSI) if $F(T_{\boldsymbol{s}}(I)) = T_{\boldsymbol{s}}(F(I))$ for all textures I and integer pixel displacements $\boldsymbol{s}$.

In the spatial domain, LSI filtering can be compactly represented as a convolution with the system impulse response. In the frequency domain, it corresponds to an attenuation of the input spectrum by the impulse response spectrum. Due to its simplicity and reduced computational cost, many of the proposed approaches to subpixel filtering correspond to LSI schemes. The scheme we propose in this work, for the unconstrained optimization case, also lies in this category.

Suppose we have a pixel grid with n different type of subpixels, $\{\varphi_1, \dots, \varphi_n\}$, and let $I = (I_1, I_2, \dots, I_k)$ represent a multichannel input texture. Let F be a LSI filter, such that $F(I) = \boldsymbol{c} = (\boldsymbol{c}_1, \boldsymbol{c}_2, \dots, \boldsymbol{c}_n)$. By the linearity of F we have that,

$$F(I) = F(I_1, 0, \dots, 0) + F(0, I_2, \dots, 0) + \ldots + F(0, 0, \dots, I_k) \tag{8.14}$$

Let $F_i(I_i) := F(0, \dots, I_i, \dots, 0)$. Observe that each component-function F_i is LSI. Define $F_i(I_i) := (\boldsymbol{c}_{1i}, \boldsymbol{c}_{2i}, \dots, \boldsymbol{c}_{ni})$, and denote $F_{si}(I_i) := \boldsymbol{c}_{si}$. This is also a LSI system. Therefore, we can write $F_{si}(I_i) = [I_i * k_{si}]$, for some kernel $k_{si} : \mathbb{R} \to \mathbb{R}$. Adding over all the channels we obtain,

$$\boldsymbol{c}_s = \sum_i \boldsymbol{c}_{si} = \sum_i [I_i * k_{si}] = [I * k_s]^+ \tag{8.15}$$

From the results above, we conclude that a LSI scheme, in the context of subpixel filtering, can be characterized by a set of multichannel kernels $\{k_1, k_2, \dots, k_n\}$. Here, k_s is a multichannel kernel associated to s^{th} subpixel, which acts independently in each channel of the input. The operator $()^+$ consolidates the result by adding over all the channels.

Cross-Channel Filter

We say that a filter strategy F is cross-channel, if the filtering process for each subpixel involves the complete range of channels of the input texture. Traditional texture filtering in RGB striped displays is an example of non cross-channel filtering. The reconstruction coefficients for the R, G and B subpixels depends exclusively on the input texture R, G and B channels, respectively. Figure 1.5 is an example of non cross-channel filtering.

Many relevant works in the context of cross-channel schemes propose filtering the signal in an opponent color space. Some of the most common LSI filters (e.g.,[5]) can be represented by[1],

$$\boldsymbol{c}_s = [B(A(I) * k) * \delta_s]^+ \tag{8.16}$$

[1] An alternative representation is given by $\boldsymbol{c}_s = [I * A^t D_{B^t \delta_s} k(\cdot - \delta_s)]^+$. The symbol δ_s is overloaded to indicate the color coordinates of the subpixel in the reference color space, and the shift towards subpixel centroid.

In this equation, we have two color space transformations: A from the input color space to the opponent color space, and B form the opponent color space to the reference color space. k is a filter that acts in each of the opponent channels independently. δ_s is an impulse, shifted in the direction of the subpixel centroid, which also indicates the subpixel color coordinates in the reference color space.

The central discussion on the family of methods described by equation 8.16 turns around the lenght of the band that should be passed in each opponent channel. The common objective is to increase luminance resolution at reduced color fringing cost. For the luminance channel, authors suggests using low pass filter something below Nyquist, e.g., 2/3 cycles per pixel. For the chrominance channels is usually suggested to maintain 1/2 cycles per pixel as the cut off frequency.

8.3.2 Non Linear Filters

Linear filters present some limitations. Physical displays can only turn on subpixels at a limited positive range of intensities, say $[0, 1]$. Therefore, it is desirable that reconstruction coefficients always belong to this interval. For linear filtering this is in general not true, specially, whenever the filter is close to a perfect low pass filter. In linear filtering schemes, reconstruction coefficients outside the $[0, 1]$ range are usually truncated. This is sometimes undesirable, since ringing patterns may appear.

