INTERACTIVE COLOR TRANSFER BETWEEN IMAGES

SOMCHAI PHATTHANACHUANCHOM

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR
THE DEGREE OF DOCTOR OF PHILOSOPHY
(COMPUTER SCIENCE)
FACULTY OF GRADUATE STUDIES
MAHIDOL UNIVERSITY
2010

Thesis
entitled

INTERACTIVE COLOR TRANSFER BETWEEN IMAGES

.....
Mr. Somchai Phatthanachuanchom
Candidate

.....
Lect. Rawesak Tanawongsuwan,
Ph.D.
Major advisor

.....
Asst. Prof.Sukanya Phongsuphap,
Ph.D.
Co-advisor

.....
Asst. Prof.Chomtip Pornpanomchai,
D.Tech.Sc.
Co-advisor

.....
Prof. Banchong Mahaisavariya,
M.D., Dip Thai Board of Orthopedics
Dean
Faculty of Graduate Studies
Mahidol University

.....
Assoc. Prof. Supachai Tangwongsan,
Ph.D.
Program Director
Doctor of Philosophy Program in
Computer Science
Faculty of Information and
Communication Technology
Mahidol University

Thesis
entitled

INTERACTIVE COLOR TRANSFER BETWEEN IMAGES

was submitted to the Faculty of Graduate Studies, Mahidol University
for the degree of Doctor of Philosophy (Computer Science)
on
May 14, 2010

.....
Mr. Somchai Phatthanachuanchom
Candidate

.....
Asst. Prof. Vajirasak Vanijja,
Ph.D.
Chair

.....
Lect. Rawesak Tanawongsuwan,
Ph.D.
Member

.....
Asst. Prof. Sukanya Phongsuphap,
Ph.D.
Member

.....
Assoc. Prof. Damras Wongsawang,
Ph.D.
Member

.....
Prof. Banchong Mahaisavariya,
M.D., Dip Thai Board of Orthopedics
Dean
Faculty of Graduate Studies
Mahidol University

.....
Assoc. Prof. Jarernsri L. Mitrpanont,
Ph.D.
Acting Dean
Faculty of Information and
Communication Technology
Mahidol University

ACKNOWLEDGEMENTS

I would like to acknowledge several people who constantly gave me support and encouragement, and without whom I would never have come this far in this test of endurance.

I am extremely grateful to my advisor Dr. Rawesak Tanawongsuwan for his stimulating advice, inspiring feedback and insightful supervision throughout the research work.

I would also like to thank my co-advisors Asst. Prof. Sukanya Phongsuphap and Asst. Prof. Chomtip Pornpanomchai for their valuable comments and support in this study. I am indebted to the committees Assoc. Prof. Damras Wongsawang and Asst. Prof. Vajirasak Vanijja for their time.

I would also like to mention how grateful I am to all the instructors in this PhD course, including those who are currently not at Mahidol University. Much gratitude also goes to the staff of Mahidol Computer Center and the ICT faculty, including the library staff. Many thanks also to my colleagues and friends at Mahidol and elsewhere for incessant motivation and consolation that kept me going in the face of obstacles in the past years.

I acknowledge the scholarship from Mahidol University and I would like to thank for generous financial support throughout my study.

Finally, I would like to express my sincere gratitude to my parents for giving me life, understanding, love, for educating me, and giving me endless opportunity with unconditional support.

Somchai Phatthanachuanchom

INTERACTIVE COLOR TRANSFER BETWEEN IMAGES

SOMCHAI PHATTHANACHUANCHOM 4637825 ITCS/D

Ph.D. (COMPUTER SCIENCE)

THESIS ADVISORY COMMITTEE: RAWESAK TANAWONGSUWAN, Ph.D.,
SUKANYA PHONGSUPHAP, Ph.D., CHOMTIP PORNPANOMCHAI, D.Tech.Sc.

ABSTRACT

Color content in an image is important to human interpretation and can be manipulated by color transfer techniques that change the color tone of one image using color information from another image. With complex color distributions, as found in natural images, color artifacts are often found when color transfer is applied globally to an image. Moreover, there are demands for control over the color transfer processes. Two color transfer methods are being proposed to handle complex color distributions and give users more control over the processes. The first is global color transfer that automatically captures nonlinear color structures of images in high dimensional spaces, with color matching that involves searching and matching the closest pairs of source and target colors. With this method, inaccuracies due to color artifacts can be avoided. The second method is interactive color transfer by region exploration. This process starts with region definition that automatically defines and models local regions of images, and then is followed by region exploration, which is an interactive and iterative process that lets users explore results from color transfer between regions. This method generates a repertoire of possible results from region mapped pairs that users can then explore, evaluate, and select desirable results from. The findings of this research create two alternative approaches, automatic or interactive, for the manipulation of color information of images that can handle even the color complexity of natural images. The interactive color transfer method and framework proposed here allow for the transfer process to be responsive to user's decisions, artistic creativity, and unforeseeable demands.

KEY WORDS : COMPUTER GRAPHICS / IMAGE PROCESSING
COLOR TRANSFER / INTERACTIVE TOOLS

107 pages

การถ่ายโอนสีแบบโต้ตอบระหว่างรูปภาพ

INTERACTIVE COLOR TRANSFER BETWEEN IMAGES

สมชาย พัฒนาชวนชม 4637825 ITCS/D

ปร.ด. (วิทยาการคอมพิวเตอร์)

คณะกรรมการที่ปรึกษาวิทยานิพนธ์: รวีศักดิ์ ธนวงศ์สุวรรณ, Ph.D., สุกัญญา พงษ์สุภาพ, Ph.D, ชมทิพ พรพนมชัย, D.Tech.Sc.

บทคัดย่อ

สีของรูปภาพมีความสำคัญต่อการมองเห็นและการตีความ การจัดการสีด้วยเทคนิคการถ่ายโอนสีเป็นการเปลี่ยนโทนสีของภาพหนึ่งโดยอาศัยข้อมูลสีจากภาพอีกภาพหนึ่ง เนื่องจากการถ่ายโอนสีแบบทั้งภาพระหว่างภาพธรรมชาติที่มีการกระจายของสีที่ซับซ้อนมักพบสีแปลกปลอม อีกทั้งผู้ใช้ยังมีความต้องการการควบคุมในกระบวนการถ่ายโอนสี งานวิจัยนี้นำเสนอกระบวนการถ่ายโอนสี 2 วิธี เพื่อจัดการปัญหาการกระจายของสีที่ซับซ้อนและให้ผู้ใช้งานสามารถควบคุมกระบวนการถ่ายโอนสีได้มากขึ้น วิธีแรกคือการถ่ายโอนสีทั้งภาพซึ่งจับโครงสร้างของสีที่ไม่เป็นแบบเชิงเส้นของรูปภาพในปริภูมิมิติสูงได้โดยอัตโนมัติ ด้วยการค้นหาและจับคู่สีที่ใกล้เคียงกันที่สุดของพิกเซลต้นฉบับกับพิกเซลเป้าหมาย ทำให้สามารถหลีกเลี่ยงการเกิดสีแปลกปลอมในรูปภาพผลลัพธ์ได้ วิธีที่สองคือการถ่ายโอนสีแบบโต้ตอบด้วยการสำรวจส่วนของภาพ โดยกระบวนการเริ่มจากการกำหนดส่วนภาพ ซึ่งเป็นการระบุขอบเขตและสร้างโมเดลส่วนของภาพโดยอัตโนมัติ ตามด้วยการสำรวจส่วนภาพซึ่งเป็นกระบวนการทำซ้ำแบบโต้ตอบ โดยให้ผู้ใช้งานเป็นผู้สำรวจผลลัพธ์จากการถ่ายโอนสีระหว่างส่วนของภาพ วิธีนี้ได้สร้างรูปภาพผลลัพธ์ที่เป็นไปได้จำนวนมากจากการจับคู่ระหว่างส่วนต่างๆของภาพต้นฉบับกับภาพเป้าหมาย เปิดโอกาสให้ผู้ใช้สำรวจ ประเมิน และเลือกผลลัพธ์ที่พึงพอใจได้ตามต้องการ การพัฒนาเทคนิคถ่ายโอนสี 2 เทคนิคที่เป็นข้อค้นพบในงานวิจัยนี้ ได้สร้างทางเลือกที่มีประสิทธิภาพในการจัดการข้อมูลสีของภาพธรรมชาติที่มีความซับซ้อน ทั้งในแบบอัตโนมัติและแบบโต้ตอบ วิธีการและกรอบการทำงานในการถ่ายโอนสีแบบโต้ตอบที่ได้เสนอในที่นี้ยังเปิดโอกาสให้กระบวนการถ่ายโอนสีได้ตอบสนองต่อการตัดสินใจของผู้ใช้ รวมถึงความคิดสร้างสรรทางศิลปะและความต้องการอื่นๆที่ไม่สามารถคาดการณ์ได้

107 หน้า

CONTENTS

CONTENTS (cont.)

CONTENTS (cont.)

LIST OF FIGURES

LIST OF FIGURES (cont.)

LIST OF FIGURES (cont.)

CHAPTER I
INTRODUCTION

People usually edit an image for a specific purpose. For example, they edit an image to improve its qualities by increasing contrast to reveal more details, deblurring or removing noise, artifact, or distortion in order to alleviate recognizability of the image. People also modify image characteristics or image features such as color, brightness, style, or texture to adjust the appearance of an image, which sometimes referred to as "the look and feel" of the image.

In general, an image appearance is edited by adjusting some features in either specific locations or the whole image. The features are modeled by parameters which can be tuned. For instance, to adjust the color appearance of an image, the color feature is modeled by parameters such as color means and color deviations of major color channels of red, green, and blue. The difficulties of image editing by parameter tuning depend on how well the features are modeled and how easily the parameters can be understood by a user.

There is a number of software tools available for image editing, such as tools for adjusting image contrast or tools for changing image color tone. Some tools require parameter insertion or interaction from a user. Therefore, the quality of the resulting image depends largely on the experience of a user whose skill in adjusting parameters by hand is the key to success.

Figure 1.1 shows an example tool for image editing by a parameter tuning method, illustrating the user interface of the software tool that makes a grayscale image from a color image via a channel mixer. A user needs to understand the color channels in order to adjust an appropriate combination of each channel and make a final grayscale image.

Editing an image so that the resulting appearances are closely similar to that of another image, a reference image, is a complicated process, especially when

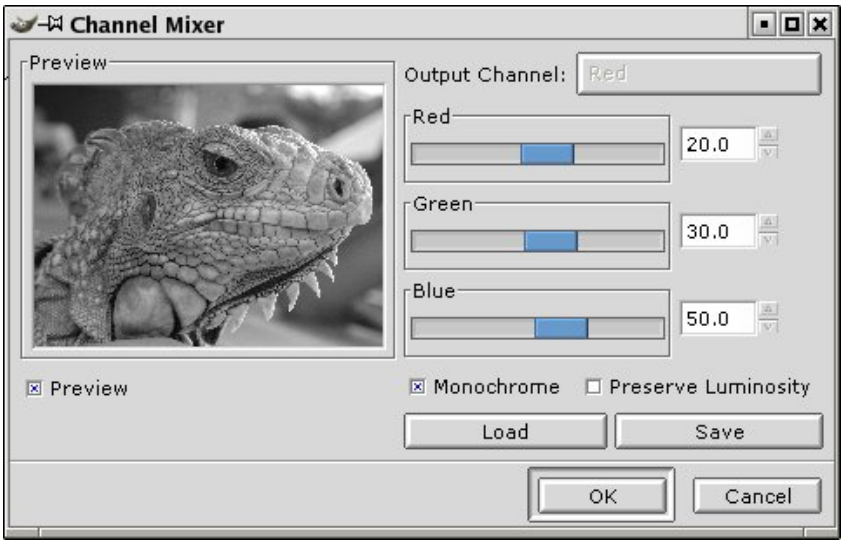


Figure 1.1: A user interface of image editing by parameter tuning using GNU image manipulation program (GIMP). This example comes from the GIMP tutorial [1].

attempted by hand, even for an experienced user. It is usually difficult to find the right model parameters of the image characteristics of the reference image. The trial and error strategies can be used but require a user experience and a lot of time for guessing.

Instead of using parameter tuning method, another technique is image characteristic transfer. The transfer technique requires less inputs from a user but needs a reference image. The transfer technique is also known as example-based edition. Example-based edition extracts image characteristics from the reference image and subsequently transfers them to another image in order to imitate the image characteristics of the reference image. The quality of the result by example-based edition does not depend on the editing skills of a user, but the quality of the reference image and the effectiveness of the transfer technique itself.

The algorithm of the transfer technique influences or affects the time used. Comparing with the parameter tuning technique, the transfer technique takes less time for overall processes with the current computer specification. Especially, when dealing with a complex image, editing by hand takes very long time and hardly returns a convincing result.

Example-based edition also allows us to take advantages of reference images because some visual appearances are hard to describe or explain, and are difficult to imitate by parameter tuning. Nowadays, some web sites such as www.flickr.com collect

photographs from users around the world, particularly from professional photographers, and also make some free photo collections available. This raises opportunities to use these available photographs in the image characteristic transfer.

In the past decades, researchers are interested in and proposed a number of image characteristic transfer methods. For example, Fig.1.2 shows an example of color transfer by Pitié et al. [2]. There are other color transfer techniques such as Reinhard et al. [3] and Neumann et al. [4]. The technique aims to transfer the color appearance of a target image (b) to a source image (a). The result of color transfer (c) shows the source image with a new color appearance similar to that of the target image.

(a) Source image (b) Target image (c) Result image

Figure 1.2: The result image together with its source and target images isi derived by color transfer [2].

