UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

GIOVANE ROSLINDO KUHN

Image Recoloring for Color-Vision Deficients

Thesis presented in partial fulfillment
of the requirements for the degree of
Master of Computer Science

Prof. Dr. Manuel Menezes de Oliveira Neto
Advisor

Porto Alegre, April 2008

CIP – CATALOGING-IN-PUBLICATION

Kuhn, Giovane Roslindo

Image Recoloring for
Color-Vision Deficients / Giovane Roslindo Kuhn. – Porto Alegre: PPGC da UFRGS, 2008.

61 f.: il.

Thesis (Master) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2008. Advisor: Manuel Menezes de Oliveira Neto.

1. Color Reduction, Color-Contrast Enhancement, Color-to-Grayscale Mapping, Color Vision Deficiency, Dichromacy, Error Metric, Image Processing, Monochromacy, Recoloring Algorithm. I. Oliveira Neto, Manuel Menezes de. II. Title.

“Knowledge becomes knowledge, when it’s in the mind of everyone.”
— ANONYMOUS

ACKNOWLEDGMENTS

I dedicate this work to two people in particular: Roberta Magalhães, my faithful companion, even in the distance gave me all your love and comprehension, fundamental that I do not give up the master's program, make my bags and get back home. Edir Roslindo Kuhn, my dear mother, for her thirst for life and strength in difficult moments, which spread me and keep moving forward.

I thank immensely my advisor Manuel Menezes de Oliveira Neto, even with his post-abolitionist method, he always took me to my limit and made me grow professionally in a unprecedented way in my life. I thank him for the various conversations about his experience both in professional and private life, which certainly were a learning process for me. I would also like to thank for his style of work, which always yielded the best jokes in the labs and conversations at parties.

I would like to thank Flavio Brum, a dichromat who was the great inspiration for the development of this work and responsible for valuable priceless suggestions. I thank Sandro Fiorini, an anomalous trichromat, for the various testimonies of the difficulties faced by him on daily tasks. I am grateful to Carlos Dietrich for the discussions about mass-spring systems and for the base code to solvers on the CPU and GPU. I thank Eduardo Pons by illustrations developed for this work and for the papers submitted, Bárbara Bellaver for the provision in the development of videos to the papers of this work, Leandro Fernandes by the partnership in the submission of one of the papers, and André Spritzer by English revisions.

I thank the professors Carla Freitas, João Comba, Luciana Nedel and Manuel Oliveira for their mastery in leading the Computer Graphics Group at UFRGS, and for their willingness to assist the students in the program. Thanks to professors Paulo Rodacki, Jomi Hübner and Dalton Solano dos Reis, from the Regional University of Blumenau (FURB), for the letters of recommendation. I also thank my colleagues in the Computer Graphics Group, that always propitiated a creative and relaxed environment in the labs, important for the development of this work and my intellectual growth.

Thanks to my parents José Cildo Kuhn and Edir Roslindo Kuhn for my education and freedom of choice, my brothers Giselle Roslindo Kuhn and Vitor Roslindo Kuhn by affection even in the distance, and my family and all friends who understood my absence in the last festive events. I also thank my neighbor Márcia Moares and her family, with all the fundamental support and friendship during my stay in Porto Alegre. Thanks to Triangle Fans, the best soccer team in the world, okay, in the CG world at UFRGS, for the glorious moments in the field, and I also thank Polar beer that always refreshed my moments in Porto Alegre.

I thank CAPES Brazil by partial financial support for this work and a special thanks to my friend Vitor Pamplona for financial help throughout my stay in Porto Alegre.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

This thesis presents an efficient and automatic image-recoloring technique for dichromats that highlights important visual details that would otherwise be unnoticed by these individuals. While previous techniques approach this problem by potentially changing all colors of the original image, causing their results to look unnatural both to dichromats and to normal-vision observers, the proposed approach preserves, as much as possible, the naturalness of the original colors. The technique described in this thesis is about three orders of magnitude faster than previous approaches. This work also presents an extension to our method that exaggerates the color contrast in the recolored images, which might be useful for scientific visualization and analysis of charts and maps.

Another contribution of this thesis is an efficient contrast-enhancement algorithm for color-to-grayscale image conversion that uses both luminance and chrominance information. This algorithm is also about three orders of magnitude faster than previous optimization-based methods, while providing some guarantees on important image properties. More specifically, the proposed approach preserves gray values present in the color image, ensures global color consistency, and locally enforces luminance consistency. A third contribution of this thesis is an error metric for evaluating the quality of color-to-grayscale transformations.

Keywords: Color Reduction, Color-Contrast Enhancement, Color-to-Grayscale Mapping, Color Vision Deficiency, Dichromacy, Error Metric, Image Processing, Monochromacy, Recoloring Algorithm.

Recoloração de Imagens para Portadores de Deficiência na Percepção de Cores

RESUMO

Esta dissertação apresenta um método eficiente e automático de recoloração de imagens para dicromatas que destaca detalhes visuais importantes que poderiam passar despercebidos por estes indivíduos. Enquanto as técnicas anteriores abordam este problema com a possibilidade de alterar todas as cores da imagem original, resultando assim em imagens com aparência não natural tanto para os dicromatas quanto para os indivíduos com visão normal, a técnica proposta preserva, na medida do possível, a naturalidade das cores da imagem original. A técnica é aproximadamente três ordens de magnitude mais rápida que as técnicas anteriores. Este trabalho também apresenta uma extensão para a técnica de recoloração que exagera o contraste de cores na imagem recolorida, podendo ser útil em aplicações de visualização científica e análise de gráficos e mapas.

Outra contribuição deste trabalho é um método eficiente para realce de contrastes durante a conversão de imagens coloridas para tons de cinza que usa tanto as informações de luminância e crominância durante este processo. A técnica proposta é aproximadamente três ordens de magnitude mais rápida que as técnicas anteriores baseadas em otimização, além de garantir algumas propriedades importantes da imagem. Mais especificamente, a técnica apresentada preserva os tons de cinza presentes na imagem original, assegura a consistência global de cores e garante consistência local de luminância. Uma terceira contribuição desta dissertação é uma métrica de erro para avaliar a qualidade dos algoritmos de conversão de imagens coloridas para tons de cinza.

Palavras-chave: Algoritmo de Recoloração, Deficiência Visual de Cores, Dicromatismo, Mapeamento de Cores para Tons de Cinza, Métrica de Erro, Monocromatismo, Processamento de Imagem, Realce de Contraste, Redução de Cores.

1 INTRODUCTION

Color vision deficiency (CVD) is a genetic condition found in approximately 4% to 8% of the male population, and in about 0.4% of the female population around the world (SHARPE et al., 1999), being more prevalent among caucasians. According to the estimates of the U.S. Census Bereau for the world population, we can predict that approximately 200,000,000 (two hundred million) people suffer from some kind of color vision deficiency. Human color perception is determined by a set of photoreceptors (cones) in the retina. Once stimulated, they send some signals to the brain, which are interpreted as color sensation (WANDELL, 1995). Individuals with normal color vision present three kinds of cones called *red*, *green*, and *blue*, which differ from each other by having photopigments that are sensitive to the low, medium, and high frequencies of the visible electromagnetic spectrum, respectively. Thus, individuals with normal color vision are called *trichromats*. Except when caused by trauma, anomalies in color vision perception are caused by some changes in these photoreceptors, or by the absence of some kind of cone. Thus, there are no known treatments of surgical procedures capable of reverting such a condition.

Figure 1.1: On the left, the visible spectrum as perceived by trichromats and dichromats. In the middle, a scene as observed by a subject with normal color vision. On the right, the same scene as perceived by a subject lacking green-cones (deuteranope). Note how difficult it is for this subject to distinguish the colors associated to the fruits.

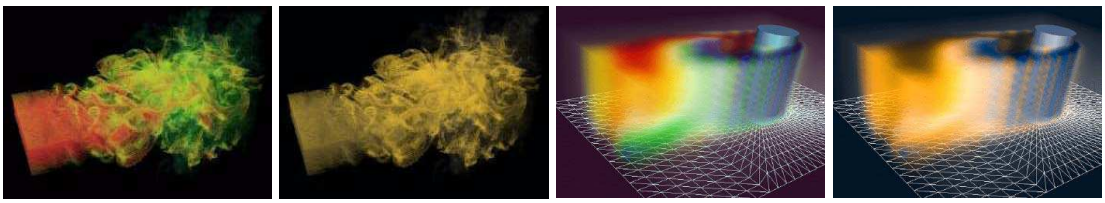
Changes in the cones' photopigments are caused by natural variations of some proteins, causing them to become more sensitive to a different band of the visible spectrum, when compared to a normal vision person (SHARPE et al., 1999). Such individuals are called *anomalous trichromats*. In case one kind of cone is missing, the subjects are called *dichromats*, and can be further classified as *protanopes*, *deuteranopes*, and *tritanopes*, depending whether the missing cones are red, green, or blue, respectively. A much rarer condition is characterized by individuals having a single or no kind of cones, who are called *monochromats*. Figure 1.1 shows the visible electromagnetic spectrum as perceived by tichromats and dichromats, and compares a scene as perceived by an individual with

normal color vision and a subject lacking green-cones (deuteranope). Although there are no reliable statistics for the distribution of the various classes of color vision deficiencies among all ethnic groups, these numbers are available for the caucasian population and are shown in Table 1.1 (RIGDEN, 1999).

Classification	**Incidence (%)**	
	Men	**Women**
Anomalous trichromacy	**5.9**	**0.37**
Protanomaly (red-cones defect)	1.0	0.02
Deuteranomaly (green-cones defect)	4.9	0.35
Tritanomaly (blue-cones defect)	0.0001	0.0001
Dichromacy	**2.1**	**0.03**
Protanopia (red-cones absent)	1.0	0.02
Deuteranopia (green-cones absent)	1.1	0.01
Tritanopia (blue-cones absent)	0.001	0.001
Monochromacy	**0.003**	**0.00001**

Table 1.1: Classification of color vision deficiencies and the respective incidence in the caucasian population (RIGDEN, 1999).

Color vision deficiency tends to impose several limitations, specially for dichromats and monochromats. Children often feel frustrated by not being able to perform color-related tasks (HEATH, 1974), and adults tend face difficulties to perform some daily activities. Figure 1.2 shows some images of recent works in the scientific visualization field, and their respective images simulating the dichromat's perception. Note how difficult it is for the dichromats to distinguish the colors used to represent the datasets, and to reach a better understanding of the data meaning.



(a) (b) (c) (d)

Figure 1.2: Scientific visualization examples: (a) Visualization of a flame simulation and (b) how this image is perceived by a subject lacking green-cones (deuteranope). (c) Simulation of a fluid dynamic and (d) how this simulation is perceived by a protanope (*i.e.*, a subject lacking red-cones).

Another important segment that permeates our daily life and also ignores the limitations of CVDs is the industry of digital entertainment. Despite the respectable 32.6 billion-dollar billing obtained in 2005 by the gaming industry (consoles and computer) and the expectation of doubling this amount by 2011 (ABI Research, 2006), this segment leaves out of its potential market a significant part of population consisting of color-vision deficients. Figure 1.3 illustrates a common situation faced by CVDs dealing video games and digital media in general. The image on the left presents a computer game scene showing the colors of opponent soldiers. On the right, one sees the same image as perceived by deuteranopes. Note that the colors are essentially indistinguishable.

Figure 1.3: Opponent soldiers identified by the armor color in the Quake 4 (Id Software, Inc, 2006). On the left, the image as perceived by subjects with normal color vision. Note how the colors of the soldiers' armors are almost indistinguishable to these individuals. The image on the right simulates the perception of subjects lacking green-cones (deuteranopes).

Recently, several techniques have been proposed to recolor images highlighting visual details missed by dichromats (ICHIKAWA et al., 2004; WAKITA; SHIMAMURA, 2005; RASCHE; GEIST; WESTALL, 2005a,b; JEFFERSON; HARVEY, 2006). Although these techniques use different strategies, they all approach the problem by potentially changing all colors of the original image. In consequence, their results tend to look unnatural both to dichromats and to normal-vision observers. Moreover, they tend to present high computational costs, not scaling well with the number of colors and the size of the input images. This thesis presents an efficient and automatic image-recoloring technique for dichromats that preserves, as much as possible, the naturalness of the original colors.

Despite the very small incidence of monochromats in the world population, color-to-grayscale is, nevertheless, an important subject. Due to economic reasons, the printing of documents and books is still primarily done in "black-and-white", causing the included photographs and illustrations to be printed using shades of gray. Since the standard color-to-grayscale conversion algorithm consists of computing the luminance of the original image, all chrominance information is lost in the process. As a result, clearly distinguishable regions containing isoluminant colors will be mapped to a single gray shade (Figure 1.4). As pointed out by Grundland and Dodgson (GRUNDLAND; DODGSON, 2007), a similar situation happens with some legacy pattern recognition algorithms and systems that have been designed to operate on luminance information only. By completely ignoring chrominance, such methods cannot take advantage of a rich source of information.

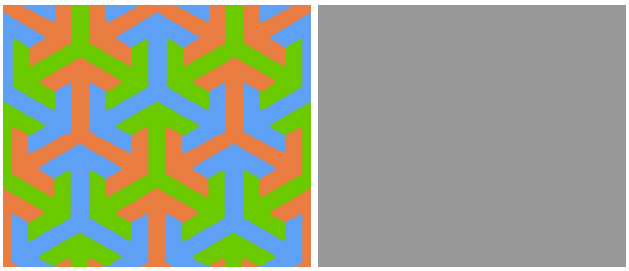


Figure 1.4: Color-to-Grayscale mapping. On the left, isoluminant color image. On the right, grayscale version of the image on the left obtained using the standard color-to-grayscale conversion algorithm.