Another limitation of linear filters arise when we are interseted to find an image reconstruction that approximate the input texture in a perceptual sense. If we take into account the blur and the nonlinear intensity response of the human visual system, a nonlinear filtering scheme is required [7].

Nonlinear filtering approaches are usually formulated as linear, quadratic, or even nonlinear programming problems with box constraints. Due to its computational cost, these approaches are not used in practice.

Chapter 9

Appendix

In this appendix we introduce some notation and definitions used for the mathematical treatment of our problem. In the general notation section we review well established operators common to the standard signal processing literature. On the hierarchical notation section we present our own definitions and operators with the aim of getting to more compact mathematical expressions (e.g., avoid indices overload).

9.1 General Notation

9.1.1 Functional Domains

In this work we will be concerned on functions over four different domains:

- $\mathbb{R}^n$ denote the usual n-dimensional continuous domain.
- $\mathbb{Z}^n$ denote the usual n-dimensional discrete domain.
- $\mathbb{T}_{\boldsymbol{m}}$ denote the n-dimensional torus, $\mathbb{T}_{\boldsymbol{m}} = [0, m_1] \times [0, m_2] \ldots \times [0, m_n]$, for certain $\boldsymbol{m} \in \mathbb{N}^n$.
- $\mathbb{Z}_{\boldsymbol{m}}$ denote the cyclic discrete domain $\mathbb{Z}_{\boldsymbol{m}} = \mathbb{Z}_{m_1} \times \mathbb{Z}_{m_2} \times \ldots \times \mathbb{Z}_{m_n}$ for certain $\boldsymbol{m} \in \mathbb{N}^n$.

The following table summarizes the nature of each domain:

	Continuous	Discrete
Aperiodic	$\mathbb{R}^n$	$\mathbb{Z}^n$
Periodic	$\mathbb{T}_{\boldsymbol{m}}$	$\mathbb{Z}_{\boldsymbol{m}}$

We will denote the continuous domains by Ω_C. Therefore, Ω_C will refer to either $\mathbb{R}^n$ or $\mathbb{T}_{\boldsymbol{m}}$. Similarly, we will use Ω_D to refer either $\mathbb{Z}^n$ or $\mathbb{Z}_{\boldsymbol{m}}$.

To indicate functions over continuous domains, we will use *unbolded* letters, e.g, f, g, h. For functions over discrete domains we will use ***bolded*** letters $\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}$.

For aperiodic domains, coordinates arithmetic is as usual. For periodic domains coordinate arithmetic is modular.

Additionally, we use the notation $\{\cdot\}$ to refer to a set of indices. In particular, if n is a natural number, we denote

$$\{\boldsymbol{n}\} = \{1, 2, \ldots n\} \tag{9.1}$$

9.1.2 Sampling Operator and Dirac δ

- Given $f : \Omega_C \to \mathbb{R}$, we define $[f] : \Omega_D \to \mathbb{R}$, by:

$$[f]_k = f(k) \tag{9.2}$$

- The Dirac δ is a distribution which concentrates all its weight at the origin. For any $f : \Omega_C \to \mathbb{R}$, we have

$$\int_{\Omega_C} f(x)\delta(x)dx = f(0) \tag{9.3}$$

  Given $\boldsymbol{a} : (\Omega_D \to \mathbb{R})$, we define $a_\delta : (\Omega_C \to \mathbb{R})$, by:

$$a_\delta = \sum_{i \in \Omega_D} \boldsymbol{a}_i \delta(\cdot - i) \tag{9.4}$$

9.1.3 Norms

Instances.

- Given $\boldsymbol{a} \in \Omega_D \to \mathbb{R}$, we define:

$$||\boldsymbol{a}||_1 = \sum_{i\in\Omega_D} |\boldsymbol{a}_i|, \qquad ||\boldsymbol{a}||_2 = \sum_{i\in\Omega_D} |\boldsymbol{a}_i|^2 \tag{9.5}$$

  When $||\boldsymbol{a}||_1 < \infty$ and $||\boldsymbol{a}||_2 < \infty$, we say $\boldsymbol{a} \in l_1$ and $\boldsymbol{a} \in l_2$, resp.