Adding color to a grayscale image is called a colorization technique. Figure 1.3 shows an example-based colorization technique by Welsh et al. [5] that colorizes a grayscale image (a) using colors from the reference image (b). The result of the colorization technique is shown in (c). Another colorization technique is the interactive colorization [6] shown in Fig. 1.4. In this figure, a grayscale image with stroke colors specified by a user is shown in (a). The result of interactive colorization (b) is compared with the original image (c) which is the colored version of the grayscale image.

Texture transfer can be achieved by synthesizing texture patches from a sample texture image on object structures that appear in a source image. There are many texture transfer techniques such as Efros et al. [7], [8], Wei et al. [9], Hertzmann et al. [10], and Ashikhmin et al. [11], [12]. Figure 1.5 shows an example of texture transfer that generates texture patterns from a sample texture image (b) and patches into a source image (a). The result (c) can show the man face with the new texture pattern

(a) Source image (b) Target image (c) Result image

Figure 1.3: The result image together with its grayscale and color images is derived by example-based colorization [5].

(a) Grayscale image (b) Result image (c) Original color image

Figure 1.4: The result image together with its grayscale and original colored version images is derived by interactive colorization [6].

synthesized from the sample texture image.

There are many types of style transfer techniques that usually aim to imitate styles from paintings or drawings as in Hertzmann et al. [10], Neumann et al. [4], Chen et al. [13], and Xie et al. [14]. Figure 1.6 shows an example of a style transfer technique that transfers the pencil style from a drawing (b) to a source image (a). The result (c) shows the pencil-styled drawing of the source image.

As aforementioned, there are many characteristic transfer techniques in this research area. Our focus is color transfer which is an example-based method of image characteristic transfer. As illustrated in Fig. 1.2, color transfer is a technique that changes the color appearance of one image (a source image) to appear like that of a reference image (a target image) by using the color transfer function. A user may need to get involved in the color transfer process. Interactive color transfer between images is introduced as one type of color transfer techniques that lets a user manipulate local regions and control the processes of color transfer via interactive tools.

(a) Source image (b) Sample texture (c) Result image

Figure 1.5: The result image together with its source and sample texture images is derived by texture transfer [8].

(a) Source image (b) Drawing (c) Result image

Figure 1.6: The result image together with its source image and a drawing is derived by pencil-style transfer [14].

1.1 Research Motivation

The global color transfer proposed by Reinhard et al. [3] is effective when applies to simple color distributions that can be modeled by Gaussian models. These model parameters are suitable for the linear color transfer function. However, some images such as natural images have complex color distributions which cannot be modeled by Gaussian models and usually cause color artifacts in the result image.

Figures 1.7 illustrates the variety of color distributions in photos of natural sky which ranges from simple to complex color distributions. In general, a simple color distribution represents one color tone such as an empty blue sky or yellow sky (one color tone) while more complex color distributions represent more than one color tone such as a cloudy sky (two color tones) or a colorful sky (3-4 color tones).

A simple solution is using swatches in order to limit the regions of inter-

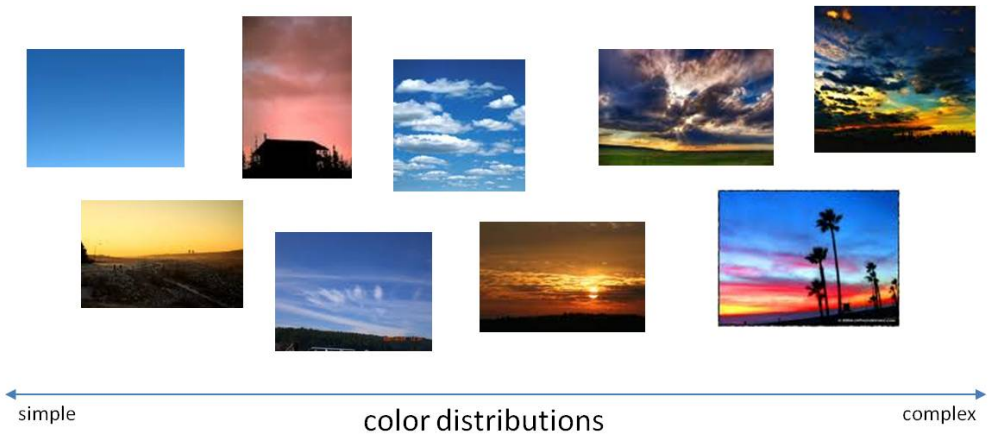


Figure 1.7: A range of color distributions in the image of the sky

est and use these regions as representatives of the similar and larger regions. Another method is to use image segmentation. Both regions inside swatches and segmented regions are expected to have simpler color distributions than the whole image. The regions can then be modeled by Gaussian models. We are curious about how to model nonlinear characteristics in a complex color distribution at once without user intervention or a pre-processing such as image segmentation. The linear method is sufficient to handle simple color distributions but not complex ones.

There are more and more demands from users to control some processes of color transfer. The existing techniques of interactive color transfer provide opportunities for a user to define regions of interest, to have reference regions from different target images, to map region pairs of source and target regions, and to visualize the result image. These controlled processes are closely associated. One change in one process would affect another process. This interdependence motivates us to break the processes down in order to make the interactive color transfer method more flexible and responsive to changes.

Giving a user more control may introduce human-error. For example, a less-skilled user may put strokes in wrong places when define regions, resulting in regions that are mixed up or imperfect. Manual region mapping by a user may cause an unexpected or undesirable result which needs to be revised. We need to find a balance between the automatic process and the manual process. The interactive approach in color transfer would be semi-automatic. The algorithm would provide some decent results as

choices and then let a user select or manipulate the choices for the final result.

1.2 Problem Statement

Our main problems in this thesis are how to improve color transfer techniques that can handle complex color distributions and enhance interactive color transfer methods with a balance between user controls and automatic processes. To solve these problems, we are considering the following subproblems.

- A user may expect to apply color transfer to a wide range of images including natural images which constitute a large proportion of photographs taken by general public these days. The natural images, for example: photographs of plants, animals, or natural landscapes usually have complex color distributions. Linear color transfer usually generates color artifacts when applies to images with complex color distributions because Gaussian models commonly used in the linear transfer are not sufficient to model complex color distributions.

- Color transfer between the whole images are intuitive as early introduced by Reinhard et al. [3]. However, a user may need to apply color transfer to just some parts of an image. As shown in Fig. 1.8, color transfer can be applied to the whole image (a) or just some parts of the image such as the sky region (b).

(a) Whole image (b) Parts of an image

Figure 1.8: The whole image versus parts of an image in color transfer

Swatch tools or image segmentation techniques are used to identify local regions in order to simplify the complex color distribution of the whole image into smaller

parts or regions with simpler color distributions. In general, the regions are expected to have simpler color distributions which can be modeled by Gaussian models and handled with a linear method of color transfer. Nevertheless, to perform color transfer between regions involves additional tasks such as region definition, region extraction, region mapping, region color transfer, and result recomposition.

- Most existing color transfer techniques are automatic. However, a user increasingly demands more controls in, for example, selecting target images to be used as reference images, defining regions in an image, picking regions in a source image to perform color transfer, and revising the unsatisfied result. The user controls give a user more options to generate the desired results.

  One piece of jigsaw that drives the interactive color transfer process is a set of interactive tools that must be designed and included in the user interface. The algorithm of the color transfer technique also depends on the interactive tools used. Such algorithm that makes use of the interactive tools is another crucial key to success.

  Existing techniques in interactive color transfer allow a user to get involved in a sequence of color transfer processes. By using manual control approach, the consecutive processes are tightly coupled. The interactive color transfer method using this approach is not flexible to revise the result. Especially, to manually manipulate regions, a user uses a trial and error strategy which can generate unexpected regions within the result image and can cause the whole color transfer process to be performed again in order to revise the result.

1.3 Research Objectives

Our research objective is to design and develop two color transfer techniques that can better handle the problem concerning complex color distribution.

1. The first technique is a global color transfer technique that automatically generates the result without inputs from a user. The whole image will be modeled in high

dimensional space using a nonlinear analysis technique. Color transfer is then performed by color matching.

2. The second technique is an interactive color transfer technique using the exploration approach. This method is a semi-automatic method that automatically generates initial results, and lets users make a series of decision including refining the result or manipulating the rest of the processes in order to get the desired results.

1.4 Our Approach

We propose two color transfer methods to address complex color distributions. One is an automatic approach and the other is an interactive approach. Overviews of these two techniques are explained below.

Global color transfer in the high dimensional space is a global and automatic method that aims to handle complex color distributions without user interaction. A complex color distribution usually has a nonlinear structure which is not suitable for representation by a Gaussian model. Our method uses a nonlinear analysis to capture the nonlinear color structures of source and target images in the high dimensional spaces or the image spaces. Source and target pixels are projected to their own image spaces where color transfer is performed by color matching. The color matching method searches and matches the closest pairs between source and target pixels.

Interactive color transfer by region exploration is a semi-automatic method which generates series of results for a user to consider and repeatedly make decision. The method is separated into two processes, the region definition process and the region exploration process, in order to reduce dependent errors between these two processes.

The region definition automatically defines local regions of an image as well as extracts these regions and their models. A user can accept the results or refine them by region merging or region marking features of the application. The region exploration is an interactive and iterative process, that widens the opportunities for a user to navigate back and forth with different choices of source regions and see the results from each region mapped pair. Without specific region mapping rules, a user evaluates the results by his or her own opinion. Moreover, a user can get more results from a large number

of possible results as a result of the exploration process.

1.5 Scope of Thesis

The scope of this thesis can be described as followed.

- As mentioned before, there are many characteristics that can be transferred between images or regions. This thesis is focusing on the color characteristic. The color is an important feature that largely determines looks and feels of an image and is naturally perceived by human.

- The proposed color transfer techniques aim to handle natural images which usually have complex color distributions.

- Since color transfer is a pixel process, the computational time and the memory usage depend on the number of pixels or the image size. The algorithm of color transfer in the high dimensional space in Chapter 3 quadratically demands the computation and memory usage when the image size increases. The image size in our experiments with this algorithm is restricted to 120×160 pixels.

1.6 Thesis Contribution

One complication in global color transfer is that linear color transfer in images with complex color distributions usually produces color artifacts in the results. A simple solution is to break the image into regions with simpler color distributions using tools like swatches. We proposed an alternative and novel technique coined global color transfer in high dimensional space using nonlinear analysis to solve the complex color distribution problem automatically without additional contribution from a user.

Our interactive color transfer method , called region exploration takes a different viewpoint from previous interactive color transfer techniques that involving region defining, region extracting, region mapping, and region color transfer. Instead of using region mapping criteria as in local color transfer or manually mapping region pairs by a user, the region exploration method proposed here lets a user explore all possibilities of region mapped pairs of source and target regions. The advantages of this approach

include abilities to preview the results, evaluate them, and eventually select the results that the user prefers. Our approach has two key features described as followed.

- Defining local regions using strokes as found in the existing interactive color transfer techniques is dependent on the quality of the user inputs otherwise the resulting regions may be corrupted. Moreover, the resulting region may still have a complex color distribution if a user place a stroke across areas with different color distributions. Human errors in region definition are overcome in our approach by automatically generating initial regions for a user. Hence, a user may refine the initial results to get more meaningful results using the features of region merging and region marking. These two features enable a user to get the meaningful regions without increasing the complexity in color distributions.

- Our approach takes advantages of separating the region exploration process from the region definition process. Region exploration focuses on region color transfer so that it is responsive and effective. It is easier to revise the result since there is no coupling between region definition and region color transfer. Comparing with a wildly used stroke technique, user selection by clicking on the results of interest is a simpler interactive method.

1.7 Thesis Outline

- Chapter 2 gives a literature review in color transfer techniques. The existing techniques are analyzed and classified into three groups: global color transfer, local color transfer, and interactive color transfer. Our proposed color transfer techniques in Chapter 3 and 4 are based on the fundamental knowledge in this chapter.

- Chapter 3 will introduce a global color transfer technique in the high dimensional space. This technique aims to handle natural images that usually have complex color distributions. The nonlinear characteristics of the complex color distributions in RGB color space will become linear and can be handled more easily in the high dimensional space.

- Automatic methods give users no options to manipulate local regions of an image.

Chapter 4 will demonstrate the interactive color transfer technique using the exploration approach. This method gives a user more options to define regions, to explore the results from region mapped pairs, to evaluate the results while exploring, and to get more desired results.

- The thesis conclusion is given in Chapter 5. The appendix provides more details in the experimental programs used in this thesis.

CHAPTER II
LITERATURE REVIEW

2.1 Color Transfer

The ability to make changes or manipulate the image content is important in the image editing task. Color content in an image is important to the human interpretation. Therefore, there has been research work in the past to manipulate color information in the image. Color transfer is one simple idea with the goal to change a color tone in one image (a source image) to look like another image (a target image).

Color is one of appearance characteristics of an image. In term of data analysis, color of a pixel is a point in n-dimensional space; e.g. RGB and YUV for $n = 3$, CMYK for $n = 4$. Because an image consists of a number of pixels; therefore, the colors in the image we perceive come from a distribution of each pixel color. To refer colors in an image, color model is needed. Color distribution is a color model used often in many image applications. Color transfer application uses color distributions as color models of a source and target images. It also needs the transfer function to transforms the color distribution of a source image to match that of a reference image or a target image as identically as possible. In general, a framework of color transfer can be considered in three steps:

1. **Model definition:** a color model of an image or a local region is first defined. The image colors is described by the model parameters. For example, a color model defined by a Gaussian model has a mean and a standard deviation as model parameters.

2. **Model extraction:** the model parameters of each source and a reference images need to be extracted from their own pixels. The model extraction techniques are vary and depend on the definition of the model.

3. **Color transfer:** the last step is to transfer the colors of the reference image to the source image using a transfer function. The transfer function can be linear or nonlinear and usually involves with the model parameters from both images.