In order to address these limitations, a few techniques have been recently proposed to convert color images into grayscale ones with enhanced contrast by taking both luminance and chrominance into account (GOOCH et al., 2005a; GRUNDLAND; DODGSON, 2007; NEUMANN; CADIK; NEMCSICS, 2007; RASCHE; GEIST; WESTALL, 2005b). The most popular of these techniques (GOOCH et al., 2005a; RASCHE; GEIST; WESTALL, 2005b) are based on the optimization of objective functions. While these two methods produce good results in general, they present high computational costs, not scaling well with the number of pixels in the image. Moreover, they do not preserve the gray values present in the original image. Grayscale preservation is a very desirable feature and is satisfied by the traditional techniques that perform color-to-grayscale conversion using luminance only. This thesis presents an efficient approach for contrast enhancement during color-to-grayscale conversion that addresses these limitations.

1.1 Thesis Contributions

The main contributions of this thesis include:

1. A new efficient and automatic image-recoloring technique for dichromats that preserves, as much as possible, the naturalness of the original colors (Section 4.1). An extension of this technique that exaggerates color contrast and might be useful for visualization of scientific data as well as maps and charts (Section 4.2);

2. A new efficient contrast-enhancement algorithm for color-to-grayscale image conversion that uses both luminance and chrominance information (Section 5.1);

3. A new contrast error metric for evaluating the quality of color-to-gray transformations (Section 5.2).

1.2 Structure of the Thesis

Chapter 2 discusses some related work to ours. In particular, it covers the state-of-the-art techniques in terms of image recoloring for dichromats.It also reviews the current approaches for color-to-grayscale conversion. Chapter 3 reviews the basic concepts of mass-spring systems, which provide the background for understanding the techniques presented in this thesis. Chapter 4 presents the details of the proposed image-recoloring technique for dichromats, analyzes some of its fundamental properties and guarantees, introduces an extension for exaggerating color contrast, and presents various examples illustrating the obtained results. Chapter 5 describes the details of the proposed color-to-grayscale technique, introduces a new perceptual error metric for evaluating the quality of color-to-grayscale transformations, and discusses some results. Finally, Chapter 6 summarizes the work presented in this thesis and concludes with directions for further investigation.

2 RELATED WORK

There is significant amount of work in the literature that attempts to address the problems of image recoloring for color-vision deficients and color-to-grayscale conversion. This chapter begins covering some techniques to simulate the perception of dichromats, then it discusses some recent image-recoloring techniques for these subjects. Finally, it covers several approaches of color-to-grayscale conversion. For both cases (*i.e.*, image recoloring and color-to-grayscale mapping) the proposed techniques are compared to the state-of-the-art ones.

2.1 Simulation of Dichromat's Perception

The simulation of dichromat's perception allows individuals with normal color vision to experience how these subjects perceive colors. Such simulations became possible after some reports in the medical literature (JUDD, 1948; SLOAN; WOLLACH, 1948) about unilateral dichromats (*i.e.*, individuals with one dichromatic eye, but with normal vision in the other eye). These reports account for the fact that two spectral colors and neutral colors are perceived as equals by both eyes. The spectral colors are blue and yellow for protanopes and deuteranopes, while for tritanopes these colors are cyan and red.

Using this information, some researches have proposed techniques to simulate the colors appearance for dichromats (MEYER; GREENBERG, 1988; BRETTEL; VIéNOT; MOLLON, 1997; WALRAVEN; ALFERDINCK, 1997; VIéNOT; BRETTEL, 2000). In this thesis, all simulations were performed using the algorithm described by Brettel et al.'s (BRETTEL; VIéNOT; MOLLON, 1997), the most referenced technique to simulate the dichromat's perceptions.

Brettel et al. use two half-planes in the LMS color space to represent the color gamut perceived by dichromats. The half-planes were based on the reports of unilateral dichromats in the medical literature. Each class of dichromacy has one type of missing cone, and they confuse colors that fall along lines parallel to the axis that represent the missing cone. Brettel et al. assumed the intersection between such lines and the half-planes as the colors perceived by dichromats. Figure 2.1 illustrates a geometric representation of Brettel et al.'s algorithm to simulate the perception of each class of dichromacy.

Although these simulation techniques allow individuals with normal color vision to appreciate the perception of dichromats, as we noted in the examples of Figures 1.1 to 1.3, they do not help to minimize the limitation of these individuals to perceive the contrast between the colors.

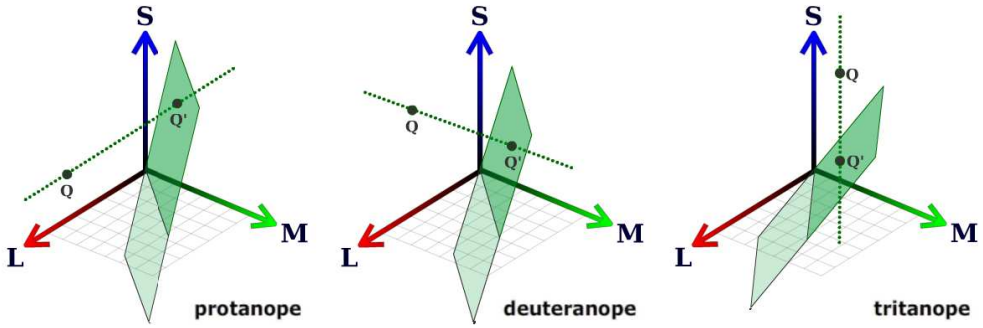


Figure 2.1: Geometric representation of Brettel et al.'s algorithm to simulate perception of dichromats. The green represent the reduced color gamut of dichromats. The orthographic projection of color Q onto the reduced surface gives the color Q' perceived by dichromats. Figure adapted from (BRETTEL; VIéNOT; MOLLON, 1997).

2.2 Recoloring Techniques for Dichromats

The image-recoloring techniques for dichromats aim at minimizing the constraints faced by these subjects in the perception of colors. Basically, the colors of the resulting image are changed to allow dichromats to perceive color contrast in portions of images where they originally missed it. Recoloring techniques can be broadly classified as *user-assisted* and *automatic* techniques.

2.2.1 User-Assisted Recoloring Techniques

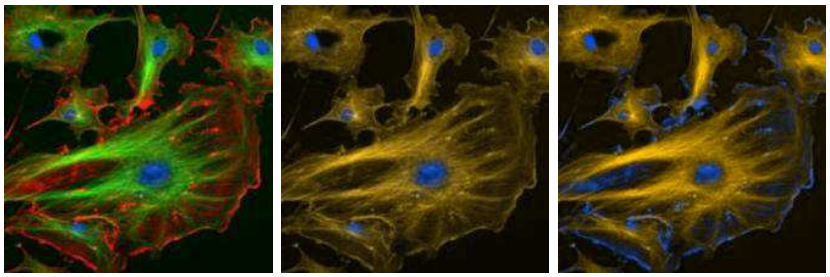
Daltonize (DOUGHERTY; WADE, 2002) is a web application for recoloring images for protanopes and deuteranopes (*i.e.*, subjects also known as red-green colorblind). The technique splits the image into luminance, red-green, and blue-yellow channels. It requires three user-provided parameters: *red-green stretch factor*, used to increase the contrast in the red-green channel; *luminance projection scale*, that tells how much of red-green channel is projected into luminance channel; and *blue-yellow projection scale*, that informs how much of red-green channel is projected into blue-yellow channel.

Fidaner et al. (FIDANER; LIN; OZGUVEN, 2005) also proposed a technique that projects information from one channel into other channels, like in the Daltonize approach. They worked with the differences between the original images and the simulated images for the respective dichromacy, that corresponds to the information lost by dichromats. They required a 3×3 matrix, defined by the user, that specifies how the differences between the original and the simulated dichromatic images should be accumulated to the channels of the original image.

Working on the HSL color space, Iaccarino et al. (IACCARINO et al., 2006) modulate the original image colors using six user-provided parameters. Pixels with hue (H) close to green color are modulated by the first three parameters, while pixels with hue close to red are modulated by the other three parameters.

The quality of the results obtained with user-assisted recoloring techniques is highly dependent on the user-provided parameters. Essentially, a trial-and-error strategy is required to choose the best set of parameters for each individual image. Another limitation is that the presented user-assisted techniques are based on point operations, hence they may introduce new color confusions. Figure 2.2 illustrated this situation. The red and blue colors in (a) have become indistinguishable by a deuteranope in (c) after (a) has been

recolored using Fidaner et al.'s technique.



(a) (b) (c)

Figure 2.2: Example showing a limitation of user-assisted techniques based on point operations. (a) Color image as perceived by a subject with normal color vision. (b) Same image as perceived by a subject lacking green-cones (deuteranope). (c) Simulation of deuteranope's perception after recoloring the image (a) using Fidaner et al.'s technique. Note that red and blue colors in (a) have become indistinguishable in (c).

2.2.2 Automatic Recoloring Techniques

Ichikawa et al. (ICHIKAWA et al., 2003) used an objective function to recolor web pages for anomalous trichromats. The objective function tries to preserve the color differences between all pairs of colors as perceived by trichromats in the reduced anomalous trichromats' gamut. A genetic algorithm was used to minimize the objective function. Note, however, that this problem is relatively simpler, as both groups are trichromats and no reduction in color space dimension is required. Ichikawa et al. (ICHIKAWA et al., 2004) extended their previous technique for use on color images, but they did not consider the preservation of color naturalness (*i.e.*, preservation of colors that are perceived as similar by both trichromats and anomalous trichromats).

Wakita and Shimamura (WAKITA; SHIMAMURA, 2005) proposed a technique to recolor documents (*e.g.*, web pages, charts, maps) for dichromats using three objective functions aiming, respectively, at: (i) color contrast preservation, (ii) maximum color contrast enforcing, and (iii) color naturalness preservation. However, in their technique, the colors for which naturalness should be preserved must be specified by the user. The three objective functions are then combined by weighting user-specified parameters and optimized using simulated annealing. They report that documents with more than 10 colors could take several seconds to be optimized (no information about the specs of the hardware used to perform this time estimate were provided).

Jefferson and Harvey (JEFFERSON; HARVEY, 2006) select a set of key colors by sampling the difference histogram (Figure 2.3 e) between the trichromat's color histogram (Figure 2.3 c) and dichromat's color histogram (Figure 2.3 d). They use four objective functions to preserve brightness, color contrast, colors in the available gamut, and color naturalness of the selected key colors. Again, the user must specify the set of colors whose naturalness should be preserved. They optimize the combined objective functions using a method of preconditioned conjugate gradients. They report times of several minutes to optimize a set of 25 key colors on a P4 2.0 GHz using a Matlab implementation.

Rasche et al. (RASCHE; GEIST; WESTALL, 2005a) proposed an automatic recoloring technique for dichromats as an optimization that tries to preserve the perceptual color differences between all pairs of colors using an affine transformation. Such trans-

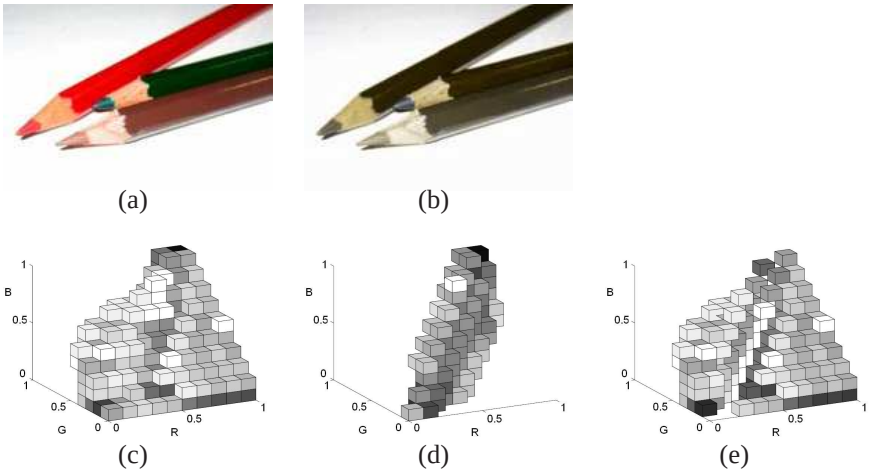


Figure 2.3: (a) Color image and (b) simulation of protanope's view for image (a). (c) and (d) are the 3D color histograms for the images shown in (a) and (b), respectively. (e) Difference histogram obtained by subtracting (d) from (c). The RGB color space is discretized into 1,000 equal volume cubes. The gray shade of a cube is proportional to the number of pixels in the image with the corresponding RGB values. Figures extracted from (JEFFERSON; HARVEY, 2006).

formation, however, does not capture color variations along many directions and does not ensure that the mapped colors are all within the available gamut. Rasche et al. (RASCHE; GEIST; WESTALL, 2005b) addressed these limitations using a constrained multivariate optimization procedure applied to a reduced set of quantized color, which are in turn used to optimize the entire set of colors. The authors did not consider the problem of naturalness preservation and the technique can arbitrarily change the colors of the original images (Figure 2.4 c). Moreover, the algorithm does not scale well with the number of quantized colors and the size of the input images.

The image recoloring technique presented in this thesis can optimize hundreds of colors in real time, and can be used to create images that have a much more natural look (Figure 2.4 d). Contrary to all previous automatic techniques, the proposed approach is deterministic, always producing the same result for a given input image.

2.3 Color-to-Grayscale Techniques

Mapping a color image to grayscale is a dimensionality reduction problem. Traditional techniques use projections or weighted sums to map a three dimensional color space to a single dimension (*e.g.*, the luminance value of *XYZ*, *YCbCr*, *L*a*b**, or *HSL* color spaces). These are the common methods implemented in commercial applications, such as Photoshop (BROWN, 2006) and Gimp (JESCHKE, 2002). These approaches, however, do not take into account any chrominance information, mapping isoluminant pixels to the same gray value, as shown in Figure 1.4 (b).

A popular dimensionality reduction technique is Principal Component Analysis (PCA) (DUNTEMAN, 1989). However, as pointed out by (GOOCH et al., 2005a; RASCHE; GEIST; WESTALL, 2005b), since PCA ignores the directions with low variation, small detail can be missed in favor of larger detail.