- Given $f \in \Omega_C \to \mathbb{R}$, we define:

$$||f||_1 = \int_{\Omega_C} |f(t)|dt, \qquad ||f||_2 = \int_{\Omega_C} |f(t)|^2 dt \tag{9.6}$$

  When $||f||_1 < \infty$ and $||f||_2 < \infty$, we say $f \in L_1$ and $f \in L_2$, resp.

Properties.

- $||u||_p \geq 0$. $||u||_p = 0 \iff u \equiv 0$.
- $||\alpha u||_p = |\alpha|||u||_p$.
- $||u+v||_p \leq ||u||_p + ||v||_p$.

9.1.4 Inner Product

Instances.

- Discrete Inner Product: Given $\boldsymbol{a}, \boldsymbol{b} \in l_2(\Omega_D \to \mathbb{R})$, we define:

$$\langle \boldsymbol{a}, \boldsymbol{b} \rangle = \sum_{i\in\Omega_D} \boldsymbol{a}_i \boldsymbol{b}_i \tag{9.7}$$

- Continuous Inner Product: Given $f, g \in L_2(\Omega_C \to \mathbb{R})$, we define:

$$\langle f, g \rangle = \int_{\Omega_C} f(t)g(t)dt \tag{9.8}$$

The norm $||\cdot||_2$ is induced by the inner product:

$$||u||_2 = \sqrt{\langle u, u\rangle} \tag{9.9}$$

Properties.

- $\langle \alpha u + v, w\rangle = \alpha\langle u, w\rangle + \langle v, w\rangle$.
- $\langle u, v\rangle = \langle v, u\rangle$.
- $|\langle u, v\rangle| \leq ||u||_2||v||_2$.

9.1.5 Convolution

Instances.

- Continuous Convolution: Given $f, g \in L_1(\Omega_C \to \mathbb{R})$ we define, $h := f * g \in L_1(\Omega_C \to \mathbb{R})$, by:

$$h(x) = (f * g)(x) = \int_{\Omega_C} f(x-t)g(t)dt \tag{9.10}$$

- Discrete Convolution: Given $\boldsymbol{a}, \boldsymbol{b} \in l_1(\Omega_D \to \mathbb{R})$, we define $\boldsymbol{c} := \boldsymbol{a} * \boldsymbol{b} \in l_1(\Omega_D \to \mathbb{R})$ by:

$$\boldsymbol{c}(k) = (\boldsymbol{a} * \boldsymbol{b})_k = \sum_{i\in\Omega_D} \boldsymbol{a}_{k-i}\boldsymbol{b}_i \tag{9.11}$$

- Discrete-Continuos Convolution: Given $\boldsymbol{a} \in l_1(\Omega_D \to \mathbb{R})$, $f \in L_1(\Omega_C \to \mathbb{R})$, we define $\boldsymbol{g} := \boldsymbol{a} * \boldsymbol{f} \in L_1(\Omega_C \to \mathbb{R})$ by:

$$\boldsymbol{g}(x) = (\boldsymbol{a} * f)(x) = \sum_{i\in\Omega_D} \boldsymbol{a}_i f(x-i) \tag{9.12}$$

  Remark:The definition above corresponds to the continuous convolution $a_\delta * f$.

Properties.

- $u * v = v * u$
- $(u * v) * w = u * (v * w)$
- $u * (\alpha v + w) = \alpha u * v + u * w$
- $||u * v||_1 = ||u||_1 ||v||_1$

9.1.6 Fourier Transform

Instances.

- Continuous Fourier Transform: Given $f \in L_2(\mathbb{R}^n \to \mathbb{R})$, we define $\hat{f} \in L_2(\mathbb{R}^n \to \mathbb{C})$ by:

$$\hat{f}(\omega) = \int_{\mathbb{R}^n} f(x) e^{-2\pi i \omega x} dx \tag{9.13}$$

- Discrete Fourier Transform: Given $\boldsymbol{a} \in l_2(\mathbb{Z}_{\boldsymbol{m}} \to \mathbb{R})$, we define, $\hat{\boldsymbol{a}} \in l_2(\mathbb{Z}_{\boldsymbol{m}} \to \mathbb{C})$ by:

$$\hat{\boldsymbol{a}}_{\boldsymbol{\omega}} = \sum_{\boldsymbol{k} \in \mathbb{Z}_{\boldsymbol{m}}} \boldsymbol{a}_k e^{-2\pi i \langle \boldsymbol{\omega}, \boldsymbol{k} \rangle} \tag{9.14}$$

Properties.