Color transfer techniques have been developed for a while. We analyzed the existing techniques and categorized them into 3 types. One calls a global method if a source image is performed color transfer for the whole image while a local method is called if local regions of the source image are performed separately. The global and local methods are usually automatic. The last type, on the other hand, is an interactive method which extends the local method by letting a user get involved in some processes via interactive tools. Next sections will go into details of each category.

2.2 Global color transfer

Global color transfer is usually an automatic and example-based method. This technique aims to transfer colors of a source image to have its color appearance close to that of a reference image (a.k.a. a target image) as a whole image.

2.2.1 Global Color Transfer Framework

Figure 2.1 illustrates a framework of global color transfer in general. A whole color appearances of any image can be represented by a color model. To perform global color transfer between a source and a target images, each color model of both images needs to be extracted from all image pixels. Instead of tuning model parameters manually, the color transfer function uses the model parameters from both source and target images in the color transfer function in order to change color values of source pixels. Modifying source pixels will eventually change the color model of the source image to match that of the target image.

There are many existing techniques of global color transfer. The differences are in defining the color models and the transfer functions. In general, a color model of an image can be categorized into two types; i.e., a simple color model usually modeled by a simple Gaussian model; and a complex color model or a non-Gaussian model. The transfer functions depend on the color models used. The linear transfer function is fast

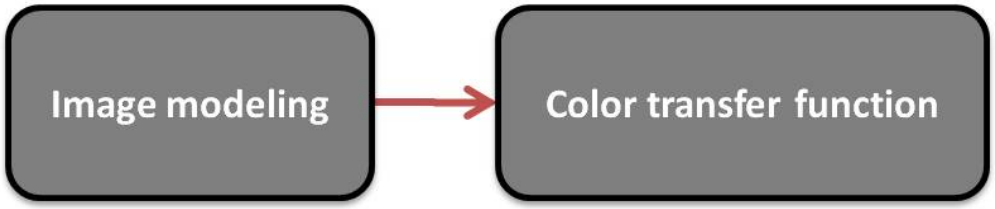


Figure 2.1: Global color transfer diagram

and effective especially with Gaussian models while the more complex transfer function, usually nonlinear, can handle complex color models better than the linear transfer function.

2.2.2 Global Linear Color Transfer

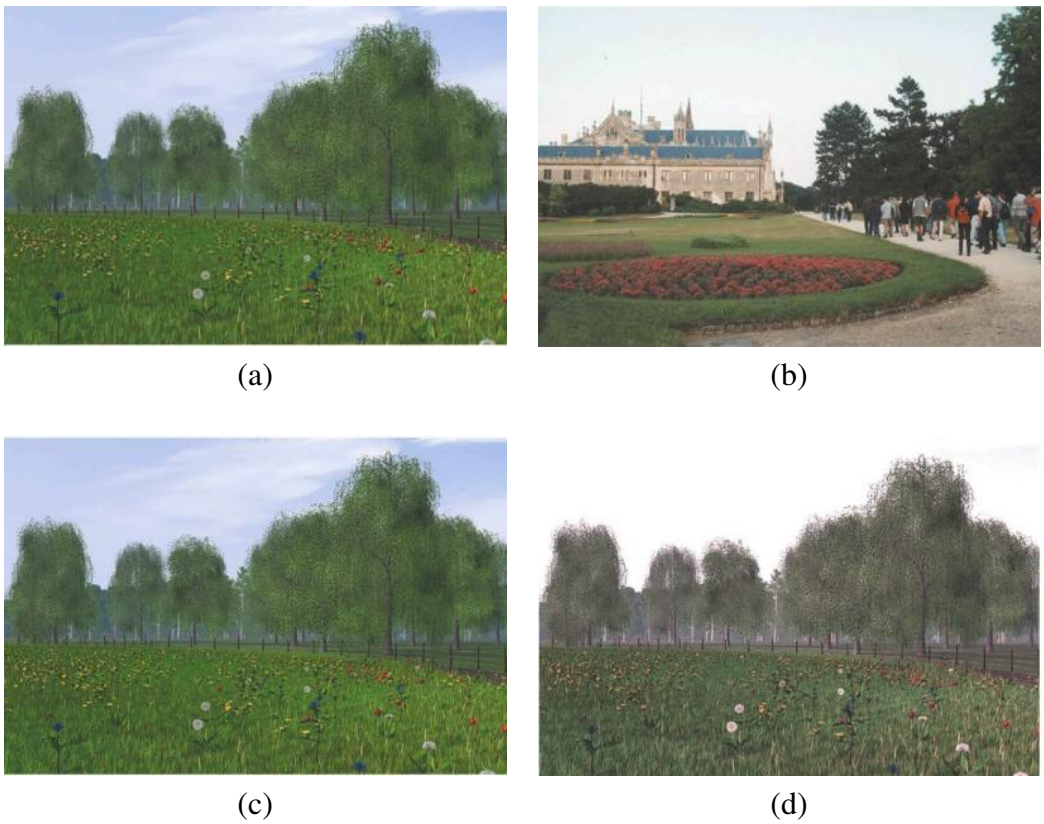
Global linear color transfer is a global method that uses a linear color transfer function. Figure 2.2 illustrates a diagram of global linear color transfer. In order to use a linear color transfer function, pixels must be in an uncorrelated color space. Since the RGB color space is a correlated color space, any operation on one color channel will affect to the other channels. Therefore, for each of source and target images, the first step is to convert image pixels in the RGB color space to an uncorrelated color space. The color model in the uncorrelated color space is then extracted. Hence, the color transfer function can be performed on each uncorrelated color channel separately without any effect to each other.

Figure 2.2: Global linear color transfer diagram

Reinhard et al. [3] may be the first group of pioneers in color transfer. Their technique models pixel colors by Gaussian models in $\ell\alpha\beta$ color space [15] rather than the RGB color space. The choice of color space used in color transfer by a linear function is important. The RGB color space is a highly correlated color space. The pixel value in each channel of RGB are dependent to each other. In any correlated color space

like RGB, the effect of performing the linear transform on one channel will cause the other channels to change too. Reinhard et al. transformed the source and the target images from RGB to $\ell\alpha\beta$ in order to de-correlate the color data before performing the linear transform on each channel separately.

In the Reinhard et al. paper, Fig. 2.3 shows the resultant comparison of color transfer between a source image (a) and a target image (b) that performed in different color spaces. Figure 2.3.(c) shows the result in the RGB color space whereas (d) in the $\ell\alpha\beta$ color space. The effect of correlated color space can be seen in the result in the RGB case which is almost the same as the original source image while the color appearance of the result in $\ell\alpha\beta$ looks more like that of the target image.



(a) (b)

(c) (d)

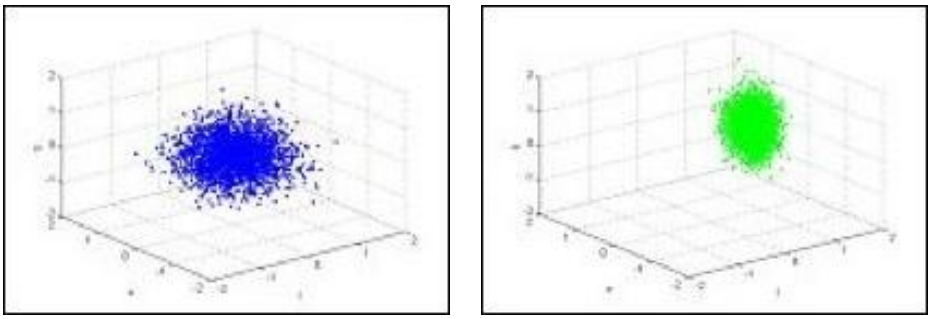
Figure 2.3: Color space comparison in color transfer: RGB versus $\ell\alpha\beta$

The linear color transfer function performs scaling and shifting in the color distribution of the source image to match that of the target image. By this technique, each source pixel color is scaled and shifted by the model parameters. Such parameters are standard deviations and means of the source and the target color distributions.

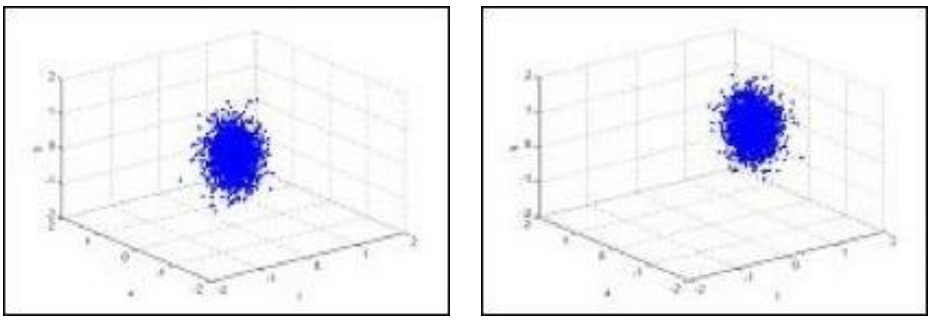
The color transfer algorithm from Reinhard et al. simply computes a color mean (μ) and a standard deviation (σ) of the color distribution from each source (I_s) and target (I_t) images in the $\ell\alpha\beta$ color space. For each source pixel (p_s), the resultant pixel (p'_s) can be derived by the linear function in Eq. 2.1 which is computed for each color channel, separately.

$$p'_s = \frac{\sigma_t}{\sigma_s}(p_s - \mu_s) + \mu_t \tag{2.1}$$

From Eq. 2.1, each mean of the source and target images is used as a center of each distribution representing the position of the whole distribution in the color space. The means are used to shift the source distribution to the target distribution. It also implies that every source pixel is moved to the new position in the same distance. Each standard deviation represents the size of each distribution and is used to scale the source color distribution to match the target one. Figure 2.4 illustrates the linear color transfer between the source color distribution (a) and the target color distribution (b) by scaling the source color distribution (c) and then shifting it to match the target color distribution (d).



(a) (b)



(c) (d)

Figure 2.4: Shift and scale in the linear color transfer

One key point when performing linear color transfer function with pixels in the RGB color space is to de-correlate pixels before performing color transfer. The $\ell\alpha\beta$ color space is not the only way to make a color space uncorrelated. Abadpour et al. [16] [17] and Kotera et al. [18] used Principal Component Analysis (PCA) to de-correlate pixels in the RGB color space. In theory, PCA is used to project pixels to a new orthogonal space which is also an uncorrelated space. PCA is also a technique for dimensionality reduction with minimum errors. To take this advantage on color transfer, pixels represented in 3-dimension color space can be transformed to one dimensional data which are then performed color transfer by the linear transfer function. Figure 2.5 shows an example of PCA-based color transfer.

(a) source image (b) target image (c) result image

Figure 2.5: PCA-based color transfer

Another PCA-based color transfer is a scene-referred color transfer is shown in Fig. 2.6. This example also illustrated a case that color transfer in the $\ell\alpha\beta$ color space did not give a good result Fig. 2.6.(c) comparing with the result Fig. 2.6.(d) using algorithm in Kotera et al. work.

Xiao and Ma [19] used the statistical method to decompose a covariance matrix of pixels in the RGB color space using Singular Value Decomposition (SVD). Color transfer is performed by transforming pixels (3D vectors) with three types of transformation matrices, i.e. the transition matrices derived by the color means, the scaling matrices derived by the standard deviations, and the rotation matrices derived from the SVD algorithm. Similar to PCA, the rotation matrix transforms pixels into the uncorrelated space. Figure 2.7 shows an example using this technique.

(a) source image (b) target image (c) Reinhard result (d) result image

Figure 2.6: Scene-referred color transfer

(a) source image (b) target image (c) result image

Figure 2.7: SVD-based color transfer

2.2.3 Global Nonlinear Color Transfer

The linear color transfer technique used in Reinhard et al. work is suitable for a uni-modal distribution that has its shape roughly in an ellipse. This is not the case when an image has a multi-modal distribution in an irregular shape. Such a case, the linear transform will transfer the wrong colors in the wrong areas, especially when a source and a target color distributions have the unbalance number of modes. Moreover, for either source or target color distribution that has the unbalance numbers of pixels in each mode, colors in the dominant modes will influence the transfer process more than the ones in the minority modes.

Swatches can alleviate the errors in some situations. The idea of swatches is simple. A swatch is defined by a rectangle and simply considers a color distribution inside the rectangle as a representative for similar areas. Using swatches is a supervised method that requires some skills and understanding from a user to define regions correctly.

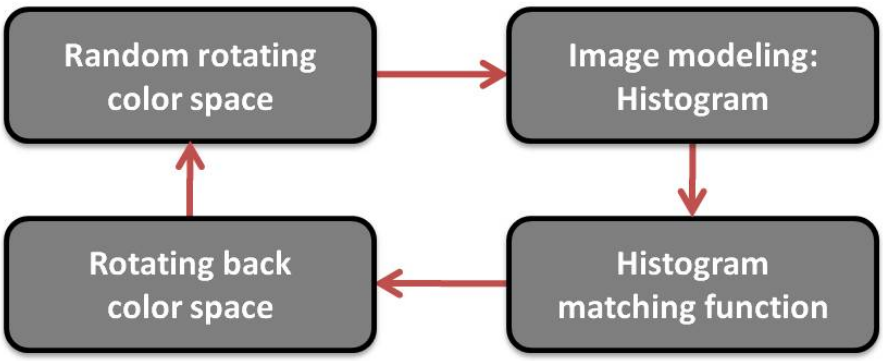


Figure 2.8: Global nonlinear color transfer diagram

Figure 2.8 illustrates a diagram of global nonlinear color transfer found in Pitié et al. work [2]. This is an iterative method that makes the source color distribution evolved in each iteration and eventually converged to match the target color distribution. Similar to the linear one, the first step in each iteration is to convert image pixels in the RGB color space to an uncorrelated color space using a random rotation matrix. A color model in the uncorrelated color space is a color histogram. Hence, the histogram matching function can be performed on each color channel separately without any effects to each other. The result in each iteration needs to rotate back with the same rotation matrix at the beginning of iteration in order to get back to the original coordinate system. The process will converge if get enough random rotation matrices.