Grundland and Dogdson (GRUNDLAND; DODGSON, 2007) approach color-to-grayscale

(a) (b) (c) (d)

Figure 2.4: Blue sky and building: (a) Color image. (b) Simulation of deuteranope's view for image (a). (c) and (d) are the results produced by Rasche et al.'s and the proposed techniques, respectively, as seen by deuteranopes. Note that colors of the sky and yellow foliages in (a) were unnecessary changed by Rasche et al.'s approach (c), not preserving the naturalness of such colors. Compare such a result with the one obtained with the proposed technique (d).

problem by first converting the original RGB colors to their YPQ color space, followed by a dimensionality reduction using a technique they called *predominant component analysis*, which is similar to PCA. In order to decrease the computational cost of this analysis, they use a local sampling by a Gaussian pairing of pixels that limits the amount of color differences processed and brings the total cost to convert an $N \times N$ image to $O(N^2 \log(N^2))$. This technique is very fast, but its local analysis may not capture the differences between spatially distant colors and, as a result, it may map clearly distinct colors to the same shade of gray. Figure 2.5 (a) illustrates the USA time zones map using distinct isoluminant colors for each time zone. Note that in the result produced by Grundland and Dogdson's approach, shown in (c), the color contrast between some time zones (*e.g.*, HST and AKST time zones, CST and EST time zones) were not preserved, illustrating the limitation previously described. The grayscale image (d), obtained using our color-to-grayscale approach, successfully mapped the various time zones to distinct shades of gray.

Neumann et al. (NEUMANN; CADIK; NEMCSICS, 2007) presented an empirical color-to-grayscale transformation algorithm based on the Coloroid system (NEMCSICS, 1980). Based on an user-study, they sorted the relative luminance differences between pairs of seven hues, and interpolated between them to obtain the relative luminance differences among all colors. Their algorithm requires the specification of two parameters, and the reported running times are of the order of five to ten seconds per megapixel (hardware specs not informed).

Gooch et al. (GOOCH et al., 2005a) find gray levels that best represent the color difference between all pair of colors by optimizing an objective function. The ordering of the gray levels arising from the original colors with different hues is resolved with a user-provided parameter. The cost to optimize an $N \times N$ image is $O(N^4)$, causing the algorithm to scale poorly with image resolution.

Rasche et al. (RASCHE; GEIST; WESTALL, 2005b) formulated the color-to-grayscale transformation as an optimization problem in which the perceptual color difference be-

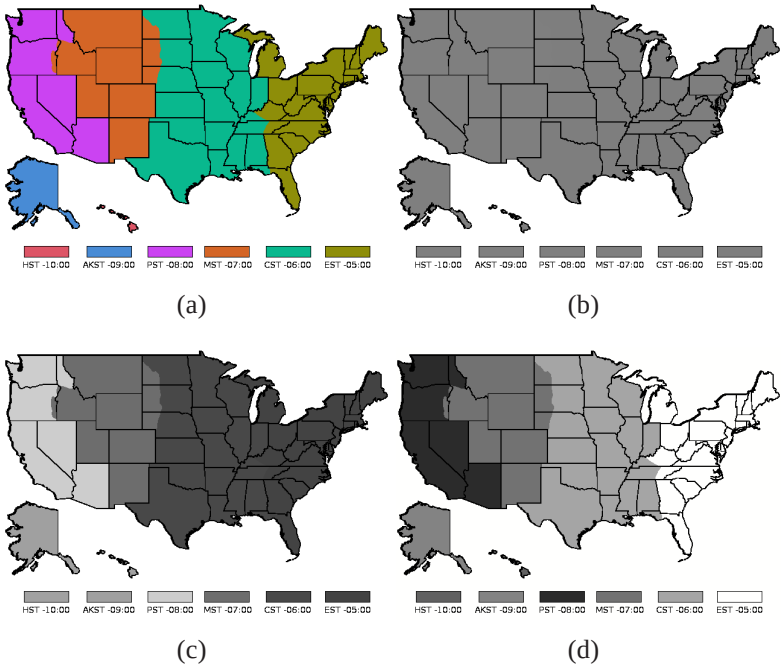


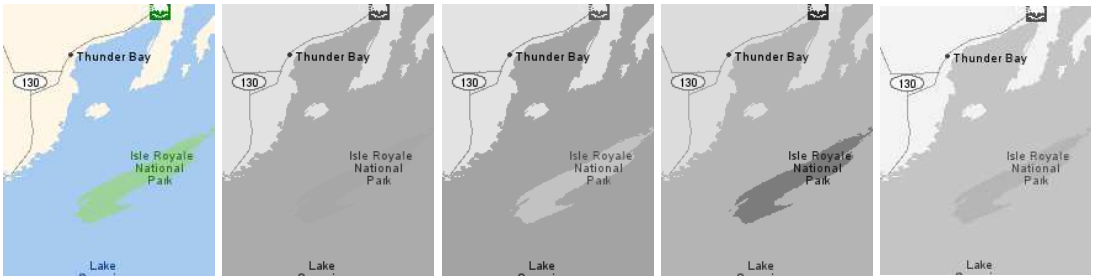
Figure 2.5: USA time zone map: (a) Color image. (b) Luminance image. (c) Grayscale image produced by Grundland and Dogdson's (GRUNDLAND; DODGSON, 2007) method. (d) Grayscale image obtained using the color-to-grayscale approach presented in this thesis. Note in (c) how the color contrast between some spatially distant regions were not preserved by Grundland and Dogdson's approach (*e.g.*, HST and AKST time zones, CST and EST time zones). The grayscale image shown in (d) successfully mapped the various time zones to different gray values.

tween any pair of colors should be proportional to the perceived difference in their corresponding shades of gray. In order to reduce its computation cost, the authors perform the optimization on a reduced set Q of quantized colors, and this result is then used to optimize the gray levels of all pixels in the resulting image. The total cost of the algorithm is $O(\|Q\|^2 + \|Q\|N^2)$. A noticeable feature of their algorithm is that in order to try to help the algorithm to scape local minima, the minimization procedure is initialized using a vector of random values, which causes the algorithm to produce non-deterministic results. This is illustrated in Figure 2.6, which shows the grayscale images produced in three executions of their algorithm. Note that in the result shown in Figure 2.6 (b) the island is barely visible, illustrating a situation in which the optimization got trapped in a local minima. Figure 2.6 (e) shows the result produced by the color-to-grayscale algorithm presented in this thesis.

2.4 Summary

This chapter discussed the most relevant techniques to simulate color perception by dichromats. Although these techniques do not minimize the limitation of these individuals to perceive color differences, they are important because they eliminate the need for the presence of dichromats along the development and tests of the image-recoloring algorithms.

The chapter also presented the state-of-the-art on image-recoloring techniques for



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

Figure 2.6: Effect of the use of random numbers to initialize the algorithm by Rasche et al. (RASCHE; GEIST; WESTALL, 2005b) on the generated grayscale images. (a) Image containing isoluminant colors. (b) to (d) Grayscale images generated by three executions of the Rasche et al.'s algorithm. (e) Grayscale image produced by the color-to-grayscale algorithm presented in this thesis. Note that in (b) the island Isle Royale National Park is barely visible, while our color-to-grayscale technique preserved the contrast in (e).

dichromats. Basically, these techniques aim to minimize the limitations faced by these subjects in the perception of color contrasts. It was discussed two kind of approaches for image recoloring: user-assisted and automatic techniques. The user-assisted recoloring techniques, despite the low computation cost, do not take into account any analysis on the image to choose optimal parameter settings, hence they may introduce new color confusions. On the other hand, automatic recoloring techniques use optimization methods to choose the best set of colors in the dichromatic image, which tends to generate better results than user-assisted techniques. However, the current automatic methods are computationally expensive and not suitable for real-time applications. Contrary to all current automatic methods, the proposed image-recoloring algorithm can optimize hundreds of colors in real time and can produce images with more natural look.

Furthermore, this chapter presented several approaches for color-to-grayscale conversion. These techniques convert color images into grayscale ones with enhanced contrast by taking both luminance and chrominance into account. Unlike current optimization methods, the proposed color-to-grayscale technique is deterministic, can optimize hundreds of colors in real time, and scales well with the size of the input image.

3 MASS-SPRING SYSTEMS

This chapter reviews the basic concepts of mass-spring systems and provides the background for understanding both the image-recoloring and the color-to-grayscale techniques proposed in this thesis, which are cast as optimization problems.

3.1 Definition of Mass-Spring Systems

A mass-spring system consists of a set of particles (nodes) connected by springs that deform in the presence of some external forces, as illustrated in Figure 3.1. When compressed or stretched, the springs apply internal reaction forces in order to maintain their rest length (GEORGII; WESTERMANN, 2005). The system tends to stabilize when the external forces are compensated by opposing internal forces. These features make a mass-spring system a flexible optimization technique that optimizes a set of parameters (*e.g.*, the positions of the particles) that best satisfy some constraints (*e.g.*, the sum of internal and external forces is zero). Modeling a problem as a mass-spring system basically consists of mapping some properties of the problem in hand to the variables of the mass-spring system: (i) particles' positions, (ii) particles' masses, (iii) springs' rest lengths, and (iv) springs' current lengths. For example, an application could map a given property of the problem as the mass of particles in the system. In this case, particles with bigger masses tend to move less. Alternatively (or in addition to this first mapping), the application could restrict the particles to only move along a given axis.

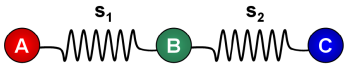


Figure 3.1: Simple mass-spring system with three particles (A, B, and C), and two springs (S_1 and S_2). Figure extracted from (DIETRICH; COMBA; NEDEL, 2006).

Due to its properties, simplicity and low computational complexity, mass-spring systems are used as an optimization tool in many areas, including body deformation and fracture, cloth and hair animation, virtual surgery simulation, interactive entertainment, and fluid animation.

3.2 Dynamic of Mass-Spring Systems

Considering a set of particles connected by springs, mass-spring systems are simulated by assigning some mass to each particle and some rest length to each spring. The entire system must obey Newton's second law:

$$F_i = m_i a_i \tag{3.1}$$

where m_i is the mass of node P_i, a_i is the acceleration caused by force F_i, which is the composition of internal and external forces. Therefore, the force applied to node P_i can be obtained from Hooke's law by summing the tensions of all the springs that connect P_i to its neighbors P_j:

$$F_i = \sum_{j \in N} k_{ij}(1 - \frac{l_{ij}}{l'_{ij}})(p_j - p_i) \tag{3.2}$$

where N is the set of neighbors linked to P_i, l_{ij} and l'_{ij} are, respectively, the rest length and current length of the spring between P_i and P_j, k_{ij} is the stiffness of the spring, and p_i and p_j are the current positions of P_i and P_j, respectively.

Verlet integration (VERLET, 1967) is often used to express the dynamics of each node. This type of integration is frequently used in simulations of small unoriented mass-points, being especially interesting when it is necessary to place constraints on the distances between the points (VERTH; BISHOP, 2004). With a time step Δt, the new position of a node P_i at time $t + \Delta t$ can be computed as:

$$p_i(t + \Delta t) = \frac{F_i(t)}{m_i} + 2p_i(t) - p_i(t - \Delta t) \tag{3.3}$$

Recently, some researchers have demonstrated efficient implementations of mass-spring systems on GPUs (GEORGII; WESTERMANN, 2005; TEJADA; ERTL, 2005; DIETRICH; COMBA; NEDEL, 2006). In each integration step, the forces acting on each mass point P_i are accumulated in a fragment shader, requiring information about the particles' geometry and the system topology, which are usually stored in three textures (Figure 3.2): *geometry texture* storing the particles' position, *neighbors texture* storing the list of neighbors of all mass-points, and *neighborhood texture* serving as a neighborhood header.

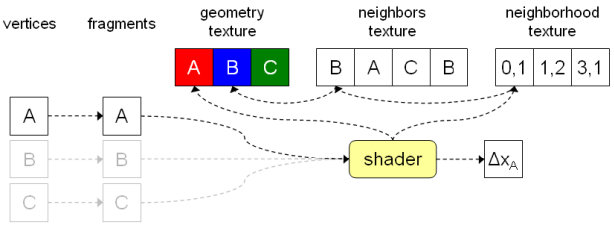


Figure 3.2: Texture representation of the mass-spring system shown in Figure 3.1. Each pixel represents one mass-point, and the pixel shader is able to recover the information of all linked springs from neighborhood and neighbors texture. Figure extracted from (DIETRICH; COMBA; NEDEL, 2006).

The resulting algorithms (TEJADA; ERTL, 2005; DIETRICH; COMBA; NEDEL, 2006) perform in real time even for large systems (*e.g.*, 65K mass-points and 275K

springs). The solutions proposed in this thesis for both image recoloring and color-to-grayscale transformations are cast as optimization problems and modeled as mass-spring systems with every mass point P_i connected to every other mass point P_j by a spring S_{ij}. This fixed and implicitly defined topology lends itself to an efficient GPU implementation, since no topology setup is needed.

3.3 Summary

This chapter detailed the key aspects of mass-spring systems, an optimization tool with low computational cost that can be efficient implemented both on CPU and on GPU. The understanding of the mass-spring's dynamic is fundamental to the comprehension of both the image-recoloring and the color-to-grayscale techniques proposed in this thesis, as they are approached as optimization problems and modeled as mass-spring systems.

4 THE RECOLORING ALGORITHM FOR DICHROMATS

This chapter presents our image-recoloring technique for dichromats. It also analyzes some fundamental properties and guarantees of the algorithm, and introduces an extension to the proposed method that exaggerates the color contrast in the result image. Along the chapter, several examples are used to illustrate the results produced by the proposed technique.

4.1 The Algorithm

Our image-recoloring technique for dichromats was designed to achieve the following goals:

- *Real-time performance*;
- *Color naturalness preservation* - It guarantees that colors of the original images will be preserved, as much as possible, in the resulting images;
- *Global color consistency* - It ensures that all pixels with the same color in the original image will be mapped to the same color in the recolored image;
- *Luminance constancy* - It ensures that the original luminance information of each input pixel will be preserved in the output one;
- *Local chrominance consistency* - It guarantees that chrominance order of colors inside the same cluster will have preserved in the recolored image;
- *Color contrast preservation* - It tries to preserve the color difference between pairs of colors from the original image in the recolored one.