- $\mathrm{FT}(\alpha u + v) = \alpha \,\mathrm{FT}(u) + \mathrm{FT}(v)$
- $\mathrm{FT}(\mathrm{FT}(u)) = u$
- $\mathrm{FT}(u * v) = \mathrm{FT}(u)\,\mathrm{FT}(v)$
- $||\,\mathrm{FT}(u)||_2 = ||u||_2$

9.2 Hierarchical Notation

In aim of simplicity, we introduce a hierachical notation that will guide the arithmetic treatment of our problem. Below we define, in order of complexity, three kind of structures: channel, channel matrix, and multichannel matrix (Figure 9.2). For each level of complexity we define some operators. All operators defined at certain level of complexity are extended to superior levels.

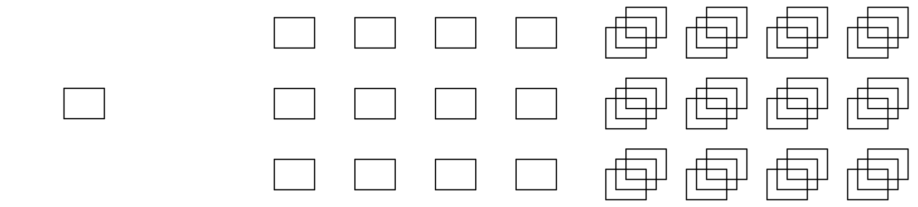

(a) Channel (b) Channel Matrix (c) Multichannel Matrix

Figure 9.1: Hierarchical Notation

9.2.1 Channel

A channel is a function $\boldsymbol{u} : \Omega_D \to \mathbb{R}$, $\boldsymbol{u} \in l_1 \cap l_2$, or $u : \Omega_C \to \mathbb{R}$, $u \in L_1 \cap L_2$. Domain $\Omega_C \in \{\mathbb{Z}^n, \mathbb{Z}_{\boldsymbol{m}}\}$, and $\Omega_D \in \{\mathbb{R}^n, \mathbb{T}_{\boldsymbol{m}}\}$. According to the nature of the domain we will distinguish between four *types* of channels: discrete-periodic, discrete-aperiodic, continuous-periodic, and continuous-aperiodic.

Operators.

- Arithmetic Operators $(+, -, \ldots)$: For u and v channels of the same *type*, arithmetic operators are defined as usual in scalar functions. This is, $(\alpha u)(x) = \alpha(u(x))$, $(u + v)(x) = u(x) + v(x)$, $(uv)(x) = u(x)v(x)$, etc.
- Convolution $(*)$: For u and v channels of the same periodic *type*, convolution is as defined in 9.10,9.11, or 9.12.
- Inner Product $(\langle , \rangle)$: For u and v channels of the same *type*, inner product is defined as in 9.7 or 9.8.
- Norms $(|| \cdot ||_1, || \cdot ||_2)$: As defined in 9.5 or 9.6.
- Reflect $(^\vee)$: $u^\vee(x) = u(-x)$.
- Sampling $([\cdot])$: As is defined in 9.2.

Properties.

Since channels are defined in $l_1 \cap l_2$ and $L_1 \cap L_2$, all the properties for inner product 9.1.4 and convolution 9.1.5 hold. The following are also valid:

- If u, v,and w, are channels of the same *type*, then,

$$\langle u * v, w\rangle = \langle v, u^{\vee} * w\rangle = \langle u, w * v^{\vee}\rangle \tag{9.15}$$

- If $\boldsymbol{u}, v$,and w, are channels of the same periodic *type*, $\boldsymbol{u}$ discrete, v, w continuous, then,

$$\langle \boldsymbol{u} * v, w\rangle = \langle \boldsymbol{u}, [w * v^{\vee}]\rangle \tag{9.16}$$

9.2.2 Channel Matrix

A channel matrix is a two dimensional array formed by channels of the same *type*. A channel matrix corresponds to a function $\Omega \times (\{\boldsymbol{n}\} \times \{\boldsymbol{m}\}) \to \mathbb{R}$. The *type* of the channel matrix is inherited from its channels. If U is a channel matrix with n rows and m columns, we say its *dimensions* are $n \times m$. We denote by U_{ij} to the channel at the i-th row and j-th column.

Operators.