The color distribution can be represented by a color histogram. Color transfer can then be thought of as histogram matching or histogram transfer [20]. Histogram transfer is a nonlinear and non-parametric transfer technique. When considering a one dimensional histogram transfer, a simple solution is a histogram mapping based on a histogram equalization method [21].

Figure 2.9 illustrates how to map data u generated from a probability density function (PDF) f to data v generated from PDF g using their respective cumulative PDF (CDF) F and G. The figure gives one example which data u and v are distributed in range $[0, 255]$, and F and G are normalized into $[0, 1]$. This example then yields the expression for the histogram mapping function h as:

$$\begin{aligned} v &= h(u) \\ &= G^{-1}(F(u)) \end{aligned}$$

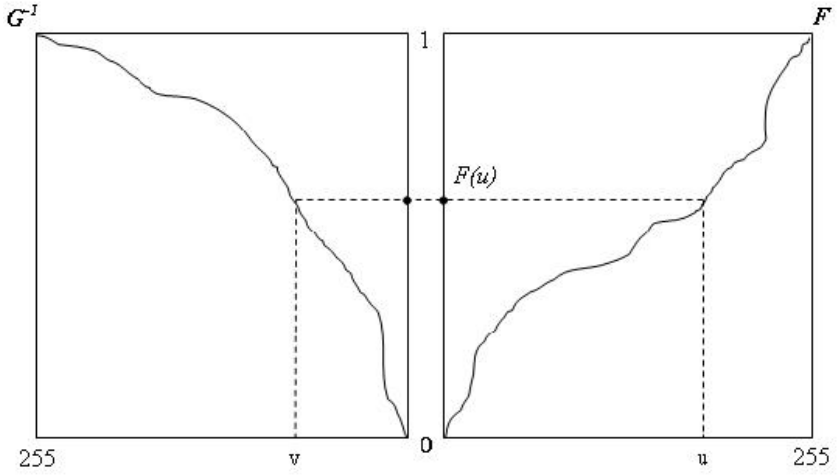


Figure 2.9: Illustration on 1D histogram mapping

However, when considering the application of color transfer between images, the solution is not simple because the color distribution is in three-dimensional space. Directly applying the histogram mapping on each axis of the space does not work as expected because of the correlation between axes. The ND-PDF transfer technique by Pitié et al. applied N-dimensional Radon transform to the N-dimensional PDF transfer before handling 1-dimensional marginal PDF transfers.

Figure 2.10 illustrates the 2-dimensional PDF transfer. The algorithm starts with rotating the color space of the source and target images by a random rotation matrix, and then marginalizing the rotated color data of each axis. For each marginal axis, the histogram transfer is performed. The above-mentioned steps are repeated until the result distribution converges to the target distribution. The figure shows the source distribution in the early iterations. The shape of the source distribution is gradually changed to closely match the target distribution as shown in the 20^{th} iteration. The prove of convergence is given in the Pitié thesis [22]. The figure also shows that the convergence actually happened before the 10^{th} iteration as shown in the KL distance (bottom right image).

Once the color space multiplied with a rotation matrix which is an orthogonal matrix, the rotated color space becomes the orthogonal space which has independent axes. The rotation makes data on one axis independent or uncorrelated to the others. The 1D histogram transfer can then be performed on each axis separately. The nonlin-

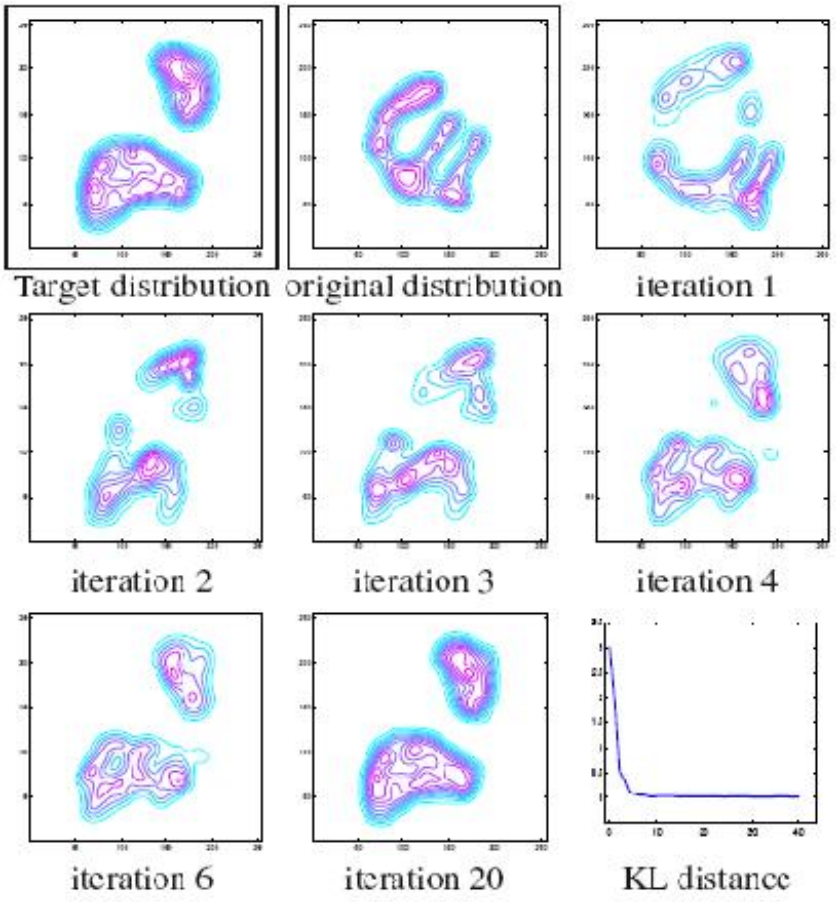


Figure 2.10: Illustration on 2D histogram transfer [2]

ear transfer does not assume the shape of color distribution to be sphere and does not require model parameters–the mean and the standard deviation. The nonlinear evolution of each iteration gradually change the source distribution to match the target distribution finally. Figure 2.11 shows an example using the Nd-PDF color transfer technique.

When contents or dynamic ranges of both images are too different, the result from the mapping function can be stretched on some parts, and its noise level is also enhanced. The unbalance number of modes or the unbalance number of mode's members cause increasing color artifacts or the graininess of the original image. To solve this problem, Pitié et al. introduced the second stage of the method in the later paper [23] to reduce the artifacts through an efficient post-processing algorithm that intends to preserve the gradient field of the original image.

Figure 2.12 shows the result (c) which grain noise is enhanced in the original version of Nd-PDF color transfer. The result (d) uses the modified version with a

(a) source image (b) target image (c) result image

Figure 2.11: Nd-PDF color transfer

gradient smoothing which gives a better result.

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

Figure 2.12: Grain artifact problem in Nd-PDF color transfer: (a) a source image, (b) a target image, (c) a result image in a normal case, (d) a result image with gradient smoothing

2.3 Local color transfer

When handling with a complex color distribution, a simple Gaussian model is not sufficient to be a color model. As explained in the previous section, some methods are used to solve this problem, i.e., using a nonlinear color transfer function or using swatches given by a user.

The key point of above solutions is to reduce the complexity in the color distribution, or to come up with a complicate technique that can handle a complex color distribution but usually takes a long time to compute. The purpose of swatches is to model colors in specific and smaller regions which expected to have simpler color distributions. These models are then used as representatives for similar regions. However, using swatches requires a user's skill and understanding in order to succeed in such problems.

2.3.1 Local Color Transfer Framework

Local color transfer shares some ideas of the global color transfer, but in the smaller scale. Instead of transferring colors for a whole image, the local method will transfer colors between local regions of source and target images.

Figure 2.13 illustrates a framework of local color transfer introduced in Tai et al. work [24] [25]. Local color transfer scales down the process from a whole image to several local regions. Additional tasks to the global approach are region definition, region extraction and region mapping rules. Some image segmentation techniques can be used for defining and extracting regions. The extracted regions is then modeled. With more regions to be considered, mapping regions between a source and a target regions need to be done before the color transfer process begins. Local color transfer function between regions is similar to global color transfer except using weights in the transfer function since a pixel can belong to many regions with different weights. Once the transfer completed for every source region, all regions are recomposed to be a final product.

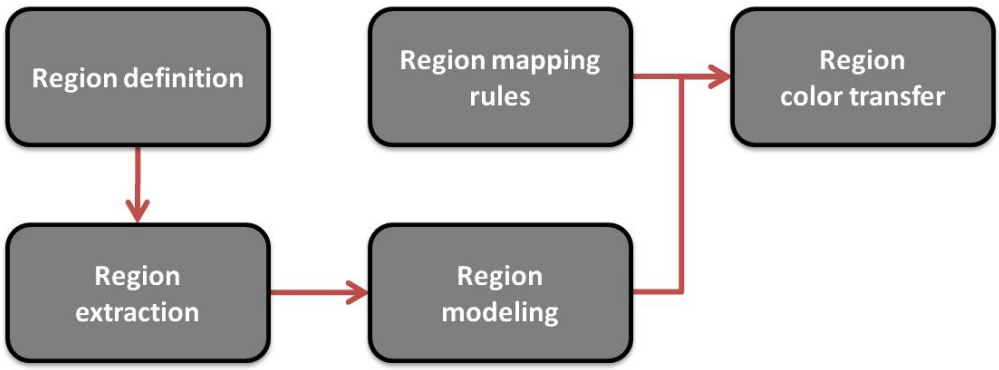


Figure 2.13: Local color transfer diagram

The local method gives many benefits when handling with complex color distributions. Each local region has a simpler color distribution which can be modeled with a simple Gaussian model. Hence, color transfer between regions can use a linear transfer function which is fast and effective.

Tai et al. introduced a local color transfer technique. They defined local regions as Gaussian mixture models (GMM) derived by Expectation-Maximization (EM)

algorithm.

After EM estimation, a source image I_s and a target image I_t will be associated with GMM denoted as $\{G_{s_i}|0 \leq i \leq N_s\}$ and $\{G_{t_j}|0 \leq j \leq N_t\}$, where N_s and N_t are the total number of components in I_s and I_t respectively.

Breaking down an image into local regions increases the number of color transfer between the source and the target regions. Tai et al. used a rule-based method for region mapping. Since humans are more sensitive to the luminance channel, the monotonic constraint should be maintained in this channel. Specifically, suppose a pair of Gaussian components $G_{t_i}(t_i; \mu_{t_i}, \sigma_{t_i})$ and $G_{t_j}(t_j; \mu_{t_j}, \sigma_{t_j})$ where $\mu_{t_i} \geq \mu_{t_j}$ in the luminance channel. Then, the transferred Gaussian means μ' should also satisfy $\mu'_{t_i} \geq \mu'_{t_j}$. The rules for mapping G_{t_j} to G_{s_i} are valid if:

- The means in luminance channel of G_{t_j} and G_{s_i} are closest.
- A monotonic constraint is enforced simultaneously.

Since each model contributes in different proportions on each pixel, the probabilities of each pixel can be used as weights for the image composition. After obtaining the mapping function $f(\cdot)$ which maps Gaussian component from G_{t_j} to G_{s_i}, the final transferred color in a pixel $I_t(x, y)$ is computed as:

$$I'_t(x, y) = \sum_j P_{xy}^{t_j} \left(\frac{\sigma_{s_i}}{\sigma_{t_j}}(I_t(x, y) - \mu_{t_j}) + \mu_{s_i}\right) \tag{2.2}$$

where $P_{xy}^{t_j}$ is the probability of the pixel $I_t(x, y)$ that belongs to the region t_j.

Defining local regions in local color transfer reduces the complexity of color distribution for each region. This also means decreasing color mixed-up in each region. The probabilistic segmentation also help to reduce the spurious colors around the boundaries of each region. This is because the final color of each pixel include probabilistic weights into the color transfer function, as shown in Eq. 2.2.

Figure 2.14 shows an example of local color transfer. Local color transfer between a source image (a) with its 3 regions (b)-(d) and a target image (e) with its 3 regions (f)-(h) returns a result image in (i). Using the same source and target images,

the result using global color transfer in (j) shows undesirable mixed up colors in leaves and the river.