The proposed algorithm uses a mass-spring system to optimize the colors in the input image to enhance color contrast for dichromats. As presented in Section 2.1, the color gamut of each class of dichromacy can be represented by two half-planes in the LMS color space (BRETTEL; VIéNOT; MOLLON, 1997), which can be satisfactorily approximated by a single plane passing through the luminance axis (VIéNOT; BRETTEL; MOLLON, 1999). For each class of dichromacy, we map its color gamut to the approximately perceptually-uniform CIE L*a*b* color space and use least-squares to obtain a plane that contains the luminance axis (*i.e.*, L*) and best represents the corresponding gamut (Figure 4.1). This is similar to what has been described by Rasche et al. (RASCHE; GEIST; WESTALL, 2005b), who suggested the use a single plane for both protanopes

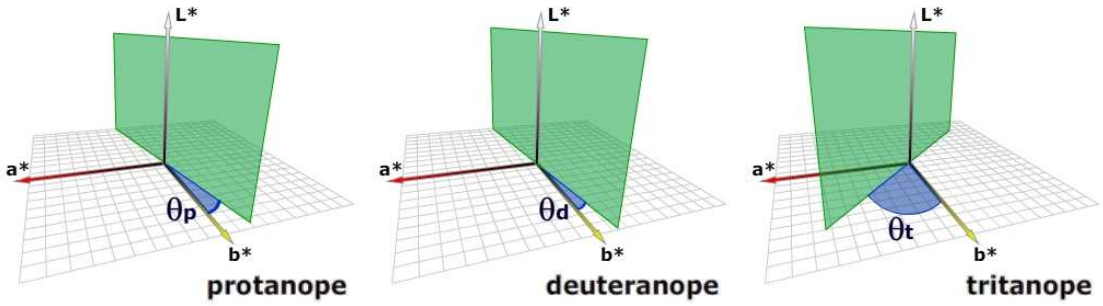


Figure 4.1: Approximating each dichromatic color gamut by a plane in the CIE L*a*b* color space. $\Theta_p = -11.48°$, $\Theta_d = -8.11°$, and $\Theta_t = 46.37°$.

and deuteranopes as a further simplification. Based on our least-squares fitting, the angles between the recovered planes and the L*b* plane are $\Theta_p = -11.48°$, $\Theta_d = -8.11°$, and $\Theta_t = 46.37°$, for protanopes, deuteranopes, and tritanopes, respectively (Figure 4.1). These angles are used to align their corresponding planes to the L*b* plane, reducing the optimization to 1D along the b* axis (the luminance values are kept unchanged). After the optimization, the new colors are obtained by rotating the plane back.

Our algorithm has three main steps: (i) *image quantization*, (ii) *mass-spring optimization* of the quantized colors, and (iii) *reconstruction* of the final colors from the optimized ones. The first step consists in obtaining a set Q of quantized colors from the set of all colors C in the input image I. This can be performed using any color quantization technique (GONZALEZ; WOODS, 2007), such as uniform quantization, k-means, minimum variance, median cut or color palettes.

4.1.1 Modeling the Problem as a Mass-Spring System

Working in the CIE L*a*b* color space, each quantized color $\vec{q_i} \in Q$ is associated to a particle P_i with mass m_i. The position $\vec{p_i}$ of the particle P_i is initialized with the coordinates of the perceived color by the dichromat after the gamut plane rotation:

$$\vec{p_i} = M_\Theta \; D(\vec{q_i}) \tag{4.1}$$

where D is the function that simulates the dichromat view (BRETTEL; VIéNOT; MOLLON, 1997), and M_Θ is a rotation matrix that aligns the gamut plane of the dichromat with the L*b* plane. We connect each pair of particles P_i and P_j with a spring S_{ij} with elasticity coefficient $k_{ij} = 1$ (Equation 3.2), and rest length $l_{ij} = \|\vec{q_i} - \vec{q_j}\|$, the (Euclidean) distance between colors $\vec{q_i}$ and $\vec{q_j}$ in the L*a*b* space. Note that $\vec{q_i}$ and $\vec{q_j}$ are the quantized colors as perceived by a trichromat.

At each optimization step, we update the positions $\vec{p_i}$ and $\vec{p_j}$ of particles P_i and P_j and compute S_{ij}'s current length as $l'_{ij} = \|\vec{p_i} - \vec{p_j}\|$. Given the restoring forces of the springs, the system will try to converge to a configuration for which $l'_{ij} = l_{ij}$ for all S_{ij}. Thus, after stabilization or a maximum number of iterations has been reached, the perceptual distances between all pairs of new colors/positions $(\vec{p_i}, \vec{p_j})$ will have approximately the same perceptual distances as their corresponding pairs of quantized colors $(\vec{q_i}, \vec{q_j})$ from Q. The final set T of optimized colors $\vec{t_i}$ is obtained by applying, to each resulting color $\vec{p_i}$, the inverse rotation used in Equation 4.1:

$$\vec{t_i} = M_\Theta^{-1} \; \vec{p_i} \tag{4.2}$$

In order to enforce color naturalness preservation, we define the mass m_i of each particle P_i as the reciprocal of the perceptual distance (in the L*a*b* space), between $\vec{q_i}$ and $D(\vec{q_i})$:

$$m_i = \frac{1}{\|\vec{q_i} - D(\vec{q_i})\|} \tag{4.3}$$

Equation 4.3 guarantees that *any color perceived similarly by both trichromats and dichromats will have larges masses, causing their corresponding particles to move less*. If trichromats and dichromats perceive $\vec{q_i}$ exactly the same way (*e.g.*, achromatic colors), the particle would have infinite mass. In this case, we simply set the forces acting on the particle to zero (*i.e.*, $F_i = 0$ in Equation 3.1).

4.1.2 Dealing with Local Minima

Like other optimization techniques, mass-spring systems are prone to local minima. Figure 4.2 (b) depicts the problem with a configuration obtained right after the quantized colors $\vec{q_i}$ have been rotated: $q\vec{r}_i = M_\Theta \ \vec{q_i}$. In this example, particles P_1 and P_2 have large masses (m_1 is infinite) since they are perceived as having, respectively, the same and very similar colors by trichromats and dichromats. The springs $(S_{13}, S_{14}, S_{23}, S_{24})$ connecting P_1 and P_2 to P_3 and P_4 apply forces that constrain P_3 and P_4 from moving to the other half of the b* axis (Figure 4.2 b). Figure 4.3 illustrates this situation, where the resulting optimized image (c) does not represent any significant improvement over the original image perceived by the dichromat (b).

Once the plane that approximates the dichromat's gamut has been aligned to the L*b* plane, pairs of ambiguous colors with considerable changes in chrominance will have their a* color coordinates with opposite signs (*e.g.*, the red and green colors in Figure 4.2 b). We use this observation and the topology of our mass spring system to deal with local minima using the following heuristic: *we switch the sign of the b* color coordinate of all rotated quantized colors whose a* coordinates are positive and whose perceptual distance between the color itself and how it is perceived by the class of dichromacy is bigger than some threshold* ϵ (Equation 4.4).

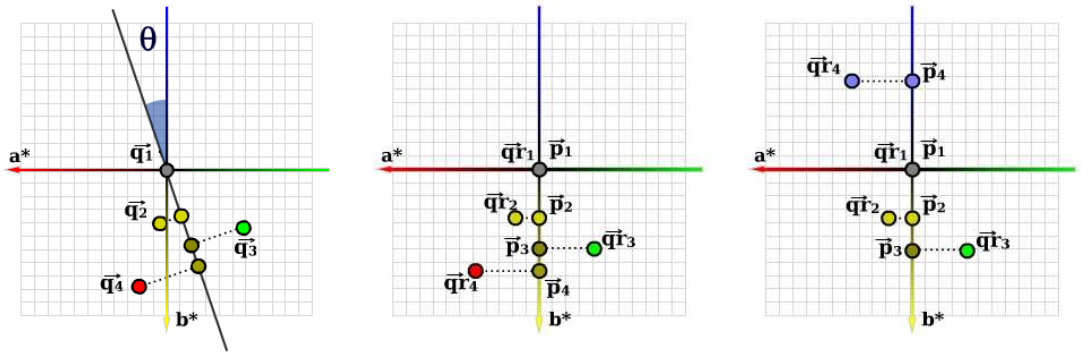
Although at first this heuristic might look too naive because one would just be replacing some ambiguities with another ones, it actually has some rational: (i) it avoids the ambiguities among some colors found in the original image and (ii) although there is the possibility of introducing new ambiguities, as we switch the sign of the b* coordinate for some colors, we are also compressing and stretching their associated springs, adding to the system a lot potential energy that will contribute to drive the solution.

Even though such a heuristic cannot guarantee that the system will never run into a local minima, in practice, after having recolored a huge number of images, we have not encountered yet an example in which our system was trapped by a local minimum. Figure 4.2 (c) illustrates the configuration obtained by applying the described heuristic to the example shown in Figure 4.2 (b). Figure 4.3 (d) shows the corresponding result of applying this heuristic to the example shown in Figure 4.3 (a).

$$p_i^{b^*} = \begin{cases} -p_i^{b^*} & \text{if } (qr_i^{a^*} > 0) \text{ and } (\|\vec{q_i} - D(\vec{q_i})\| > \epsilon) \\ p_i^{b^*} & \text{otherwise} \end{cases} \tag{4.4}$$

In Equation 4.4, $p_i^{b^*}$ is the b* coordinate of color $\vec{p_i}$ and $qr_i^{a^*}$ is the a* coordinate of the rotated color $q\vec{r}_i$. The threshold ϵ enforces that colors that are perceptually similar to both dichromats and trichromats should not have the signs of their b* coordinates switched, in

order to preserve the naturalness of the image. In the example illustrated by Figures 4.2 and 4.3, the rotated quantized color $q\vec{r}_2$ preserved its b* coordinate. According to our experience, $\epsilon = 15$ tends to produce good results in general and was used for all images shown in this work.



(a) (b) (c)

Figure 4.2: Dealing with local minima. (a) A set of quantized colors. (b) A configuration, obtained right after plane rotation, that leads to a local minimum: since P_1 cannot move at all (it is an achromatic color) and P_2 can only move a little bit, they will constrain the optimization to the yellow portion of b* axis, leading the solution to a local minimum. (c) By switching the sign of the b* coordinate of $q\vec{r}_4$, the system escapes the local minimum.

(a) (b) (c) (d)

Figure 4.3: Example of the use of a heuristic to avoid local minima: (a) Reference image. (b) Simulation of a deuteranope's view of the image in (a). (c) Simulation of a deuteranope's view after recoloring the reference image using the mass-spring optimization without the heuristic described by Equation 4.4. (d) Result produced by our technique with the use of Equation 4.4.

4.1.3 Computing the Final Color Values

The last step of the algorithm consists in obtaining the color values for all pixels of the resulting image from the set of optimized colors $\vec{t_k}, k \in \{1, .., \|T\|\}$, where $\|T\|$ is the cardinality of T. For this task, we have developed two distinct interpolation solutions: *per-cluster interpolation* and *per-pixel interpolation*.

4.1.3.1 *Per-cluster interpolation*

Let $C_k \subset C$ be a cluster formed by all colors in C that are represented by the optimized color $\vec{t_k}$. The final value $\vec{t_m^k}$ associated to the m-th color $\vec{c_m^k} \in C_k$ is then obtained as

$$\vec{t_m^k} = \vec{t_k} + (\vec{d}.L^*, r_k\ \vec{d}.a^*, r_k\ \vec{d}.b^*) \tag{4.5}$$

where $\vec{d}.L^*$, $\vec{d}.a^*$, and $\vec{d}.b^*$ are respectively the L*, a*, and b* coordinates of the difference vector $\vec{d} = (\vec{c_m^k} - \vec{q_k})$. $\vec{q_k} \in Q$ is the quantized color associated to the optimized color $\vec{t_k}$. Equation 4.5 guarantees that the final color $\vec{t_m^k}$ has the same luminance as $\vec{c_m^k}$. r_k is an interpolation of ratios that indicates how close the transformed value $\vec{t_k}$ is to the optimal solution. This interpolation is guided by Shepard's method (SHEPARD, 1968), a standard technique for distance-weighted interpolation:

$$r_k = \frac{\sum_{i=1}^{\|T\|} w_{ki} \frac{\|\vec{t_k} - \vec{t_i}\|}{\|\vec{q_k} - \vec{q_i}\|}}{\sum_{i=1}^{\|T\|} w_{ki}}, for\ i \neq k \tag{4.6}$$

where $w_{ki} = 1/\|\vec{q_k} - \vec{q_i}\|^2$ is the distance-weighted term suggested by Shepard (SHEPARD, 1968). This transformation also guarantees local chrominance consistency, as all chrominance values inside a cluster C_k are computed relatively to the optimized color $\vec{t_k}$. Equation 4.5 can be efficiently implemented both on a CPU and on a GPU. On the CPU the cost to compute all cluster ratios using Equation 4.6 is $O(\|Q\|^2)$ for a set Q of quantized colors, and the cost to interpolate each pixel using Equation 4.5 given an image with $N \times N$ pixels is $O(N^2)$. Thus, the total cost of computing the final image colors from the set T of optimized colors is $O(\|Q\|^2 + N^2)$.

Since the final colors are computed by optimizing the set of quantized colors Q, the quality of the results depends directly on the quality of the quantization algorithm used. According to our experience, the transformation expressed by Equation 4.5 produces excellent results in combination with k-means. Unfortunately, k-means is not very fast. Table 4.1 compares the times of uniform quantization and k-means for various image resolutions and number of clusters. In the case of uniform quantization, we discretize the RGB space using a uniform $10 \times 10 \times 10$ grid. The quantized color is given by the grid color closest to the input one. Uniform quantization is fast, but tends to lead to poorer results when used in combination with the transformation defined by Equation 4.5.

Quantization technique	**Image resolution (pixels)**		
	640 x 480	**800 x 600**	**1024 x 768**
Uniform	0.055	0.087	0.150
K-means 64	2.082	3.338	5.517
K-means 128	3.932	6.029	10.432
K-means 256	7.545	11.972	20.049

Table 4.1: Time (in seconds) to quantize images with various resolutions on a 2.2 GHz PC. For k-means, we used the code available at (ZHANG, 2006).