- Arithmetic Operators $(+, -, \ldots)$: Let U and V channels matrices of the same *type*, and same *dimensions*. Arithmetic operators act componentwise: $(\alpha U)_{ij} = \alpha(U_{ij})$, $(U+V)_{ij} = U_{ij} + V_{ij}$, $(UV)_{ij} = U_{ij}V_{ij}$, etc.

- Convolution $(*)$: Let U and V be channels matrices of the same periodic *type*, where U is $n \times m$ and V is $m \times h$. Then $U * V$ is a $n \times h$ channel matrix defined by:

$$(U * V)_{ij} = \sum_{h=1}^{m} U_{ih} * V_{hj} \tag{9.17}$$

- Inner Product $(\langle, \rangle)$: Let U and V be channels matrices of the same *type* and *dimensions*, we define

$$\langle U, V\rangle = \sum_{ij} \langle U_{ij}, V_{ij}\rangle \tag{9.18}$$

- Norms $(||\cdot||_1, ||\cdot||_2)$: Let U be a channel matrix, we define,

$$||U||_1 = \sum_{ij} ||U_{ij}||_1, \qquad ||U||_2 = \sum_{ij} \sqrt{||U_{ij}||_2^2} \tag{9.19}$$

- Reflect ($^\vee$): Acts componentwise, $(U^\vee)_{ij} = U_{ij}^\vee$.
- Sampling ($[\cdot]$) : Acts componentwise, $[U]_{ij} = [U_{ij}]$.
- Transpose ($^\top$): Let U be a $n \times m$ channel matrix. $U^\top$ is $m \times n$ channel matrix defined by
$$(U^\top)_{ij} = U_{ji} \tag{9.20}$$

Properties.

- If U, V,and W, are channels matrices of the same type, then,
$$\langle U * V, W\rangle = \langle V, (U^\vee)^\top * W\rangle = \langle U, W * (V^\vee)^\top\rangle \tag{9.21}$$
- If $\boldsymbol{U}, V$,and W, are channel matrices of the same periodic type, $\boldsymbol{U}$ discrete, V, W continuous then,
$$\langle \boldsymbol{U} * V, W\rangle = \langle \boldsymbol{U}, [W * V^\vee]\rangle \tag{9.22}$$
- If U, V are channels matrices of the same periodic type, then,
$$(U * V)^\top = V^\top * U^\top \tag{9.23}$$

Remarks

- Channel matrices of *dimensions* $n \times 1$ will be called channel vectors.
- Convolution $U * V$ between channels matrices is in general non conmutative. Instead, it resembles usual matrix product.

9.2.3 Multichannel Matrix

A multichannel matrix is a one dimensional array formed by channel matrices of the same *type* and *dimensions*. A multichannel matrix corresponds to a function $\big(\Omega \times (\{\boldsymbol{n}\} \times \{\boldsymbol{m}\})\big) \times \{\boldsymbol{k}\} \to \mathbb{R}$. The *type* and *dimensions* of the multichannel matrix are inherited from its channels matrices. The number of channels matrices defines the *depth*. The channels matrix at depth l is denoted by U^l.

Operators.

- Arithmetic Operators $(+, -, \ldots)$: Let U and V multichannels matrices of the same *type*, *dimensions*, and *depth*. Arithmetic operators act componentwise: $(\alpha U)^l = \alpha(U^l)$, $(U+V)^l = U^l + V^l$, $(UV)^l = U^l V^l$, etc.

- Convolution $(*)$: Let U and V be *multichannel matrices* of the same periodic *type* and *depth* k, U of *dimensions* $n \times m$, and V $m \times h$. Then $U * V$ is a $n \times h$ multichannel matrix of *depth* k defined by:

$$(U * V)^l = U^l * V^l \tag{9.24}$$

- Inner Product $(\langle , \rangle)$: Let U and V be multichannels matrices of the same *type*, *dimensions*, and *depth*, we define

$$\langle U, V \rangle = \sum_k \langle U^l, V^l \rangle \tag{9.25}$$

- Norms $(|| \cdot ||_1, || \cdot ||_2)$: Let U be a *multchannel matrix*, we define,

$$||U||_1 = \sum_l ||U^l||_1, \qquad ||U||_2 = \sum_l \sqrt{||U^l||_2^2} \tag{9.26}$$

- Reflect $(^\vee)$: Acts componentwise: $(U^\vee)^l = (U^l)^\vee$.