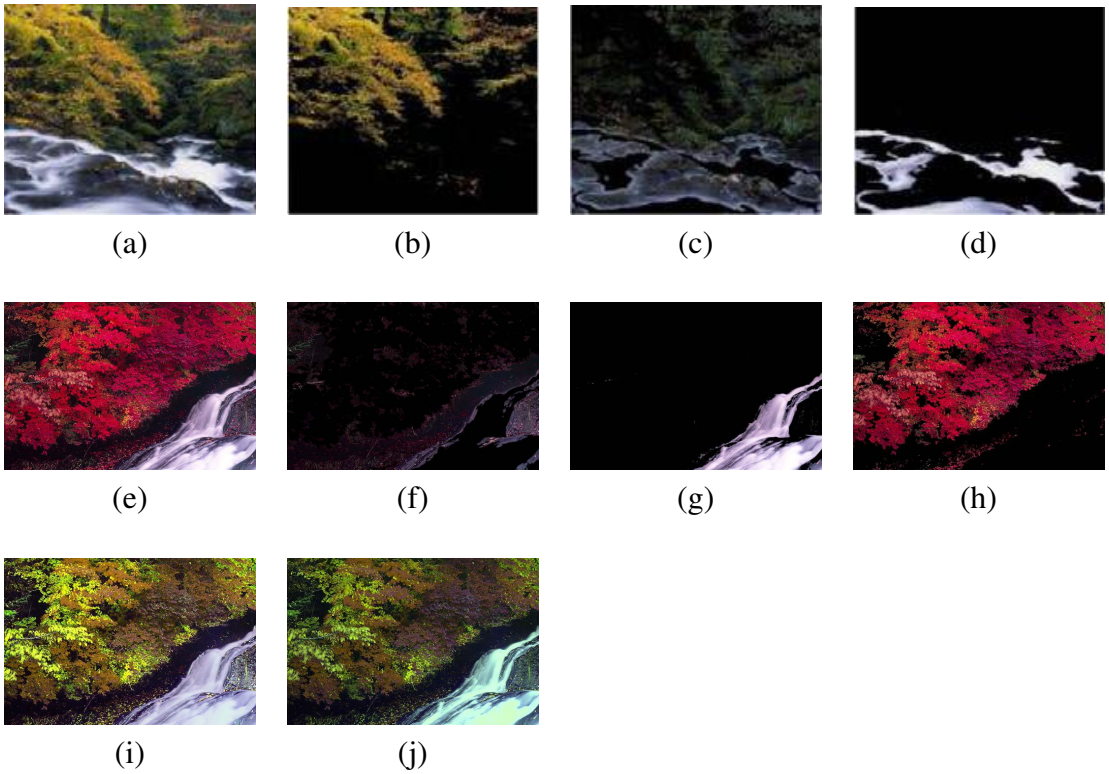


Figure 2.14: Local color transfer: the first row (a)-(d) is a source and its 3 regions, the second row (e)-(f) is a target image and its 3 regions, and the last row shows results by local color transfer (i) and by global color transfer, Reinhard et al. (j)

Xiang et al. [26] continued on Tai et al. work by adding more region mapping rules and more target images involved. They also explored in several evaluation methods of color transfer compared with a subjective evaluation.

There are some work in local color transfer trying to define local regions in different ways. Chang et al. [27] used pre-defined basic colors to classify pixels and group them as local regions. Both source and target images are segmented according to the basic colors. Each corresponding pair of basic color regions are then performed color transfer using a linear function. In another technique, Greenfield and House [28] had a similar idea with palette colors which are derived from image pixels instead of pre-defined colors and are not fixed or limited to the number of palette colors as in [27].

2.4 Interactive color transfer

Global and local color transfer techniques are automatic methods without interactions from a user. The local approach scales down the area from a whole image to smaller local regions and then performs color transfer between local regions instead. The approach also resolves the problem of complex color distributions in global color transfer as mentioned above. Interactive color transfer extends the idea of local color transfer by adding user interactions via interactive tools.

The interactive approach gives a user more options to manipulate the regions of interest. Color transfer should not be limited to a process between two images. Color transfer between regions lets a user change colors for specific source regions from target regions that possibly come from many target images. With more options to manipulate the source regions, a user can opt for which regions will be performed color transfer and which regions may be kept unchanged.

The color transfer effects of each region in global color transfer depend on the transfer function which derived from a whole image. On the other hand, the effects on local regions depend only on associated regions since the transfer function uses the region models instead of the image models.

Local regions can be derived from a number of segmentation techniques such as probabilistic segmentation–GMM, hard segmentation, soft segmentation–matting, and multi-labeling. In the interactive approach, the interactive tools are necessary and important to manipulate regions.

2.4.1 Interactive Tools

Perhaps, swatches have been so far counted as an interactive tool that lets a user to refine the result from the global method. Swatches are rectangle areas given by a user. They are used to scale down an image to specific regions of interest and also limit the color complexity only inside the rectangle areas. The usage of swatches is limited in many cases. Alternatively, strokes are recently used as interactive tools and widely accepted in the research community because of their flexibilities in size and shape and a simple usage. The stroke's size can vary. When the stroke's size is small, it is usually called a scribble whereas a brush refers to a large stroke.

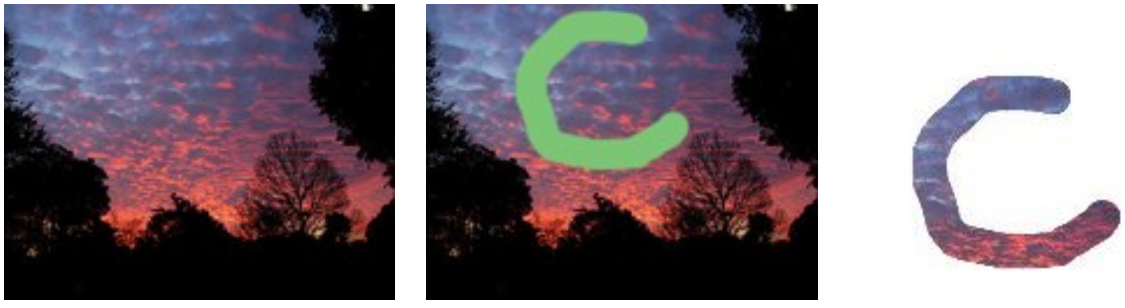
(a) Swatches [5] (b) Scribbles [29] (c) Brushes [30]

Figure 2.15: Flexibility of different interactive tools

Figure 2.15 shows a limitation of using swatches. A stroke is more flexible in shape and size than a swatch which is a rigid rectangle (a). In case of the spider web image, it is not feasible to mark the web network using swatches. Scribbles are more suitable, see (b). In case of defining large areas as in (c), brushes can be used instead. However, the time taken to define a large region using strokes will be relatively longer in comparison to swatches, especially when the rough region is sufficient to derive a model.

The influence of strokes in recent work can be described and listed as following.

- **Features under strokes:** strokes are used to extract features or characteristics such as color, texture, or style from specific areas where a user is interested. A user puts strokes on interesting areas, and the features are then extracted from the pixels under the given strokes.



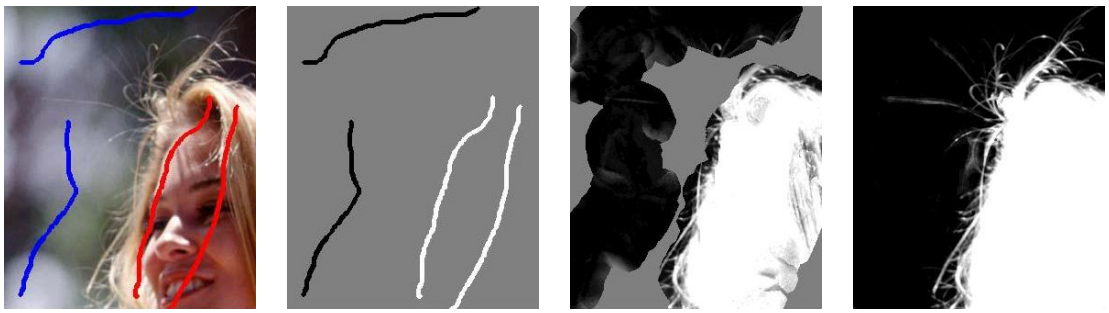
(a) (b) (c)

Figure 2.16: Feature extraction by strokes

Figure 2.16 illustrates the feature extraction using a stroke. A user simply puts a green stroke (b) on an image (a). The only pixels under the green stroke are extracted as shown in (c), and their features are extracted for use later on.

- **Strokes as initial inputs:** some techniques such as interactive matting by Wang et al. [29] and Levin et al. [31] use strokes given by a user as initial inputs in their algorithms. The techniques are usually evolved iteratively. At the first iteration, the pixels under strokes are feeded into the iterative algorithm as the initial data.

  Figure 2.17 illustrates the evolution of the interactive matting algorithm [29]. Given strokes on an image (a), the pixels under the strokes (b) can be extracted and proceeded as initial inputs in the first iteration. For the next iterations, the algorithm progresses to the next pixels further from the strokes' boundary (c), and propagates toward the rest pixels until covering all pixels in the image (d).



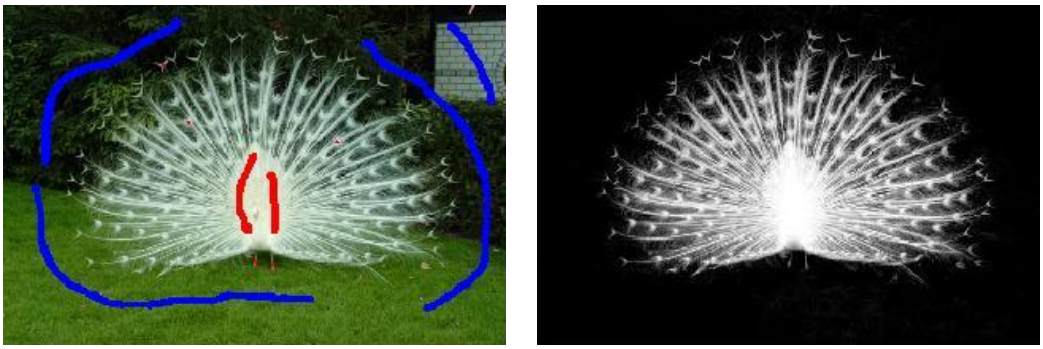
(a) (b) (c) (d)

Figure 2.17: Strokes used as initial seeds in the iterative algorithm

Figure 2.18 shows another usage of strokes in the closed form matting [31]. Given stroke labels on an original image (a), the red labels are used to define a foreground object and the blue labels for a background. The matting algorithm extracts pixels under the stroke labels as initial seeds, then iteratively evolve until converged, and at last return the result (b).

- **Regions definition with strokes:** labeling is a technique used to define local regions and often uses strokes as interactive tools. To segment regions according to labels, a user puts stroke labels in different colors on an image as guidelines for grouping regions together as a meaningful region or as a user's desire. The stroke color indicates the regions the stroke passing through are grouped into the same region.

  Stroke are recently used to define specific areas on an image or a set of images



(a) (b)

Figure 2.18: Stroke used as labels in the closed form matting

such as the work from Agarwala et al. [32] that uses strokes to define desired areas from a photo collection in order to perform photomontage as shown in Figure 2.19. Each photo in a collection (a) is labeled by a different stroke color as shown in (b). The photomontage algorithm allocates and composes each segmented region from the photo collection and return the result image (c).

(a) (b) (c)

Figure 2.19: Stroke labeling in the interactive digital photomontage

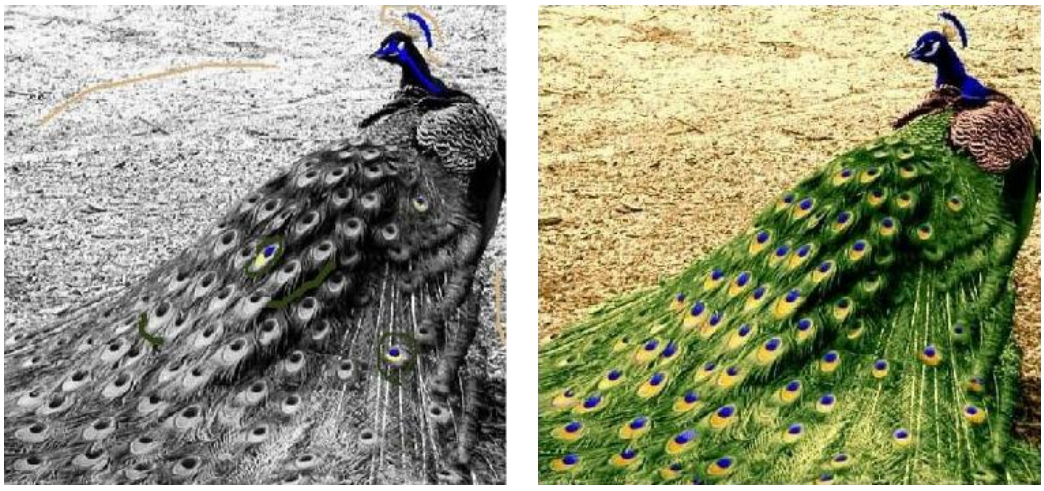
In the interactive approach, a user can make a meaningful region. For example, a user should label on a cloud area, a blue space, and the sun by the same stroke color in order to define a sky region instead of separately labeling on these three areas with different label colors. Figure 2.20 illustrates the usage of strokes labels in a matting application. The image in (a) is an original image before labeling. The red labels in (b) are put on a child and pumpkins to define one region or one object. The result of two segmented regions is shown in (c), i.e. the background region defined by the blue stroke and the foreground region from the red strokes.

In some cases, strokes are used to label the objects that are scattered around the image; for example, flowers in a field. A few strokes on some objects should

(a) (b) (c)

Figure 2.20: Defining an area by grouping different regions

be enough to let the segmentation process extract all the same objects and group them into one disjoint region. Figure 2.21 illustrates the usage of strokes to define the same type of objects scattered around an image in a colorization application. (b) shows the result of colorization using the given strokes from (a) as guidelines. Notice that it is unnecessary to put strokes on all scattered areas. Some of them are enough to perform colorization for the whole image.



(a) (b)

Figure 2.21: Defining the same type of objects scattered in multiple areas

2.4.2 Interactive Color Transfer Framework

Interactive color transfer gives more controls to a user with the interactive tools. In general, the interactive tools are used to define regions and map pairs of source and target regions. Regions are extracted according to the given user inputs. The extracted regions are then modeled before performing color transfer according to the already defined region maps.

In brief, the framework of interactive color transfer can be illustrated by the diagram in Fig. 2.22. Some blocks in the diagram apply for each source and target image and some apply for both images together as the context implied below.