4.1.3.2 *Per-pixel interpolation*

One can benefit from the speed of uniform quantization by performing a more expensive reconstruction of the final set of colors, achieving results similar to the ones obtained

when using k-means. In this case, the final shading of each pixel is obtained by optimizing it against the set of already optimized colors T. This is modeled by setting up a mass-spring system, and creating springs between the current pixel (treated as a particle initialized with Equation 4.4) and all optimized colors $t_k,\ k\ \in\ [1,..,\|T\|]$. For this refining optimization stage, we force the particles associated to the shades in T to remain stationary by setting the forces acting on them to zero (F_i in Equation 3.1). For each color $\vec{c_m} \in C$, we define its mass as $m_{c_m} = 1/\|\vec{c_m} - D(\vec{c_m})\|$. This way, we allow a color to change by an amount directly proportional to the difference of how it is perceived by trichromats and dichromats. *This mechanism guarantees the color naturalness in the resulting image*.

The cost of this optimization procedure is $O(\|Q\|^2 + \|Q\|N^2)$ for an $N \times N$ image, which can be significantly higher than the mapping defined by the per-cluster interpolation technique ($O(\|Q\|^2 + N^2)$) for large values of $\|Q\|$ or N. However, since the color of each output pixel can be computed independently of each other, the computation can be efficiently implemented in a fragment program. Table 4.2 compares the times for recoloring images with various resolutions using different algorithms. *MS-PC CPU* and *MS-PC GPU* are respectively the CPU and GPU versions of our mass-spring algorithm using per-cluster interpolation to obtain the final colors. *MS-PP GPU* optimizes the final colors using our per-pixel interpolation method as described in the previous paragraph. Table 4.2 shows that in all of its variations, our approach is a few orders of magnitude faster than Rasche et al.'s approach. All images (and execution times) reported for the technique of Rasche et al. (RASCHE; GEIST; WESTALL, 2005b) were obtained using the software available at (RASCHE, 2005).

Recoloring technique	**Image resolution (pixels)**		
	640 x 480	**800 x 600**	**1024 x 768**
Rasche	225.16	349.31	580.49
MS-PC CPU	0.41	0.46	0.54
MS-PC GPU	0.20	0.22	0.26
MS-PP GPU	0.22	0.23	0.27

Table 4.2: Performance of various algorithms on images of different resolutions. Times measured in seconds on a 2.2 GHz PC with 2 GB of memory and on a GeForce 8800 GTX GPU. Quantization times not included. All techniques used a set of 128 quantized color. Mass-spring (MS) optimized the set of quantized colors using 500 iterations. Our per-pixel interpolation version (MS-PP GPU) obtains the final colors using 100 iterations for each pixel in the image.

4.2 Exaggerated Color-Contrast

For applications involving non-natural images (*e.g.*, scientific and information visualization) contrast enhancement is probably more important than preserving color naturalness. This comes from the fact that colors used in such applications tend to be arbitrary, usually having little connection to the viewer's previous experiences in the real world. In scientific visualization, the details presented by the datasets are interactively explored via transfer functions (PFISTER et al., 2001). Until now, transfer function design has largely ignored the limitations of color-vision deficients. Popular color scales usually range from red to green, colors that are hard to be distinguished by most dichromats.

By supporting real-time recoloring of transfer functions for dichromats, our approach can assist color-vision deficient to exploit the benefits of scientific visualization. This kind of assistance can be made with the following changes in our image-recoloring algorithm:

1. Modifying the spring's rest length to exaggerate the contrast between the colors during the optimization process: $l_{ij} = x \, \|\vec{q_i} - \vec{q_j}\|$, where x is a scalar used to exaggerate the perceptual difference between any pair of color $\vec{q_i}$ and $\vec{q_j}$;

2. Defining the mass of particle P_i as: $m_i = 1/\|(a_i^*, b_i^*)\|$, where $\|(a_i^*, b_i^*)\|$ is the distance from color $\vec{q_i}$ to the luminance axis L^*. Thus, less saturated colors present bigger masses and tend to move less, as expected, this preserves the achromatic colors;

3. Initializing the mass-spring system with $\epsilon = 0$ (Equation 4.4), since we do not need to preserve the naturalness of colors.

4.3 Results and Discussion

We have implemented the described algorithms using C++ and GLSL, and used them to recolor a large number of images. The reported times were measured using a 2.2 GHz PC with 2 GB of memory and on a GeForce 8800 GTX with 768 MB of memory. Figures 4.4, 4.5, 4.6, 4.7, and 4.8 compare the results of our technique against Rasche et al.'s approach, and Table 4.3 summarizes the performances of the algorithms. Since Rasche et al.'s technique is not deterministic, for the sake of image comparison, for each example shown in this work, we run their software five times and selected the best result. For these experiments, the input images were quantized using the following algorithms and number of colors: Figure 4.4 - *Flowers* (uniform, 227 colors), Figure 4.5 - *Bell Peppers* (k-means, 127 colors), Figure 4.6 - *Signora in Giardino* (k-means, 128 colors), Figure 4.6 - *Still Life* (uniform, 46 colors), and Figure 4.8 - *Chinese Garden* (k-means, 128 colors).

Image (size)	**Rasche et al. Time**	**MS-PC CPU Time**	**MS-PP GPU Time**
Flowers (839×602)	315.84	0.85	0.16
Bell Peppers (321×481)	114.63	0.27	0.09
Signora in Giardino (357×241)	60.68	0.26	0.08
Still Life (209×333)	47.34	0.08	0.06
Chinese Garden (239×280)	44.06	0.23	0.08

Table 4.3: Performance comparison between our technique and Rasche et al.'s for images of various sizes and different quantization strategies. Time measured in seconds shows that our technique scales well with image size.

Figure 4.4, *Flowers*, has 839×602 pixels and our GPU implementation performed its recolorization in 0.158 seconds. This is $2,000\times$ faster than Rasche et al.'s approach. Our CPU implementation was still $372\times$ faster than Rasche et al.'s for this example.

In the example shown in Figure 4.5, while Rasche et al.'s approach (c) enhanced the contrast among the colors of the peppers, our technique also preserved the color naturalness of the crates, yellow peppers, and other vegetables in the background (d).

In Figure 4.6, Monet's painting *Signora in Giardino* was recolored to enhance contrast for protanopes. In this example, note how difficult it is for these individuals to distinguish

(a) (b)

(c) (d)

Figure 4.4: Images of natural flowers: (a) Photograph. (b) Same image as perceived by protanopes (*i.e.*, individuals without red cones). (c) Simulated view of a protanope for a contrast-enhanced version of the photograph recolored by Rasche et al.'s approach. (d) Simulated view of a protanope for the result produced by our technique. Note how our approach enhanced the overall image contrast by selectively changing only the colors for which there is a significant perceptual difference between thrichromats and dichromats. As a result, it preserved the naturalness of the colors (from the perspective of the dichromat) of the flowers' nuclei and of the background foliage (compare images (b) and (d)). For this 839×602-pixel image, our approach performs approximately 2,000× faster than Rasche et al.'s technique.

the red flowers from the green leaves and grass (Figure 4.6 b). Our approach clearly improved the perception of these flowers, while preserving the naturalness of the sky and the other elements of the scene, as perceived by the protanope (Figure 4.6 d). Compare our results with the ones produced by Rasche et al.'s approach (c).

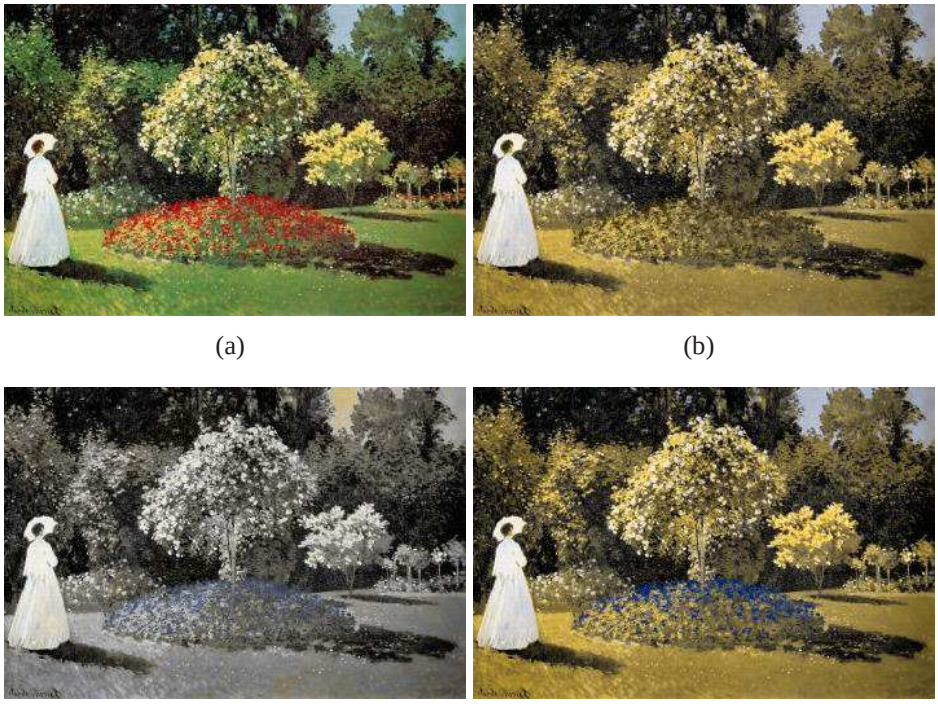
Figure 4.7 shows a Pablo Picasso's painting recolored for deuteranopes. Both Rasche et al.'s result (c) and ours (d) enhanced color contrast, but only ours preserved the naturalness of the yellow shades as seen by the dichromat (b).

Chinese Garden (Figure 4.8) provides an example of image recoloring for tritanopes. Note how our technique preserved the naturalness of the sky, while enhancing the contrast for the purple flowers. Rasche et al.'s approach, on the other hand, recolored the sky as pink and did not sufficiently enhanced the contrast of the purple flowers.

Figures 4.9 and 4.10 illustrate the use of our exaggerated color-contrast approach. Figure 4.9 shows the result of a simulated flame. Red and green colors in (a) mean high and low temperatures, respectively. Note how difficult it is for deuteranopes to distinguish regions of high from regions of low temperatures in (b). Figures 4.9 (c) and (d) present the results produced by our image-recoloring and exaggerated color-contrast approaches, respectively. Figure 4.10 (a) shows the visualization of carp dataset using a

(a) (b) (c) (d)

Figure 4.5: Color peppers: (a) Original image. (b) Simulation of a deuteranope's view for image (a). (c) Simulation of a deuteranope's view for the results produced by Rasche et al.'s technique. (d) Simulation of a deuteranope's view for the results produced by our approach, which preserved the color naturalness of the crates, the yellow peppers, and other vegetables in the background.



(a) (b)

(c) (d)

Figure 4.6: Signora in Giardino by Claude Monet, courtesy of Artcyclopedia.com: (a) Color image. Simulated views for a protanope for: (b) the original one, (c) result produced by Rasche et al.'s approach, and (d) result produced by the proposed technique.

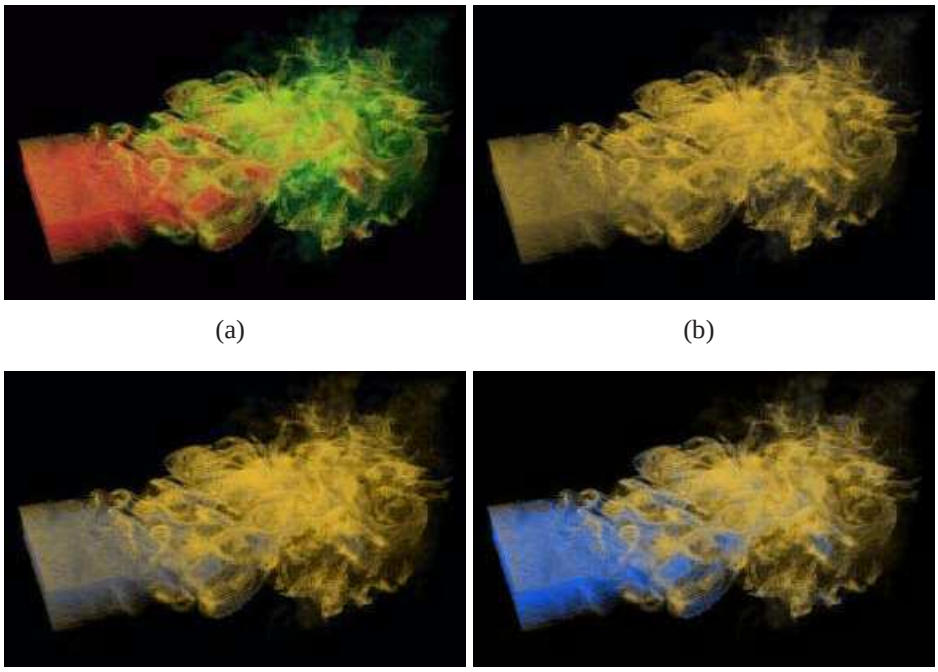
multi-dimensional transfer function and (b) presents this visualization as perceived by deuteranopes. Note how difficult it is for deuteranopes to distinguish the colors associated to the dataset. Figures 4.10 (c) and (d) show simulated views of a deuteranope for the results produced by our image-recoloring technique for dichromats and by our exaggerated color-contrast approach, respectively.

(a) (b) (c) (d)

Figure 4.7: Still Life by Pablo Picasso, courtesy of Artcyclopedia.com: (a) Color image. (b) Image in (a) as perceived by subjects lacking green-cones (deuteranopes). (c) and (d) are the results of Rasche et al.'s and our techniques, respectively, as seen by deuteranopes.

(a) (b) (c) (d)

Figure 4.8: Photograph of a chinese garden: (a) Color image. Simulated views of tritanopes for: (b) the original image, (c) the recolored image by Rasche et al.'s approach, and (d) the recolored image using our technique. Note the blue sky and the enhanced contrast for the purple flowers.