- Sampling $([\cdot])$: Acts componentwise: $[U]^l = [U^l]$.

- Transpose $(^\top)$: Acts componentwise: $(U^\top)^l = (U^l)^\top$.

- Linear Transform (A): Let U a be multichannel matrix of *dimensions* $n \times m$, *depth* k, and $A \in \mathcal{L}(\mathbb{R}^k, \mathbb{R}^d)$. Then, AU is a multichannel matrix of *dimensions* $n \times m$ and *depth* d defined by:

$$(AU)^l_{ij} = \sum_{p=1}^{k} A_{lp} U^p_{ij} \tag{9.27}$$

- Reduce ($^+$): Let U a be multichannel matrix of *dimensions* $n \times m$ and *depth* k. Then, U^+ is a channel matrix of *dimensions* $n \times m$ defined by:

$$(U^+)_{ij} = \sum_{p=1}^{k} U_{ij}^p \tag{9.28}$$

Properties.

- If U, V,and W, are multichannel matrices of the same *type*, then,

$$\langle U * V, W \rangle = \langle V, (U^\vee)^\top * W \rangle = \langle U, W * (V^\vee)^\top \rangle \tag{9.29}$$

- If $\boldsymbol{U}, V$,and W, are multichannel matrices of the same periodic *type*, $\boldsymbol{U}$ discrete, V, W continuous, then,

$$\langle \boldsymbol{U} * V, W \rangle = \langle \boldsymbol{U}, [W * V^\vee] \rangle \tag{9.30}$$

- If U, V are multichannel matrices of the same periodic *type*, then,

$$(U * V)^\top = V^\top * U^\top \tag{9.31}$$

- If U, V are multichannel matrices of the same *type* and *dimensions*, U of *depth* k, V of *depth* d, and $A \in \mathcal{L}(\mathbb{R}^k, \mathbb{R}^d)$, then,

$$\langle AU, V \rangle = \langle U, A^\top V \rangle \tag{9.32}$$

- Let U, V be multichannel matrices of the same periodic *type*, U of *dimensions* $n \times m$ and *depth* k, V of *dimensions* $m \times h$ and *depth* d. Let $A \in \mathcal{L}(\mathbb{R}^k, \mathbb{R}^d)$, then,

$$(AU * V)^+ = (U * A^\top V)^+ \tag{9.33}$$

Remarks

- Reduce operator $(+)$ is a shortcut for the linear operator $\mathbf{1} = \underbrace{(1 \ldots 1)}_{k}$.

- Multichannel matrices of *dimensions* $n \times 1$ and 1×1 will be called multichannel vectors and multichannels, respectively.

9.3 Filtering

9.3.1 Sampling Theorem

The Sampling Theorem states that a band limited function f can be exactly reconstructed if it is regularly sampled with a rate which is at least twice its largest frequency (Nyquist frecuency). The exact reconstruction is given by the Whittaker - Shannon interpolation formula:

$$f = [f] * \varphi_{\text{sinc}} \qquad \varphi_{\text{sinc}}(x) = \frac{\sin(x)}{x} \tag{9.34}$$

Sampling with a frequency inferior to Nyquist introduce high frequencies to the base band. This phenomena, called aliasing, prevent an exact reconstruction of the original function.

9.3.2 Reconstruction Spaces

We denote by V_φ to the space of functions generated by the reconstruction kernel $\varphi : \mathbb{R}^n \to \mathbb{R}$ on a regular grid. Concretely,

$$V_\varphi = \{\boldsymbol{c} * \varphi \mid \boldsymbol{c} : \mathbb{Z}^n \to \mathbb{R}\} \tag{9.35}$$

The array $\boldsymbol{c}$ is called reconstruction coefficients.

Given a function $f : \mathbb{R}^n \to \mathbb{R}$, the reconstruction problem looks for a representation of f in the reconstruction space V_φ. We will be concerned with to types of reconstruction:

Interpolation

We say that $c * \varphi$ is an interpolative reconstruction of f whenever $[f] = [\boldsymbol{c} * \varphi]$. This implies:

$$\boldsymbol{c} = [\varphi]^{-1}[f] \tag{9.36}$$

If $[\varphi] = \delta$ we say that the reconstruction kernel φ is interpolative. When φ is interpolative the reconstruction coefficients corresponds to the function sampled values.