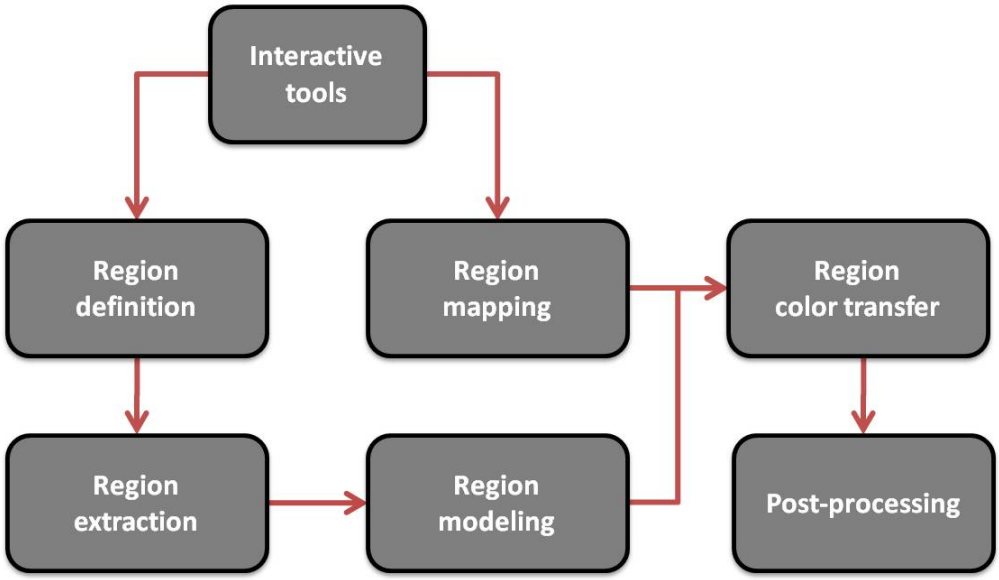


Figure 2.22: Interactive color transfer diagram

Interactive color transfer lets a user involved in many processes via interactive tools. The interactive tool is strokes in this case. For each source and target images, a user puts strokes as guideline inputs in the region definition process and the region mapping process. The local regions distinguish from each other by using different colors of strokes–labels. The region mappings also use the labels to map pairs of source and target regions; i.e., the same stroke color that applied on source and target regions indicates the mapping of these two regions. Given strokes, local regions can be extracted and modeled. Finally, given pairs of region maps and extracted models, region color transfer is then proceeded. However, some tasks in post-processing are needed in some

cases such as boundary smoothing when using hard segmentation. A user also has an opportunity to revise the result if not satisfied via the interactive tools.

Image segmentation with state-of-the-art techniques cannot provide semantic results automatically. For instance, to define a sky region means that clouds, empty space, and the sun are grouped together. Even image segmentation could separate these areas using features like colors, intensity, or textures; the segmentation algorithm does not know a meaning of each area. To solve this problem, the technique needs intelligence of a user to interact to the program; that is, the user must supervise or provide guidelines to the program in order to segment image according to the user's desire–the sky region.

From a point of view on programming and usage, a stroke tool can be used easily and efficiently to define local regions. Strokes are used to label expected regions by different label colors. The segmentation algorithm will use pixels under stroke labels as guidelines to achieve segmentation. Different label colors indicate the different expected regions.

There are different effects in interactive color transfer when using different types of image segmentation techniques. Soft segmentation techniques such as matting techniques, which return soft boundary regions, do not cause color artifacts around region boundaries whereas hard segmentation techniques do. The boundary problem can be solved by using gradient smoothing as found in [30], [33].

A user, not mapping criteria, decides which source regions should be mapped to which target regions. The stroke colors are also used to map regions. A user puts different stroke colors on both source and target images not only to define different local regions, but also define region mappings.

Local regions of both source and target images are extracted and modeled. Each local region is expected to have a simple color distribution which can be modeled by a Gaussian model. This assumption makes the process of color transfer between regions simple with a linear transfer function. Similar to the transfer function of Reinhard et al. in Eq. 2.1, the region color transfer algorithm computes color means (μ_{sr}, μ_{tr}) and standard deviations (σ_{sr}, σ_{tr}) of the color distributions from respective source regions (R_s) and target regions (R_t). For each source region pixel (p_{sr}), the resultant pixel (p'_{sr})

can be derived by the linear function in Eq. 2.3.

$$p'_{sr} = \frac{\sigma_{tr}}{\sigma_{sr}}(p_{sr} - \mu_{sr}) + \mu_{tr} \tag{2.3}$$

As mentioned before, the segmentation affects the color transfer process since all color transfer regions will be recomposed again at last. Soft segmentation provides weights which can be used as alpha values in the image composition [34]. On the other end, hard segmentation needs a smoothing filter to solve the color discontinuity around the region boundaries.

Luan et al. work [30] uses a brush tool to multi-label regions on both source and target images. It uses brush colors to distinguish between matching pairs of regions as well as special color for the region a user wants to keep untouched from the color transfer process. To solve the discontinuity on the region boundaries, it uses similar technique to Poisson image editing [35]. The technique uses a global optimization to minimize the gradient changes on the boundaries; however, it does not guarantee to converge because of no Dirichlet boundary condition.

Due to a manual region selection by a user, this paper has not shown experiments on natural images with complex scenes. The texture is also not considered in this work; therefore, it is up to a user to handle this problem. One example is shown in Figure 2.23. This example uses this technique to demonstrate the cosmetic effect that transfers make-up among faces of models.

Work in [36] is interactive local color transfer. However, it does not define an object or a specific area to take a transfer effect. Instead, all pixels in a source image are changed their values by the weighting technique. Each source pixel finds a distance between its color and the mean color of the marked area, and use the inverse of distance as its weight.

Figure 2.24 shows the comparison between original color transfer from Reinhard et al. [3] which transfers color between images, and the local color transfer which takes effects locally. The first row is target and source images with marked areas. The second row shows the differences in results between these two techniques, especially the ground color of the right column (d) shows unaffected in color changes while transferring color between sky areas of target and source marked areas. The comparison can

(a) (b)

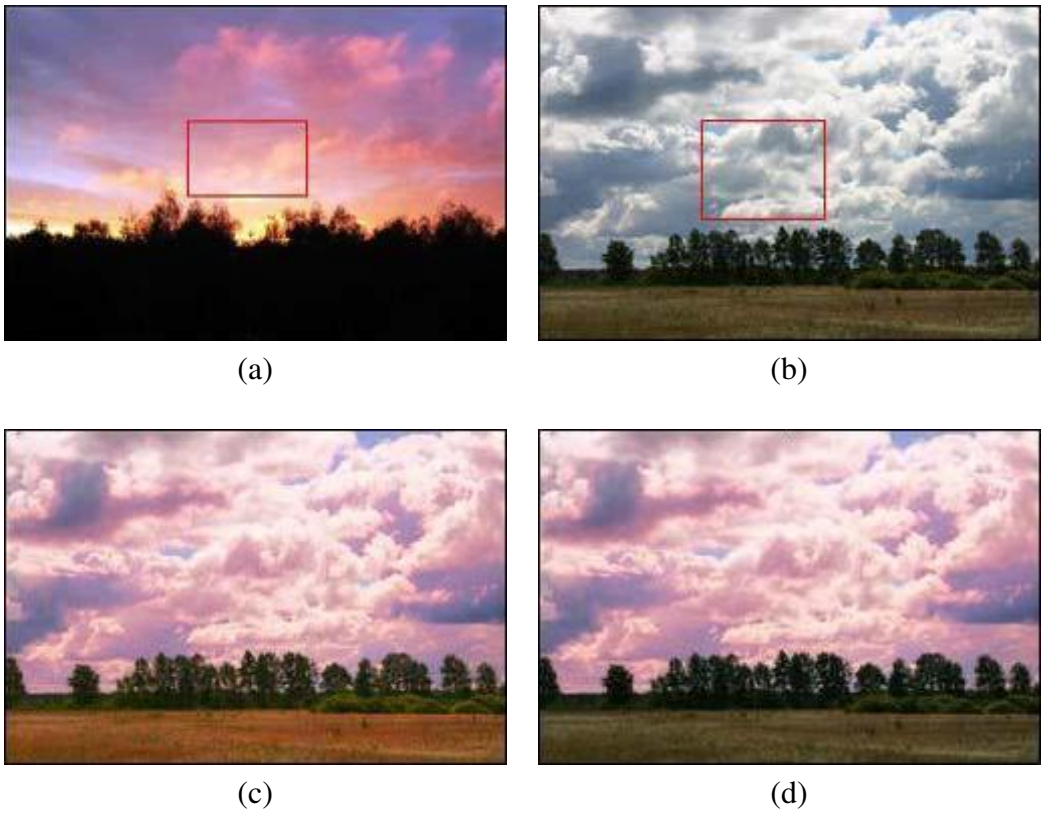
(c) (d)

(e) (f) (g)

Figure 2.23: Make-up transfer using interactive brush tools

show different effects between two techniques – global and local. However, it is not necessary to mark areas with the Reinhard work. It is possible to have statistical data in the marked area of the target image, but when transferring color by Reinhard technique, it still applies for the whole source image, not only in the marked area.

However, the drawbacks of this technique are that it is not a real local color transfer because there is no real boundary of the local area. It is possible to have a conflict if different objects have similar colors and have approximately the same distance from the marked area; for example, the similar blue colors of sky area and ocean area. That means the algorithm will transfer color to both two areas even though only one is intended.



(a) (b)

(c) (d)

Figure 2.24: Interactive local color transfer

Konushin et al. [6] also provides a stroke tool for labeling. However its segmentation process uses cellular automata, which starts the process from the pixels around the provided strokes, then the algorithm iteratively evolves the states of the pixels next to the stroke boundaries and spreads out to cover the whole image. Figure 2.25 shows the input image with strokes provided by a user as in the left image, while the right image shows the colorized result by the cellular automata technique.

This work also re-colorizes an image using the same technique with some modifications. Of course, the feature vector of each cell is color in stead of intensity. The outside areas of the object are marked with strength 1, while the inside areas, the user provides a new color by strokes. Figure 2.26 shows two examples of re-coloring – left column shows original images, central column shows how to keep original color unchanged outside object areas by using white marked pixels as constrains, and right column shows resulting images.

By this technique, users can specify an individual object; even there are the same objects, if not too many, in the other area of the image. Moreover, its refinement

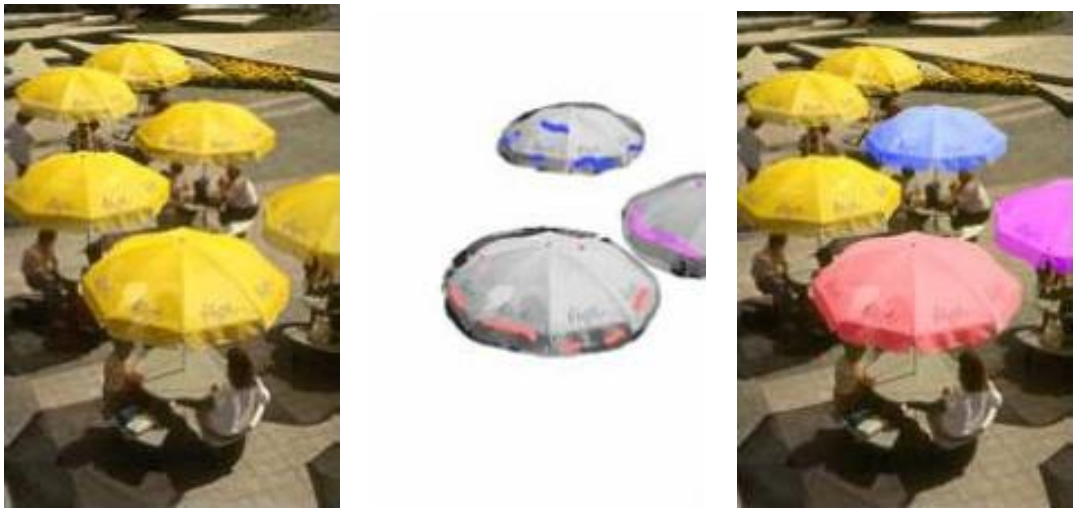
(a) (b)

Figure 2.25: Interactive image colorization

method is rather easy. After the initial colorization is computed, the user can refine it by marking additional pixels with some colors. Strengths of cells, corresponding to the marked pixels are set to 1, and their feature vectors are set to newly assigned colors. They start spreading and trying to occupy neighboring pixels until converged again. This refinement produces only a local change and is computed very fast.

The limitation of this work is color blending around boundary areas. A user takes responsibility to provide color stroke carefully and even more when handling with a natural image which has a complex scene and more expected boundary problems. To assign colors to the natural images can be time consuming as well.

Dong and Xu [37] continued the Konushin et al. work [6] in the re-coloring part, also using the same Growcut technique [38]. Unlike re-coloring, their method used an example-based method.



(a) (b) (c)

Figure 2.26: Interactive image re-coloring

CHAPTER III
GLOBAL COLOR TRANSFER IN HIGH DIMENSIONAL SPACE

3.1 Introduction

In chapter 2, the global color transfer technique introduced by Reinhard et al. [3] was discussed. The algorithm computes simple statistical data of the source and target color distribution in $\ell\alpha\beta$ color space. In that scheme, the whole image is defined as a single color model. Therefore, the method works well with images with simple Gaussian color distributions. However, many images have non-Gaussian or complex color distributions which may produce obvious color artifacts when using this technique.

In Reinhard et al. [3] work, swatches were used to handle the complex color distributions. The swatches are rectangles determined by a user and used to alleviate the complex color distribution problem by considering pixels within the swatches as the representatives for similar pixels.

In the swatch approach, a user first manually decomposes the source and target images into several pairs of matched swatches. The match is normally under a user's subjective observation of the color characteristics between the swatch pair, and the mapping is normally a one-to-one operation to ensure the simplicity of the manual process. Hence, the pixels within swatches are now modeled by Gaussian distributions separately. Finally, the color transfer algorithm is applied to these swatch pairs, generating the final image.

However, using swatches has some drawbacks. First of all, the user has to supervise some elaborate and trial-and-error work to generate swatches, both for the source and target images. The user has to match the swatch pairs manually. As a consequence, the quality of the result normally relies on intuition and experience of a user when locating the swatch pairs. Another drawback of using the swatch approach is that artifacts are normally visible on the boundaries between two swatches after the color transfer.