(a) (b)

(c) (d)

Figure 4.9: Simulation of a flame: (a) Color image. Simulated views of deuteranopes for: (b) original image, (c) result produced by our image-recoloring technique for dichromats, and (d) result produced by our exaggerated color-contrast approach using $x = 2$.

(a) (b)

(c) (d)

Figure 4.10: Visualization of a carp dataset using a multi-dimensional transfer function: (a) Color image. Simulated view of deuteranopes for: (b) original image, (c) result produced by our image-recoloring technique for dichromats, and (d) result produced by our exaggerated color-contrast approach using $x = 2$.

4.4 Summary

This chapter presented an efficient and automatic image-recoloring algorithm for dichromats that, unlike the current image-recoloring methods, allows these subjects to benefit from contrast enhancement without having to experience too unnatural colors. The proposed method uses a mass-spring system to obtain the set of optimal colors in the resulting image, and can be efficiently implemented both on CPU and on modern GPUs.

The chapter also introduced an extension to the proposed image-recoloring algorithm that exaggerates color contrast. This kind of feature might be useful for applications dealing with non-natural images, like scientific and information visualization.

5 THE COLOR-TO-GRAYSCALE ALGORITHM

This chapter describes our color-to-grayscale technique that uses both luminance and chrominance information. It also introduces a new error metric for evaluating the quality of color-to-grayscale transformations and discusses the results obtained with the proposed technique.

5.1 The Algorithm

Our color-to-grayscale algorithm is a specialization of the recoloring algorithm for dichromats presented in Chapter 4. In the recoloring algorithm, we searched for an optimal color contrast after projecting samples from a 3D color space into the 2D color gamut of a dichromat. For the color-to-grayscale problem, we search for the optimal contrast after projecting samples of a 3D color space now into a 1D color space. For this purpose, many of the same strategies used in the previous chapter can be reused here. Not surprisingly, both algorithms have many things in common, including the number and order of the steps, and the use of a mass-spring system as the optimization tool.

Thus, like the previous proposed algorithm, this one also has three main steps. The first step consists in obtaining a set Q of quantized colors from the set of all colors C in the input image I, and can be performed using any color quantization technique. The second step performs a constrained optimization on the values of the luminance channel of the quantized colors using a mass-spring system. At this stage, the chrominance information is taken into account in the form of constraints that specifies how much each particle can move (Section 5.1.1). The final gray values are reconstructed from the set of gray shades produced by the mass-spring optimization (Section 5.1.2). This final step guarantees local luminance consistency preservation.

5.1.1 Modeling and Optimizing the Mass-Spring System

Similar to the recoloring algorithm for dichromats, the color-to-grayscale mapping is modeled as a mass-spring system whose topology is a complete graph (*i.e.*, each particle P_i is connected to each other particle P_j by a spring S_{ij} with fixed stiffness $k_{ij} = 1$). Each particle P_i is associated to a quantized color $\vec{q_i} \in Q$ (represented in the almost perceptually uniform CIE L*a*b* color space) containing some mass m_i. Here, however, the particles are only allowed to move along the L*-axis of the color space and each particle P_i has its position p_i (in 1D) initialized with the value of the luminance coordinate of $\vec{q_i}$. Between each pair of particles (P_i, P_j), we create a spring with rest length given by

$$l_{ij} = \frac{G_{range}}{Q_{range}} \|\vec{q_i} - \vec{q_j}\| \tag{5.1}$$

where Q_{range} is the maximum difference between any pair of quantized colors in Q, G_{range} is the maximum possible difference between any pair of luminance values, and $\|\vec{q_i} - \vec{q_j}\|$ approximates the perceptual difference between colors $\vec{q_i}$ and $\vec{q_j}$. Note that since the luminance values are constrained to the L^*-axis, $G_{range} = 100$.

The instantaneous force applied to a particle P_i is obtained by summing the tensions of all springs connecting P_i to its neighbors P_j, according to Hooke's law (Equation 3.2). At each step of the optimization, we update l'_{ij} as $|L^*_i - L^*_j|$, and the new position p_i (actually the gray level L^*_i) according to Verlet's integration (Equation 3.3). The resulting system tends to reach an equilibrium when the perceptual differences between the optimized gray levels are proportional to the perceptual differences among the quantized colors in Q.

In order to enforce grayscale preservation, we set the mass m_i of particle P_i as the reciprocal of the magnitude of $\vec{q_i}$'s chrominance vector (Figure 5.1):

$$m_i = \frac{1}{\|(a^*_i, b^*_i)\|} \tag{5.2}$$

Note that $d = \|(a^*_i, b^*_i)\|$ is the distance from color $\vec{q_i}$ to the luminance axis L^*. Thus, less saturated colors present bigger masses and tend to move less. For achromatic colors, whose mass should be infinity, we avoid the division by zero simply by setting $F_i = 0$ (Equation 3.2). This keeps achromatic colors stationary.

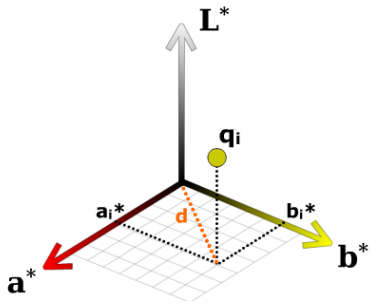


Figure 5.1: The mass of particle associated with a quantized color $\vec{q_i}$ is computed as the reciprocal of its distance d to the luminance axis L^*: $m_i = 1/(\|(a^*_i, b^*_i)\|)$. This enforces grayscale preservation, as achromatic colors will remain stationary.

5.1.2 Interpolating the Final Gray Image

The last step of the algorithm consists in obtaining the gray values for all pixels of the resulting image. For this task, we have adapted the two approaches described in Sections 4.1.3.1 and 5.1.2.2: *per-cluster interpolation* and *per-pixel interpolation*. The choice for one interpolation method depends on the application requirements.

5.1.2.1 Per-cluster interpolation

Consider the set $q_k \in Q$ of quantized colors and the respective associated set $g_k \in G$ of optimized gray levels. Let $C_k \subset C$ be a cluster composed of all colors in C that in the optimization are represented by the quantized color $\vec{q_k}$. The final gray level associated to the m-th color $\vec{c^k_m} \in C_k$ is then obtained as

$$gray(\vec{c^k_m}) = \begin{cases} g_k + r_k\|\vec{q_k} - \vec{c^k_m}\| & lum(\vec{c^k_m}) \geq lum(\vec{q_k}) \\ g_k - r_k\|\vec{q_k} - \vec{c^k_m}\| & otherwise \end{cases} \tag{5.3}$$

where lum is the function that returns the coordinate L* of a color in the L*a*b* color space, and r_k is Shepard's (SHEPARD, 1968) interpolation of ratios, in this case computed as

$$r_k = \frac{\sum_{i=1}^{\|Q\|} w_{ki} \frac{|g_k - g_i|}{\|\vec{q_k} - \vec{q_i}\|}}{\sum_{i=1}^{\|Q\|} w_{ki}}, for\ i \neq k \tag{5.4}$$

r_k indicates how close to the optimal solution is the gray value g_k, and $w_{ki} = 1/\|\vec{q_k} - \vec{q_i}\|^2$ is the distance-weighted term. For the quantized color $\vec{q_k}$ that represents the cluster, all gray values inside the k-th cluster are computed with respect to the optimized gray level g_k. Therefore, this transformation ensures local luminance consistency.

Again, given the set Q of quantized colors, the cost of computing all cluster ratios using Equation 5.4 on the CPU is $O(\|Q\|^2)$, while the cost of interpolating each pixel of an image with $N \times N$ pixels is $O(N^2)$.

5.1.2.2 Per-pixel interpolation

In this approach, each pixel's final shading is computed by optimizing it against the set $g_k \in G$ of previously optimized gray levels. This is achieved by using a mass-spring system, with springs connecting the current pixel (which is treated as a particle initialized with the pixel´s luminance value) and all optimized gray levels g_k. In this refined optimization stage, the particles associated to the optimized gray levels are kept stationary by setting the forces that act on them to zero (F_i in Equation 3.2). Equation 5.2 is then used to obtain the mass of the pixel being optimized. In this stage, all pixels with achromatic colors endup having infinite masses, remaining stationary. This ensures that all gray shades in the original color image will be preserved in the resulting grayscale image.

5.2 Error Metric for Color-to-Grayscale Mappings

We introduce an error metric to evaluate the quality of color-to-grayscale transformations. It consists of measuring whether the difference between any pairs of colors $(\vec{c_i}, \vec{c_j})$ in the original color image have been mapped to the corresponding target difference in the grayscale image. For this purpose, we defined an error function using root weighted mean square (RWMS):

$$rwms(i) = \sqrt{\frac{1}{\|K\|} \sum_{j \in K} \frac{1}{\delta_{ij}^2} \left(\delta_{ij} - |lum(\vec{c_i}) - lum(\vec{c_j})|\right)^2} \tag{5.5}$$

where, $rwms(i)$ is the error computed for the i^{th} pixel of the input color image I, K is the set of all pixels in I, $\|K\|$ is the number of pixels in I, $\delta_{ij} = (G_{range}/C_{range})\|\vec{c_i} - \vec{c_j}\|$ is the target difference in gray levels for a pair of colors $\vec{c_i}$ and $\vec{c_j}$, and lum is the function that return the component L* of a color. Since the differences are computed in the approximate perceptually uniform L*a*b* color space, $G_{range} = 100$ and C_{range} is the maximum distance between any two colors in the color image I. The weight $(1/\delta_{ij}^2)$ is used to suppress the bias toward large values of δ_{ij}.

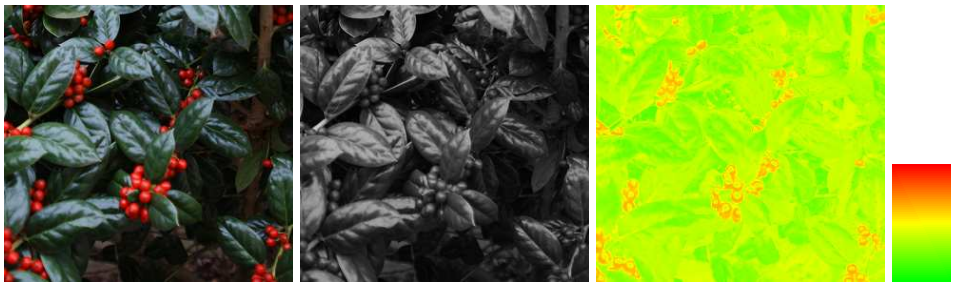
For an $N \times N$ image, evaluating Equation 5.5 for every pixel of I would take $O(N^4)$, which becomes impractical for large values of N. We can obtain a very good approximation to this error function by restricting the computation to the set $q_j \in Q$ of quantized

colors, given by:

$$rwms_q(i) = \sqrt{\frac{1}{\|K\|}\sum_{j\in Q}\frac{\|K_j\|}{{\delta_{ij}^q}^2}\left(\delta_{ij}^q - |lum(\vec{c_i}) - lum(\vec{q_j})|\right)^2} \tag{5.6}$$

In Equation 5.6, $K_j \subset K$ is the cluster of pixels represented by the quantized color $\vec{q_j}$, $\delta_{ij}^q = (G_{range}/Q_{range})\|\vec{c_i} - \vec{q_j}\|$, $\vec{c_i}$ is the color of the i-th pixel, and Q_{range} is the maximum distance between any two quantized colors in Q. We have compared the RWMS values produced by Equations 5.5 and 5.6 for a set of 50 images of natural scenes, paintings, charts, and maps. From this study, we found that the average relative difference between the two results was only 1.47%. Given these relatively small differences but its significantly smaller cost $O(\|Q\|N^2)$, all contrast errors shown in this work were computed using the metric represented by Equation 5.6. Also, in all contrast error images shown in this thesis, the green shade shown at the bottom of the error color ramp indicates $rwms = 0.0$, while the red shade at the top represents $rwms = 1.2$.

Figure 5.2 illustrates the use of our contrast error metric. Figure 5.2 (c) is the error image for the pair of color and grayscale images shown on Figures 5.2 (a) and (b), respectively, using a set of 64 quantized colors. The grayscale image was obtained as the luminance of image (a). As expected, the largest errors concentrate on the berries pixels, since these present the biggest contrast lost. Smaller errors are spread over the several green shades of the leaves.



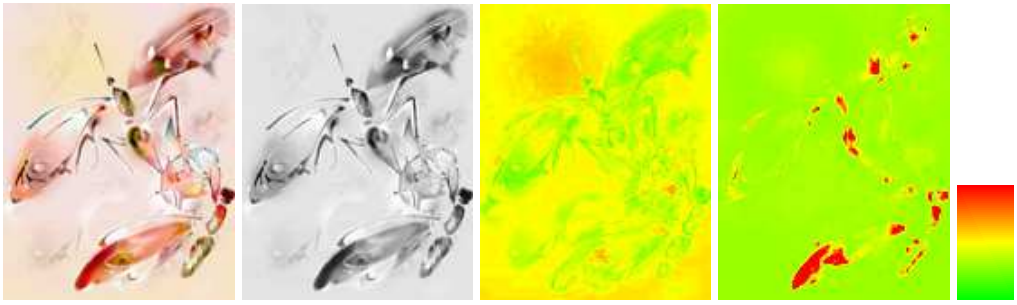
(a) (b) (c) (d)

Figure 5.2: Example of our contrast error metric. (a) Color image. (b) Luminance image of (a). (c) Error image computed using Equation 5.6. The largest contrast errors concentrate on the berry pixels.

Figure 5.3 illustrates the impact of the weighting term $1/{\delta_{ij}^q}^2$ on the metric. Figure 5.3 (b) shows a luminance image obtained from (a). The contrast error image is shown in Figure 5.3 (c) and was computed from a set of 128 quantized colors. The error image shown in (d) was computed using the RMS error (*i.e.*, removing the weighting term from Equation 5.6). Note that the region around the head of the butterfly has bigger contrast error, which has been captured by the error image in (c), while it was missed by the RMS error image shown in (d).