Orthogonal Projection

The orthogonal projection of f in the reconstruction space V_φ is the element that minimizes $||f - \boldsymbol{c} * \varphi||_2$. This is precisely attained for the reconstruction coefficient:

$$\boldsymbol{c} = [\varphi * \varphi^\vee]^{-1}[f * \varphi^\vee] \tag{9.37}$$

9.3.3 Partition of unity

Definition 9.1. *We say $\varphi : \mathbb{R}^n \to \mathbb{R}$, satisfies partition of unity if,*

$$\sum_{\boldsymbol{i} \in \mathbb{Z}^n} \varphi(x + i) = 1 \qquad \forall x \in [0,1]^n \tag{9.38}$$

Proposition 9.1. *In the frequency domain, the condition of partition of unity is equivalent to,*

$$\hat{\varphi}(n) = \begin{cases} 1 & \textit{if } n = 0. \\ 0 & \textit{if } n \in \mathbb{Z}/\{0\}. \end{cases} \tag{9.39}$$

Proposition 9.2. *Let $\varphi : \mathbb{R}^n \to \mathbb{R}$ satisfies partition of unity and $||\psi||_1 = 1$. Then $\varphi * \psi$ satisfies partition of unity.*

Bibliography

[1] Claude Betrisey, James F. Blinn, Bodin Dresevic, Bill Hill, Greg Hitchcock, Bert Keely, Don P. Mitchell, John C. Platt, and Turner Whitted. 20.4: Displaced filtering for patterned displays. *SID Symposium Digest of Technical Papers*, 31(1):296–299, 2000.

[2] Thomas L. Credelle Candice H. Brown Elliot and Michal F. Higgins. Adding a white subpixel. `http://www.clairvoyante.com/files/pdf/Adding-a-White.pdf`, 2005.

[3] Thomas Engelhardt, Thorsten-Walther Schmidt, Jan Kautz, and Carsten Dachsbacher. Low-cost subpixel rendering for diverse displays. *Computer Graphics Forum*, 2013.

[4] Lu Fang, O.C. Au, and Ngai-Man Cheung. Subpixel rendering: from font rendering to image subsampling. *Signal Processing Magazine, IEEE*, 30(3):177–189, 2013.

[5] Lu Fang, Oscar C. Au, Ketan Tang, and Aggelos K. Katsaggelos. Antialiasing filter design for subpixel downsampling via frequency-domain analysis. *Trans. Img. Proc.*, 21(3):1391–1405, March 2012.

[6] Joyce Farrell, Shalomi Eldar, Kevin Larson, Tanya Matskewich, and Brian Wandell. Optimizing subpixel rendering using a perceptual metric. *Journal of the Society for Information Display*, 19(8):513–519, 2011.

[7] J. Kajiya and M. Ullner. Filtering high quality text for display on raster scan devices. *SIGGRAPH Comput. Graph.*, 15(3):7–15, August 1981.

[8] Michiel A. Klompenhouwer and Gerard De Haan. Subpixel image scaling for color-matrix displays. *Journal of the Society for Information Display*, 11(1):99–108, 2003.

[9] D.S. Messing, L. Kerofsky, and S. Daly. Subpixel rendering on non-striped colour matrix displays. In *Image Processing, 2003. ICIP 2003. Proceedings. 2003 International Conference on*, volume 2, pages II–949–52 vol.3, 2003.

[10] Microsoft. Cleartype. `http://www.microsoft.com/typography/ClearTypeInfo.mspx`.

[11] Nouvoyance. Pentile rgbw. `http://www.nouvoyance.com/technology.html`.

[12] John C. Platt. Optimal filtering for patterned displays. *Signal Processing Letters, IEEE*, 7(7):179–181, 2000.

[13] Allen B. Poirson and Brian A. Wandell. The appearance of colored patterns: Pattern-color separability. *J. OPT. SOC. AM. A*, 10:2458–2470, 1993.

[14] Sharp. Quattron. `http://www.quattron.tv`.

[15] Raymond Soneira. Color or market gambit? `http://dl.maximumpc.com/Archives/MPC0710-web.pdf`, 2010.

[16] Brian A. Wandell. *Foundations of vision.* Sinauer Associates Inc; First Edition, 1995.

[17] X. Zhang and B. A. Wandell. A spatial extension of cielab for digital color-image reproduction. *Journal of the Society for Information Display*, 5(1):61–63, 1997.