In this chapter, a novel global color transfer method is proposed. It uses a concept of nonlinear analysis, Kernel Principal Component Analysis (KPCA), to create a high dimensional space. KPCA [39] [40] [41] is a nonlinear generalization of Principal Component Analysis (PCA). The proposed method applies KPCA for the color transfer problem and aims to be a global and automatic method that can handle images with complex color distributions such as natural images without any helps from a user. The method yields improved results comparing to the other global color transfer methods as demonstrated in the section 3.6.

3.2 Related Work

The global color transfer technique by Reinhard et al. [3] performs linear transformations (scaling and translation) on the color distribution of a source image to match that of a target image. The technique is a statistical based method. The algorithm simply computes a color mean (μ) and a standard deviation (σ) of the color distribution from each source (I_s) and target (I_t) images. For each source pixel (p_s), the resulting pixel (p'_s) can be derived from the linear transfer function as in Eq. 3.1.

$$p'_s = \frac{\sigma_t}{\sigma_s}(p_s - \mu_s) + \mu_t \tag{3.1}$$

Cheng and Hsia [42] demonstrated the advantages of taking principal component analysis (PCA) on color image processing. Abadpour et al. [17] proposed PCA-based color transfer which is another global color transfer technique.

In PCA-based color transfer, a color pixel is considered as a color vector in three-dimensional space. PCA is used to capture an underlying structure of pixels by projecting pixels in the RGB color space to the orthogonal space. PCA can reduce the data dimensions with minimum errors so that the color transfer process can reduce computational time too.

To perform PCA-based color transfer, let the vectors μ_s and μ_t be the color mean vector of the source image I_s and I_t, respectively. V_s and V_t are PCA matrices derived by performing PCA on source pixels and target pixels, respectively. The PCA matrix contains the eigenvectors as its columns sorted by the eigenvalues in a descending

fashion. The color transfer is performed by computing the color vector c'_s as the result of recoloring the color vector c_s as in Eq. 3.2.

$$c'_s = V_t V_s^{-1}(c_s - \mu_s) + \mu_t \tag{3.2}$$

Similar to [17], Kotera et al.[18] also proposed a PCA method to the color transfer problem as follows.

$$c'_s = V_t^{-1} \cdot S_{st} \cdot V_s \cdot c_s \tag{3.3}$$

The proposed technique has two basic functions, *Axes matching* by rotating the pixel cluster along the eigenvectors, and *Variance matching* by scaling the color distribution along the principal component axes. V_s and V_t denote the rotation matrix for a source cluster and a target cluster. The scaling matrix S_{st} is a diagonal matrix with the entries of eigenvalues ratio given by:

$$S_{st} = \begin{bmatrix} \sqrt{\frac{\lambda_1^t}{\lambda_1^s}} & 0 & 0 \\ 0 & \sqrt{\frac{\lambda_2^t}{\lambda_2^s}} & 0 \\ 0 & 0 & \sqrt{\frac{\lambda_3^t}{\lambda_3^s}} \end{bmatrix} \tag{3.4}$$

where λ is a eigenvalue. The superscripts s and t denote the source and the target images, respectively.

Xiao et al. [19] demonstrated another color transfer technique in the statistical based method. In this method, pixels are represented as data points in 3-dimensional space (RGB color space). The color transfer is a process that transforms data points of source image I_{src} by translation, rotation and scaling transformations to fit data points' cluster of target image as in Eq. 3.5. Notice that the order of transformations is important, not interchangeable.

$$I'_{src} = T_{tar} \cdot R_{tar} \cdot S_{tar} \cdot S_{src} \cdot R_{src} \cdot T_{src} \cdot I_{src} \tag{3.5}$$

where T, R, and S are translation, rotation, scaling matrices respectively. The subscript src is for the source image while tar for the target image.

3.3 Color Matching Approach

Images with simple color distributions are modeled by Gaussian models with means and standard deviations which can be used effectively in the linear color transfer method, as in Reinhard et al. work [3]. However, images with complex color distributions are not suitable for linear color transfer since the images can not be modeled by simple Gaussian models.

Existing global color transfer techniques cannot cope with complex or non-Gaussian color distributions. To solve the nonlinear characteristic of a complex color distribution, we propose a technique called color matching using the nonlinear analysis, KPCA.

To illustrate an idea of color matching, we begin with a simple case in a linear projection space derived by the linear analysis, PCA. Assume that the source and target images have simple color distributions that can be modeled by Gaussian models. The algorithm of color matching using PCA can be listed as following.

- Create the eigenspaces for each source and target images by solving the eigenvalue problems of each image.
- Source and target pixels are then projected into their own eigenspace in order to create their own projection space.
- For each source pixels, search for the closet target pixel in the projection spaces using similarity distance as a measure.
- Trace the closest target pixel back to the RGB color space in order to match the corresponding target pixel for the source pixel in question.

For images with complex color distributions, PCA cannot capture nonlinear color structures. Therefore, we propose color matching in high dimensional space. A complex color structure in the RGB color space is expected to become a simpler and linear color structure in the high dimensional space.

Similar to the algorithm of color matching by PCA, color matching in the high dimensional spaces uses nonlinear analysis KPCA instead. The source and target pixels in the RGB color space are first mapped into their high dimensional space using

the mapping function. The eigenspaces of each source and target images are created by solving the kernel eigenvalue problems of mapped pixels. Hence, source and target pixels are separately projected into their own projection space, called *image space*. The color matching is then performed by searching the closest target pixel for each source pixel in the image spaces. To find the corresponding pixel of the closest target pixel in the RGB color space, the projection indices that are linked between two spaces can be used instead of unnecessarily performing a pixel reconstruction which is expensive in computation.

Our color matching approach does not perform the color transfer function directly in the high dimensional space since any operation in there must be in a dot-product form which is difficult or even infeasible. Moreover, a reconstructed pixel of a returned pixel from the transfer function is not guaranteed to have a corresponding pixel in the RGB color space. However, it is possible to have an approximated version of the returned pixel, also known as a pre-image problem [43] [44] [45] [46].

3.4 Color Matching in High Dimensional Space

KPCA is capable of capturing the higher-order statistics in a complex color distribution or highly nonlinear color structure which usually found in natural images [41]. The image space of an image can be created by KPCA. Color pixels are mapped non-linearly to a high dimensional space where the nonlinear color distribution hopefully becomes linear. KPCA [39], [40] is applied in color matching according to the defined properties of the image space. The algorithm of color matching by KPCA is as follow-ings.

3.4.1 Image Space Definition

Let I be an image, x a pixel in RGB color space, $x \in R^3$ and $x_i \in I, i = 1, \ldots, n$ where n is the number of pixels in I. In KPCA, RGB color pixels are mapped into the *feature space* F by a nonlinear mapping function $\Phi(x) : R^3 \rightarrow F$. After that, linear PCA is performed in the feature space using the mapped pixels $\Phi(x_i)$. To compute PCA in the feature space F, a covariance matrix (C) is created from mapped pixels in F by:

$$C = \frac{1}{n}\sum_{i=1}^{n}\Phi(x_i)\Phi(x_i)^T \tag{3.6}$$

Since $\Phi(x)$ presents in the high dimensional space which cannot be calculated directly, the kernel trick is used to perform any operation in term of dot products in the feature space by using a kernel function k in RGB color space without explicitly mapping data between two spaces. The dot product is replaced by the kernel function which is easier to compute.

As in Eq. 3.7, a kernel function of two pixel inputs x and y replaces a dot product of these two pixels that mapped into the feature space. In color matching, any positive definite kernel functions can be used. Commonly used examples of kernel functions are given in Table 3.1. The kernel function in all experiments here is a Gaussian kernel as in Eq. 3.9.

$$k(x, y) = \Phi(x) \cdot \Phi(y) \tag{3.7}$$

Table 3.1: Commonly used kernel functions

Polynomial Kernel	$k(x, y) = (x^T y + \gamma)^d$ (3.8)
Gaussian Kernel	$k(x, y) = \exp(-\|x - y\|^2/2\sigma^2)$ (3.9)
Sigmoid Kernel	$k(x, y) = \tanh(\gamma(x^T y) + \theta)$ (3.10)

KPCA is performed by solving the kernel eigenvalue problem in Eq. 3.11 where $\lambda_j, j = 1, \ldots, n$ are eigenvalues that $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n \geq 0$ while v_j is a normalized eigenvector ($\|v_j\| = 1$) corresponding to the eigenvalue λ_j and $v_j \in F \setminus \{0\}$. The eigenvector is a linear combination of transformed pixels in the feature space as in Eq. 3.12 where $\mathbf{\Phi} = \{\Phi(x_1), \ldots, \Phi(x_n)\}$ and a vector $q \in R^n$.

$$\lambda v = Cv \tag{3.11}$$

$$v = \mathbf{\Phi} q \tag{3.12}$$

With the covariance C in Eq. 3.6 and v in Eq. 3.12, the kernel eigenvalue problem may be considered as the following equivalent problem:

$$n\lambda q = Kq \tag{3.13}$$

where K is a $n \times n$ kernel matrix (a.k.a the Gram matrix) derived by a kernel function in Table 3.1 i.e., $K[x, y] = \Phi(x) \cdot \Phi(y) = k(x, y)$. The kernel matrix K for n pixels is then written as:

$$K = \begin{bmatrix} k(x_1, x_1) & \dots & k(x_1, x_n) \\ \vdots & \ddots & \vdots \\ k(x_n, x_1) & \dots & k(x_n, x_n) \end{bmatrix} \tag{3.14}$$

Note that K is not centered. The kernel matrix for centered data, K_c, can be derived from K as the following equation:

$$K_c = K - 1_k K - K 1_k + 1_k K 1_k \tag{3.15}$$

where 1_k is a constant square matrix with value $1/k$. The Eq. 3.13 is now rewritten as:

$$n\lambda q = K_c q \tag{3.16}$$

Solving the eigenvalue problem in Eq. 3.16, the eigenvectors $v_1, \dots, v_n$ are derived according to Eq. 3.12 using corresponding vectors $q^1, \dots, q^n$, respectively.

However, the eigenvectors can not be used directly due to a normalized condition in the image space, i.e.

$$(v_k \cdot v_k) = 1 \text{ for all } k = 1, \dots, n. \tag{3.17}$$

From Eqs. 3.12 and 3.16, all vectors q are normalized by the condition in 3.17 as follows:

$$\begin{aligned} 1 &= \sum_{i,j=1}^{n} q_i^k q_j^k (\Phi(x_i) \cdot \Phi(x_j)) \\ &= \sum_{i,j=1}^{n} q_i^k q_j^k K(x_i, x_j) \\ &= (q^k \cdot Kq^k) = \lambda_k (q^k \cdot q^k) \end{aligned} \tag{3.18}$$

The image space is a projection space consisting of vector bases from eigenvectors ($v_j, j = 1, \ldots, n$). In color matching, the number of eigenvectors is set in the smaller number, m, for both source and target images. The first m principal components carry more variance than the components greater than m. The higher-ordered eigenvectors having less variance tend to represent individual pixel colors which are not suitable for comparison between pixels in the image space. In the experiments, the number of eigenvector is set heuristically to 8 because of the efficiency in computation.

Once the image space defined, pixels are projected on each eigenvector ($v_1, \ldots, v_m$) of the image space. The projection of a pixel x in the image space is a m-dimensional vector $\Phi\tilde{(}x)$ which derived by:

$$\Phi\tilde{(}x) = \left(\Phi(x)^T v_1, \ldots, \Phi(x)^T v_m\right)^T \tag{3.19}$$

For each projection component of the pixel x on eigenvector v can be derived by:

$$\begin{aligned} \Phi(x)^T v &= \Phi(x)^T \mathbf{\Phi} q \\ &= q_1 \Phi(x)^T \Phi(x_1) + q_2 \Phi(x)^T \Phi(x_2) +, \ldots, + q_n \Phi(x)^T \Phi(x_n) \\ &= \sum_{i}^{n} q_i K(x_i, x) \end{aligned} \tag{3.20}$$

From Eq. 3.20, the Eq. 3.19 is now rewritten in the kernel form as:

$$\Phi\tilde{(}x) = \left(\sum_{i}^{n} q_i^1 K(x_i, x), \ldots, \sum_{i}^{n} q_i^m K(x_i, x) \right)^T \tag{3.21}$$

It is useful to keep the projection vectors for each pixel in the matrix form, P, since it will be used to calculate a distance between two pixels in the image space as

described in the next section. P is a $m \times n$ projection matrix for n pixels and each pixel is projected on m eigenvectors.

$$P = \begin{bmatrix} \Phi(x_1)^T v_1 & \dots & \Phi(x_n)^T v_1 \\ \vdots & \ddots & \vdots \\ \Phi(x_1)^T v_m & \dots & \Phi(x_n)^T v_m \end{bmatrix} \tag{3.22}$$

According to Eq. 3.19, the matrix P in Eq. 3.22 can also be rewritten in the kernel form as in Eq. 3.23.

$$P = \begin{bmatrix} \sum_i^n q_i^1 K(x_i, x_1) & \dots & \sum_i^n q_i^1 K(x_i, x_n) \\ \vdots & \ddots & \vdots \\ \sum_i^n q_i^m K(x_i, x_1) & \dots & \sum_i^n q_i^m K(x_i, x_n) \end{bmatrix} \tag{3.23}$$

3.4.2 Pixel Search and Match

The source and target pixels in their own image spaces are not shared in the same coordinate system, but can be compared because of sharing the same properties of the image spaces. Such properties are listed as followings.

1. Pixels in each image space are centered around their mean so that the mean becomes a new origin. The new origins of source and target pixels can be virtually registered as the same reference point between two image spaces.

2. The vector bases of each image space must be orthogonal and ordered according to non-increasing color information captured in each basis.