5.3 Results and Discussion

The described algorithms have been implemented in C++ and GLSL, and used them decolorize a very large number of images. The reported times were measured using a 2.2 GHz PC with 2 GB of memory and on a GeForce 8800 GTX with 768 MB of memory.



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

Figure 5.3: (a) Color image, (b) the luminance version of image (a), (c) RWMS error image. Note the perceptual error around the head of the butterfly, and (d) RMS error image.

Figure 5.4 compares the times for quantization and decolorizing images with various resolutions using different algorithms. Using the same notation introduced in the previous chapter, *MS-PC CPU* is our mass-spring algorithm using per-cluster interpolation in combination with k-means, and *MS-PP GPU* is our mass-spring algorithm using per-pixel interpolation with uniform quantization. In the case of k-means, we used a set of 128 colors and the code available at (ZHANG, 2006). In the case of uniform quantization, we discretized the RGB space using an uniform $10 \times 10 \times 10$ grid. Figure 5.4 shows that in all of its variations, our approach is a few orders of magnitude faster than both Gooch et al.'s and Rasche et al.'s approaches. All images and execution times shown in this work regarding the techniques of Gooch et al. (GOOCH et al., 2005a) and Rasche et al. (RASCHE; GEIST; WESTALL, 2005b) were obtained using software provided by these authors at (GOOCH et al., 2005b) and (RASCHE, 2005), respectively.

Figure 5.5 compares the results produced by various techniques with respect to grayscale preservation. One should note that only the luminance image (b) and the result produced by our method (f) are capable of preserving the original shades of gray. The luminance image, however, failed to distinguish the shades of the various isoluminant circles. Gooch et al.'s (Figure 5.5 c) and Rasche et al.'s (Figures 5.5 d and e) techniques changed the original gray shades in the resulting images.

Figures 5.6, 5.8, 5.9, and 5.10 compare the results, performance, and the overall contrast errors produced by the various algorithms. Table 5.1 summarizes these data. Following the authors comments on image quality, we did not use any quantization with Gooch et al.'s algorithm. For Rasche et al.'s and ours, the input images were quantized as shown on the second column of Table 5.1.

Table 5.1 also shows that our approach simultaneously presents the smallest RWMS error and is by far faster than Gooch et al.'s and Rasche et al.'s techniques. The luminance image, on the other hand, presents the biggest overall contrast errors, which is something that was already expected, since the color-to-luminance mapping completely ignores the chrominance information of the original image.

Figure 5.6 shows four grayscale renditions of Claude Monet's *Impressionist Sunrise* (Figure 5.7), with their respective contrast error images obtained using our metric. This example illustrates the robustness of our technique to handle large images. The *Sunrise* has (839×602) pixel and our GPU implementation performs the decolorization in 0.435 seconds. This is $751\times$ faster than Rasche et al.'s approach and $77,910\times$ faster than Gooch et al.'s. Our CPU implementation is still $247\times$ faster than Rasche et al.'s and $25,379\times$

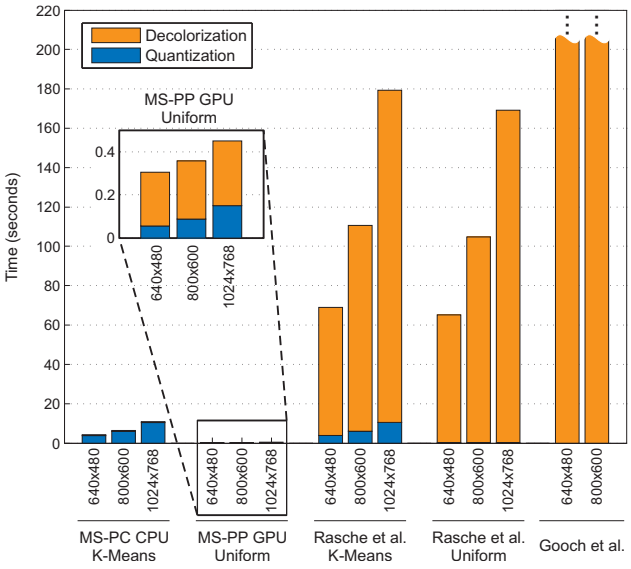
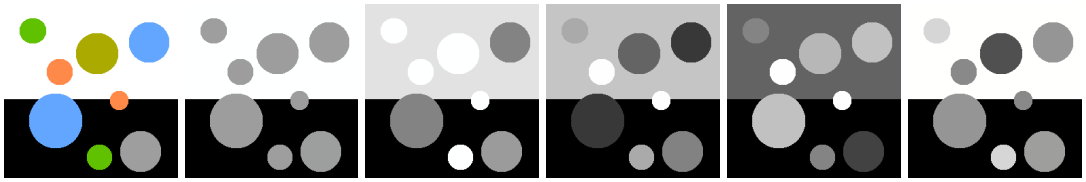


Figure 5.4: Performance comparison of various algorithms on a 2.2 GHz PC with 2 GB of memory and on a GeForce 8800 GTX GPU using images of different resolutions. Gooch et al. performed in 12,276 and 30,372 seconds for (640×480) and (800×600)-pixel images, respectively. Except for Gooch et al., all other techniques used a set of 128 quantized colors. Our mass-spring (MS) approaches optimized the set of quantized colors using 1,000 iterations. The *GPU* version obtained the final gray levels by optimizing each pixel with 100 iterations. Its results are detailed for better visualization. Note how the proposed approach scales well with the size of the input images.

faster than Gooch et al.'s.

Picasso Lovers (Figure 5.8) provides an example for which the result produced by Gooch et al.'s technique presents a large contrast error (Figure 5.8 h). For this same image, Rasche et al.'s approach produced a relatively small contrast error, but in the resulting grayscale image it is hard to distinguish between the lady's yellow skirt and the man's red clothes. For the photograph shown in Figure 5.9, the overall contrast error produced by Gooch et al.'s technique (Figure 5.9 h) is about the same as the one found in the luminance image (Figure 5.9 g).

Figure 5.10 illustrates the difficulty of Rasche et al.'s approach to capture some subtle shading variations among isoluminant colors. In this example, the smooth yellow hallo around the butterfly's head has been missed, while it was captured by Gooch et al.'s and our techniques.



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

Figure 5.5: Example of grayscale preservation. (a) Original color image with isoluminant circles; (b) Luminance image obtained from (a). Note it maps all isoluminant circle to the same shade of gray; (c) Result produced by Gooch et al.'s technique. Note that the two shades of green and the shade of orange turned into white in the resulting image; (d) and (e) are two results produced by Rasche et al.'s approach. (f) Grayscale image produced by our approach; Note that only the luminance image (b) and the result produced by our approach (f) preserved the original gray shades. The results shown in (d), (e) and (f) took a set of seven uniformly quantized colors as input.

Label	**Image (size)**	**Quant. (#colors)**
#1	Sunrise (839×602)	uniform (264)
#2	Lovers (301×407)	k-means (255)
#3	Boats (193×282)	uniform (141)
#4	Butterfly (128×164)	k-means (120)

Image	**Lum.**	**Gooch et al.**		**Rasche et al.**		**MS-PC CPU**		**MS-PP GPU**	
Label	**RWMS**	**Time**	**RWMS**	**Time**	**RWMS**	**Time**	**RWMS**	**Time**	**RWMS**
#1	0.707	33,501.4	0.557	326.78	0.564	1.32	0.429	0.43	0.425
#2	0.690	1,882.5	0.699	87.36	0.498	0.96	0.486	0.36	0.477
#3	0.634	328.3	0.624	20.10	0.513	0.35	0.432	0.17	0.428
#4	0.582	57.3	0.535	5.54	0.443	0.21	0.365	0.15	0.362

Table 5.1: Summary of the performance and overall contrast error produced by the various techniques when applied to the test images. Time measured in seconds. Our approach presents the smallest RWMS error for all examples and is significantly faster than the other techniques. The speedups increase with the image sizes. For the *Sunrise* image, with (839×602) pixel, our GPU implementation is $751\times$ faster than Rasche et al.'s (CPU) approach and $77,910\times$ faster than Gooch et al.'s (CPU).

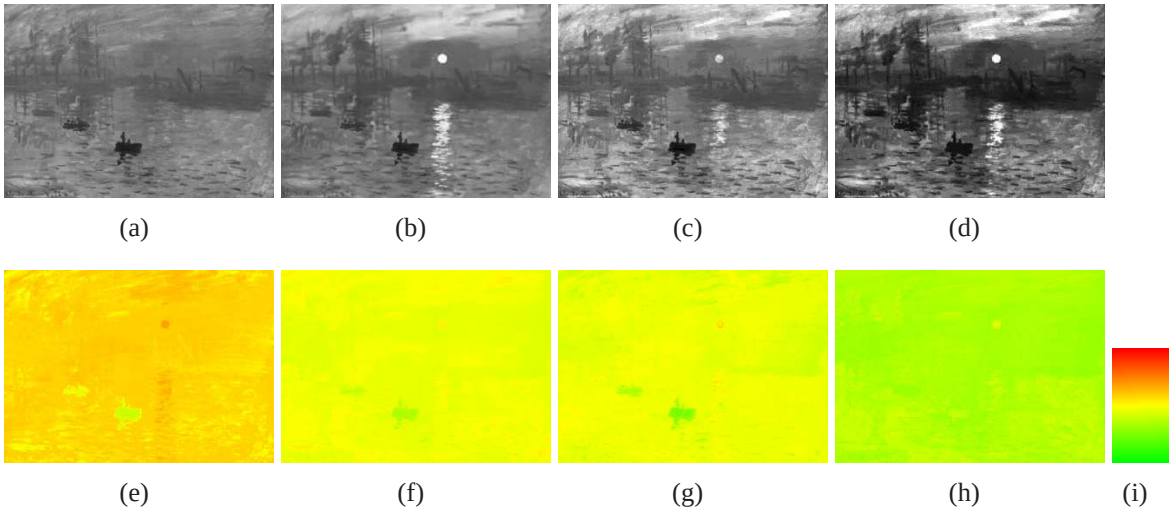


Figure 5.6: Four grayscale renditions of Claude Monet's Impressionist Sunrise (Figure 5.7), with their respective contrast error images obtained using our metric. (a) to (d) are grayscale images with their corresponding per-pixel contrast error images (e) to (h), respectively. (a) Luminance image. (b) Grayscale image produced by Gooch et al.'s method using its default parameters. (c) A grayscale image produced by Rasche et al.'s approach. (d) Grayscale image produced by our approach. RWMS error images: (e) $rwms = 0.582$, (f) $rwms = 0.535$, (g) $rwms = 0.443$, (h) $rwms = 0.365$. (i) Error scale: red means bigger error.

Figure 5.7: Color image (Impressionist Sunrise by Claude Monet, courtesy of Artcyclopedia.com).

Figure 5.8: Pablo Picasso's Lovers: (a) Color image (courtesy of Artcyclopedia.com). (b) to (e) are grayscale images with their per-pixel contrast error images (g) to (j), respectively. (b) Luminance image. (c) Grayscale image produced by Gooch et al.'s method using its default parameters. (d) A grayscale image produced Rasche et al.'s approach. Note that it is hard to distinguish between the lady's yellow skirt and the man's red clothes. (e) Grayscale image produced by our approach. (f) Error scale: red means bigger error. RWMS error images: (g) to (j).

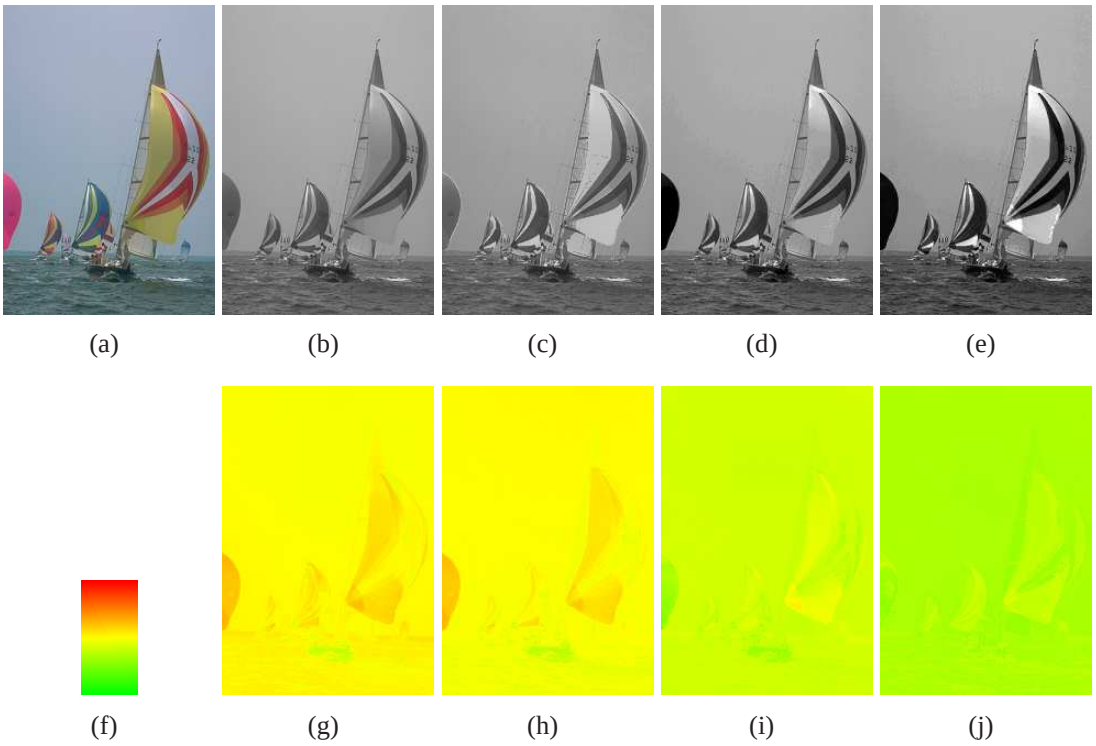


Figure 5.9: Photograph of a natural scene: (a) Color image. (b) to (e) are grayscale images with their per-pixel contrast error images (g) to (j), respectively. (b) Luminance image. (c) Grayscale image produced by Gooch et al.'s method using its default parameters. (d) A grayscale image produced Rasche et al.'s approach. (e) Grayscale image produced by our approach. (f) Error scale: red means bigger error.(g) to (j).

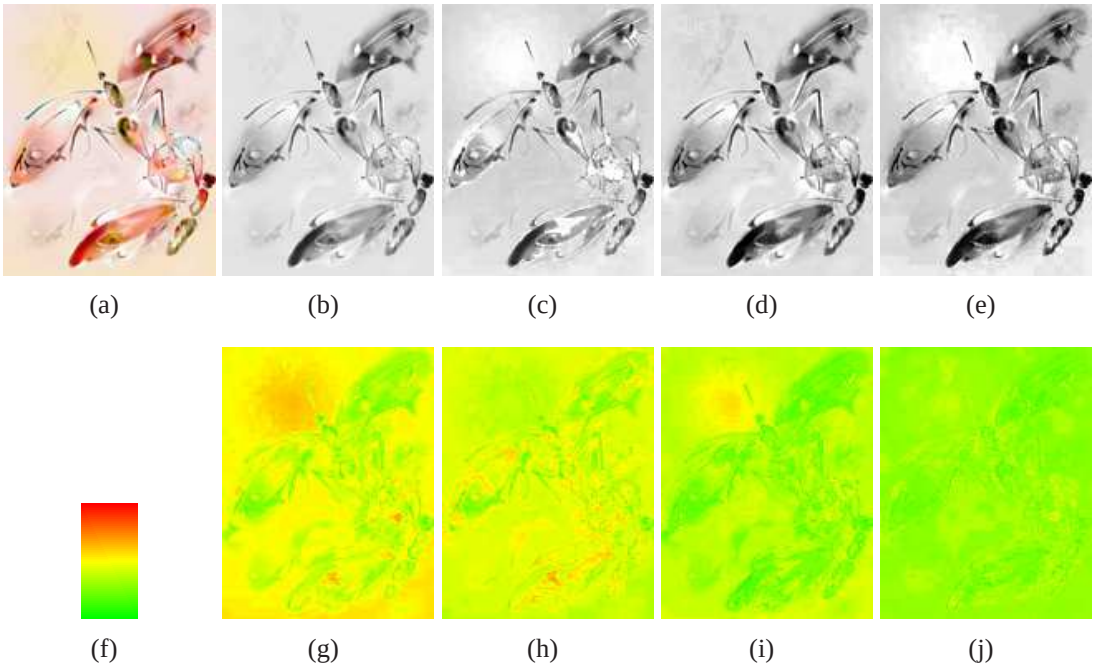


Figure 5.10: Butterfly: (a) Color image. (b) to (e) are grayscale images with their per-pixel contrast error images (g) to (j), respectively. (b) Luminance image. (c) Grayscale image produced by Gooch et al.'s method using its default parameters. (d) A grayscale image produced Rasche et al.'s approach. (e) Grayscale image produced by our approach. (f) Error scale: red means bigger error. RWMS error images: (g) to (j).

5.4 Summary

This chapter described a new color-to-grayscale technique that uses both luminance and chrominance information to preserve the contrast, found in the input color image, in the resulting grayscale one. The proposed method is based on a mass-spring optimization and is almost three orders of magnitude faster than current optimization-based techniques.

This chapter also presented an error metric for evaluating the quality of color-to-grayscale transformations. It measures the error between the differences of any pairs of colors in the original image and the corresponding differences in the grayscale image. The proposed metric is based on a RWMS error.

6 CONCLUSIONS

This thesis presented an efficient naturalness-preserving image-recoloring algorithm for dichromats based on mass-spring optimization. Contrary to previous automatic techniques, the proposed method allows dichromats to benefit from contrast enhancement without having to experience unnatural colors. Besides being deterministic, our technique has many attractive properties: (i) it satisfies global color consistency; (ii) ensures luminance constancy; (iii) maintains local chrominance consistency; and (iv) can be efficiently implemented on modern GPUs. Both CPU and GPU versions of the proposed algorithm are significantly faster than previous approaches (ICHIKAWA et al., 2004; WAKITA; SHIMAMURA, 2005; RASCHE; GEIST; WESTALL, 2005b; JEFFERSON; HARVEY, 2006). It has also presented an extension to our image-recoloring method that exaggerates the color contrast for dichromats in the result image. Such a feature is useful for applications involving non-natural images (*e.g.*, scientific and information visualization).

A second contribution of this thesis is an efficient mass-spring-based approach for contrast enhancement during color-to-grayscale image conversion. The proposed method is more than three orders of magnitude faster than previous optimization-based techniques (GOOCH et al., 2005a; RASCHE; GEIST; WESTALL, 2005b), while producing superior results both in terms of contrast preservation and image guarantees. Our algorithm satisfies a global consistency property, preserves grayscale values present in the color image, maintains local luminance consistency, is completely automatic, and can be efficiently implemented on modern GPUs.

Another contribution of this thesis is an error metric for evaluating the quality of color-to-grayscale transformations. The proposed metric is based on a RWMS error that measures whether the difference between any pairs of colors in the original image have been mapped to the corresponding target difference in the grayscale image.

The quality of the results produced by our image-recoloring and color-to-grayscale techniques depend on the quality of quantization performed in their first stage. For low-quality images, the quantization algorithm may fail to generate a good set of quantized colors, causing the result images to exhibit artifacts. Although we ensure a continuous mapping among the colors/gray-shades in any given cluster, currently the algorithms do not ensure a continuous mapping among different clusters. In practice, however, after extensive tests on a large number of images, we have not noticed any objectionable artifacts due to these limitations.

This work opens up several avenues for future exploration. As the proposed approaches were designed to deal with static images, we plan to explore ways to extend our techniques to perform video sequences recolorization/decolorization. Preliminary results show that we can enforce temporal coherence by initializing the mass-spring optimization with particles computed for previous frames, and by keeping those parti-

cles stationary. Temporal coherence is not preserved by related techniques (GOOCH et al., 2005a; GRUNDLAND; DODGSON, 2007; JEFFERSON; HARVEY, 2006; NEUMANN; CADIK; NEMCSICS, 2007; RASCHE; GEIST; WESTALL, 2005b; WAKITA; SHIMAMURA, 2005).

We believe that our image-recoloring technique can have a positive impact on the way dichromats interact with digital media, as it finally provides a practical way of disambiguating colors without contradicting, as much as possible, their memories about how the world looks like. We hope these results will inspire the design of new applications and interfaces for dichromats.

With regarding to our color-to-grayscale technique, the unique combination of high-fidelity capture of color differences, grayscale preservation, global consistency, local luminance consistency, and speed makes our technique a good candidate for replacing standard luminance-based color-to-grayscale algorithms in printing and pattern recognition applications.

REFERENCES

ABI Research. **Game Industry Revenue Expected To Double By 2011**. 2006.

BRETTEL, H.; VIéNOT, F.; MOLLON, J. D. Computerized simulation of color appearance for dichromats. **J. Opt. Soc. Am.**, [S.l.], v.14, n.10, p.2647–2655, 1997.

BROWN, R. **Photoshop**: converting color to black-and-white. 2006.

ENGEL, W. (Ed.). **Storing and Accessing Topology on the GPU**: a case study on mass-spring systems. [S.l.: s.n.], 2006. p.565–578.

DOUGHERTY, R.; WADE, A. **Daltonize**. Accessed on Oct/06, (http://www.vischeck.com/daltonize).

DUNTEMAN, G. **Principal Components Analysis**. [S.l.]: Sage Publications, Inc., 1989.

FIDANER, O.; LIN, P.; OZGUVEN, N. **Analysis of Color Blindness - Project Writeup**. 2005.

GEORGII, J.; WESTERMANN, R. Mass-Spring Systems on the GPU. **Simul. Modelling Practice and Theory**, [S.l.], v.13, p.693–702, 2005.

GONZALEZ, R. C.; WOODS, R. E. **Digital Image Processing**. 3ed. ed. [S.l.]: Prentice Hall, 2007.

GOOCH, A. A.; OLSEN, S. C.; TUMBLIN, J.; GOOCH, B. Color2Gray: salience-preserving color removal. **ACM Trans. Graph.**, New York, NY, USA, v.24, n.3, p.634–639, 2005.

GOOCH, A. A.; OLSEN, S. C.; TUMBLIN, J.; GOOCH, B. **Color2Gray**: salience-preserving color removal. 2005.

GRUNDLAND, M.; DODGSON, N. A. Decolorize: fast, contrast enhancing, color to grayscale conversion. **Pattern Recogn.**, New York, NY, USA, v.40, n.11, p.2891–2896, 2007.

HEATH, G. The Handicap of Color Blindness. **Journal of the American Optometric Association**, [S.l.], v.45, n.1, p.62–69, 1974.

IACCARINO, G.; MALANDRINO, D.; PERCIO, M. D.; SCARANO, V. Efficient edge-services for colorblind users. In: WWW '06, 2006. **Proceedings...** [S.l.: s.n.], 2006. p.919–920.

ICHIKAWA, M.; TANAKA, K.; KONDO, S.; HIROSHIMA, K.; ICHIKAWA, K.; TANABE, S.; FUKAMI, K. Web-Page Color Modification for Barrier-Free Color Vision with Genetic Algorithm. **Lecture Notes in Computer Science**, [S.l.], v.2724, p.2134–2146, 2003.

ICHIKAWA, M.; TANAKA, K.; KONDO, S.; HIROSHIMA, K.; ICHIKAWA, K.; TANABE, S.; FUKAMI, K. Preliminary study on color modification for still images to realize barrier-free color vision. In: IEEE SMC '06, 2004. **Anais...** [S.l.: s.n.], 2004. v.1, p.36–41.

Id Software, Inc. **(http**://www.idsoftware.com/). Accessed on May/06.

JEFFERSON, L.; HARVEY, R. Accommodating color blind computer users. In: ASSETS '06, 2006. **Proceedings...** [S.l.: s.n.], 2006. p.40–47.

JESCHKE, E. R. **GIMP**: converting color images to b&w. Accessed in Aug/07, (http://www.gimp.org/tutorials/Color2BW/).

JUDD, D. B. Color perceptions of deuteranopic and protanopic observers. **J. Res. Natl. Bur. Std.**, [S.l.], v.41, p.247–271, 1948.

MEYER, G. W.; GREENBERG, D. P. Color-defective vision and computer graphics displays. **IEEE Comput. Graph. Appl.**, [S.l.], v.8, n.5, p.28–40, 1988.

NEMCSICS, A. Coloroid Color System. **Color Research and Application**, [S.l.], v.5, p.113–120, 1980.

NEUMANN, L.; CADIK, M.; NEMCSICS, A. An Efficient Perception-Based Adaptive Color to Gray Transformation. In: COMPUT. AESTHETICS, 2007. **Proceedings...** [S.l.: s.n.], 2007. p.73– 80.

PFISTER, H.; LORENSEN, B.; BAJAJ, C.; KINDLMANN, G.; SCHROEDER, W.; AVILA, L. S.; MARTIN, K.; MACHIRAJU, R.; LEE, J. The Transfer Function Bake-Off. **IEEE Comput. Graph. Appl.**, Los Alamitos, CA, USA, v.21, n.3, p.16–22, 2001.

RASCHE, K. **Detail Preserving Color Transformation**. 2005.

RASCHE, K.; GEIST, R.; WESTALL, J. Detail Preserving Reproduction of Color Images for Monochromats and Dichromats. **IEEE Comput. Graph. Appl.**, Los Alamitos, CA, USA, v.25, n.3, p.22–30, 2005.

RASCHE, K.; GEIST, R.; WESTALL, J. Re-coloring Images for Gamuts of Lower Dimension. **Comput. Graph. Forum**, [S.l.], v.24, n.3, p.423–432, 2005.

RIGDEN, C. The Eye of the Beholder - Designing for Colour-Blind Users. **British Telecommunications Engineering**, [S.l.], v.17, 1999.

GEGENFURTNER, K. R.; SHARPE, L. T. (Ed.). **Color Vision**: from genes to perception. [S.l.]: Cambridge University Press, 1999. p.3–51.

SHEPARD, D. A two-dimensional interpolation function for irregularly-spaced data. In: ACM NATIONAL CONFERENCE, 1968., 1968. **Proceedings...** [S.l.: s.n.], 1968. p.517–524.

SLOAN, L. L.; WOLLACH, L. Case of unilateral deuteranopia. **J. Opt. Soc. Am.**, [S.l.], v.38, n.6, p.502, 1948.

TEJADA, E.; ERTL, T. Large Steps in GPU-based Deformable Bodies Simulation. **Simulation Practice and Theory**, [S.l.], v.Special Issue on Programmable Graphics Hardware, p.703–715, 2005.

VERLET, L. Computer "Experiments" on Classical Fluids. I. Thermodynamical Properties of Lennard-Jones Molecules. **Phys. Rev.**, [S.l.], v.159, n.1, p.98, Jul 1967.

VERTH, J. M. V.; BISHOP, L. M. **Essential Mathematics for Games and Interactive Applications**. [S.l.]: Morgan Kaufmann, 2004.

VIéNOT, F.; BRETTEL, H. Color display for dichromats. In: SPIE, 2000. **Proceedings...** [S.l.: s.n.], 2000. v.4300, p.199–207.

VIéNOT, F.; BRETTEL, H.; MOLLON, J. D. Digital Video Colourmaps for Checking the Legibility of Displays by Dichromats. **Color Research and Application**, [S.l.], v.24, p.243–252, 1999.

WAKITA, K.; SHIMAMURA, K. SmartColor: disambiguation framework for the colorblind. In: ASSETS '05, 2005. **Proceedings...** [S.l.: s.n.], 2005. p.158–165.

WALRAVEN, J.; ALFERDINCK, J. W. Color displays for the color blind. In: IS&T AND SID 5TH COLOR IMAGING CONFERENCE, 1997. **Anais...** [S.l.: s.n.], 1997. p.17–22.

WANDELL, B. A. **Foundations of Vision**. [S.l.]: Sinauer Association, 1995.

ZHANG, R. **K-means clustering**. 2006.