3. The number of vector bases between two image spaces must be the same.

4. The pixels in the image space are normalized so that the color values from different images are scaled and able to be compared equally.

Once source and target pixels are projected into their image spaces, each projected source pixel ($\Phi\tilde{(}s)$) searches for the closest projected target pixel ($\Phi\tilde{(}t')$) by using a distance function, $d(\ ,\ )$ as in Eq. 3.24.

$$\Phi\tilde{(}t') = \min_i \ d\left(\Phi\tilde{(}s), \Phi\tilde{(}t_i)\right), i = 1, \dots, t_{np} \tag{3.24}$$

where t_{np} is the number of pixels in the target image. The squared Euclidean distance between pixels x and y in RGB color space can be used as the distance function in the feature space and can be derived in the kernel matrix form by:

$$\begin{aligned} d(\Phi(x),\Phi(y)) &= \|\Phi(x)-\Phi(y)\|^2 \\ &= \Phi(x)^T\Phi(x) - 2\Phi(x)^T\Phi(y) + \Phi(y)^T\Phi(y) \\ &= K(x,x) - 2K(x,y) + K(y,y). \end{aligned}$$

However, the search in color matching takes place in the image space. The squared Euclidean distance in the image space needs to be rewritten as in 3.25. The distance between a source pixel i, $\Phi(\tilde{s}_i)$ and a target pixel j, $\Phi(\tilde{t}_j)$ in the image space can be derived in the projection matrix form as in Eq. 3.26.

$$\begin{aligned} d(\Phi(\tilde{x}),\Phi(\tilde{y})) &= \|\Phi(\tilde{x})-\Phi(\tilde{y})\|^2 && (3.25) \\ &= \Phi(\tilde{x})^T\Phi(\tilde{x}) - 2\Phi(\tilde{x})^T\Phi(\tilde{y}) + \Phi(\tilde{y})^T\Phi(\tilde{y}) \\ &= P(x)^TP(x) - 2P(x)^TP(y) + P(y)^TP(y) && (3.26) \end{aligned}$$

The search operation does not change the projected target pixels; therefore, reconstruction of the projected pixels should return the corresponding pixels in RGB color space. However, it is unnecessary and expensive in computation to reconstruct these pixels back to RGB. Instead, a pixel in RGB and its projected pixel in the image space can refer to each other by an index of the projection matrix.

As in Eq. 3.27, a pixel x in RGB color space corresponds to its projected pixel in the image space by using an index i. In color matching, each source pixel x_s can refer to its corresponding pixel in the image space via the index i of the projection matrix. Similarly, the index j is used in the same way for each target pixel. Therefore, each source pixel can find its corresponding matched target pixel in RGB via the indices i and j, and the searching process as demonstrated in Eq. 3.28.

$$x \overset{(i)}{\Rightarrow} \Phi(\tilde{x}) \tag{3.27}$$

$$x_s \overset{(i)}{\Rightarrow} \Phi(\tilde{x}_s) \Rightarrow \Phi(\tilde{x}_t) \overset{(j)}{\Rightarrow} x_t \tag{3.28}$$

To have a clear overview of color matching, Fig. 3.1 shows a diagram of color matching between source and target images. The searching part is done in the image space while the matching part is done in the RGB color space.

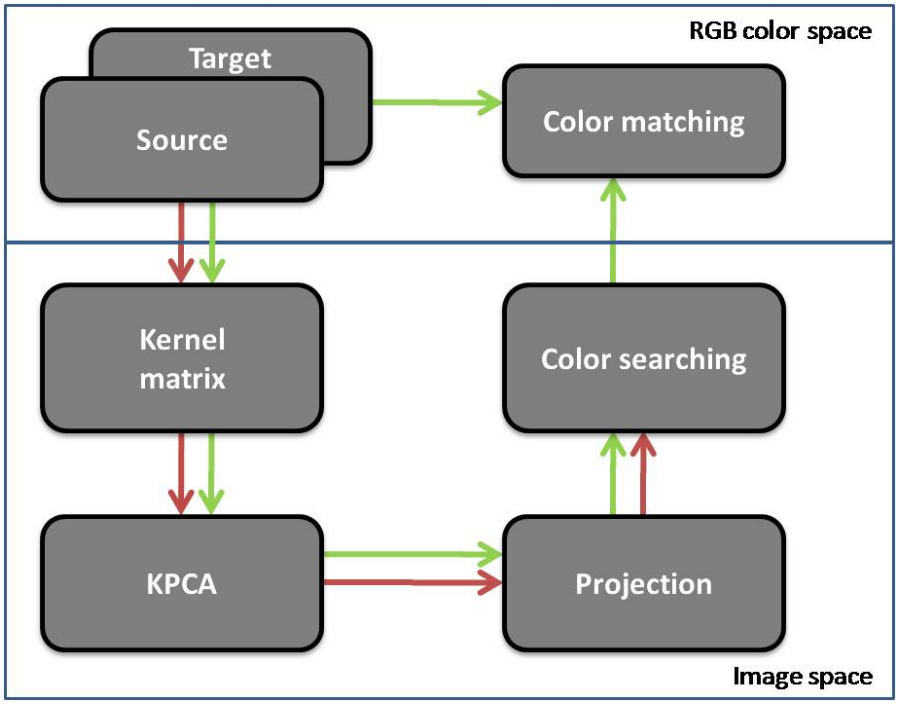


Figure 3.1: Diagram of color matching using KPCA

Below is the algorithm of color transfer via color matching according to the concept of color matching in the image space. The algorithm here is kept short and provides enough details to follow the concept so far. There are some practical issues which will be discussed later.

Color Matching Algorithm

Input: a source image I_s, a target image I_t
Output: a result image I'_s

Step 1. Read in I_s and I_t

Step 2. Create the kernel matrices K_s and K_t for respective I_s and I_t for extracting eigenvalues and eigenvectors.
For each of I_s and I_t:

(a) at first, the kernel matrix K is computed from the kernel function in Table 3.1, say, $K(x, y) = \exp(-\|x - y\|^2/2\sigma^2)$ from Eq. 3.9.

(b) The kernel matrix for the centered data K_c is derived by Eq. 3.15. Hence, calculate eigenvalues and q in Eq. 3.13;

(c) normalize q according to Eq. 3.18:

$$\bar{q^k} = \frac{q^k}{\sqrt{\lambda_k}}$$

(d) use normalized $\bar{q^1}, \ldots, \bar{q^8}$ in Eq. 3.12 for the first 8 eigenvectors to define the image space;

(e) project each pixel to the image space and save in a projection matrix P as in Eq. 3.23.

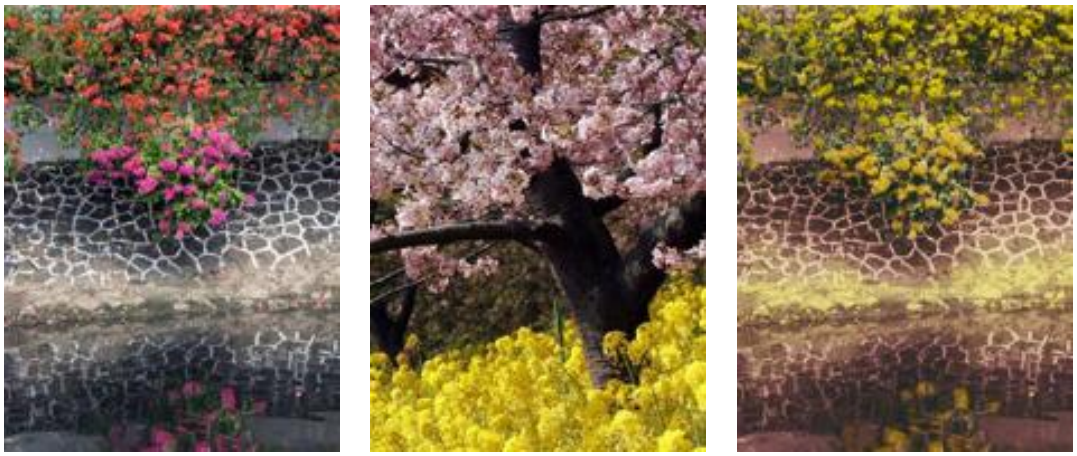
Step 3. In the image space, for each source pixel i:

(a) search for the closest target pixel as in Eq. 3.24. The distance between a source pixel $\Phi(\tilde{s_i})$ and a target pixel $\Phi(\tilde{t_j})$ in the image space can be derived in the projection matrix form as:

$$\begin{aligned} d(\Phi(\tilde{s_i}), \Phi(\tilde{t_j})) &= \|\Phi(\tilde{s_i}) - \Phi(\tilde{t_j})\|^2 \\ &= \Phi(\tilde{s_i})^T \Phi(\tilde{s_i}) - 2\Phi(\tilde{s_i})^T \Phi(\tilde{t_j}) + \Phi(\tilde{t_j})^T \Phi(\tilde{t_j}) \\ &= P_s(i)^T P_s(i) - 2P_s(i)^T P_t(j) + P_t(j)^T P_t(j) \end{aligned}$$

where P_s and P_t are the source and target projection matrices, respectively. Note that both source and target pixels have the same number of eigenvectors in the projection spaces (the image spaces).

(b) Find the corresponding pixel in RGB color space of the closest target pixel in the image space. The source and target projection matrices are also served as cross references according to 3.27, 3.28.



(a) (b) (c)

Figure 3.2: Color transfer via color matching: (a) source image, (b) target image, (c) color matching by KPCA

According to the above algorithm, Fig. 3.2 shows color transfer between the source image (a) and the target image (b) by using the color matching technique. The result shown in (c) captures colors only appearing in the target image. Notice that different colors of flowers in the source image are all transferred to yellow color without color artifacts. Section 3.6 will show more results which are also compared with other global color transfer techniques.

3.4.3 Practical Implementation

There are many issues related to practical implementation in color matching.

Speed up the kernel creation: Color matching using KPCA is a long process with a lot of memory usage. The kernel gets larger and takes longer time to compute when the image gets larger size. The next section will discuss a method to speed up the kernel creation.

Search problem: The search process also takes longer time while the image size gets bigger. There are existing techniques that can reduce searching time such as using triangular inequality [47].

As the same RGB color pixels will have the same projected pixel as well as the same matched pixel, only the representative pixels are used in calculation while the duplicated pixels can simply refer to their corresponding representative pixels.

Color dynamic range: Low color dynamic range may occur when considering only the closest target pixel. To reduce the problem, k-nearest neighbors (KNN) is considered to increase the color dynamic range. To find k-nearest pixels, it simply extends finding the closest pixel to get more pixels up to k since the pixel distances are already sorted. Hence, all k-nearest pixels are averaged using their reciprocal distance weights.

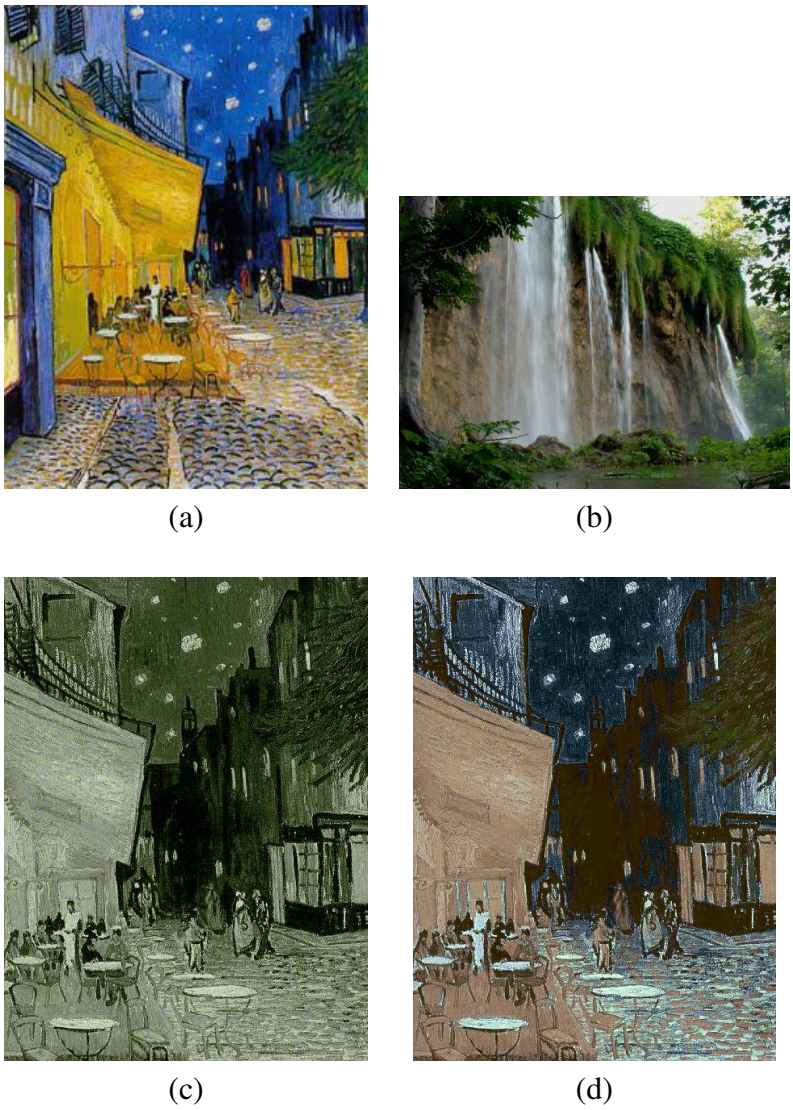
3.4.4 Speed up the Kernel Construction

Color matching by KPCA has limitation of high computation and high memory usage. Particularly, when dealing with high resolution images, calculating the eigenvectors of such large matrices is expensive and sometimes prohibitive as it requires $O(N^3)$ operations and $O(N^2)$ memory storage, where N is the number of pixels.

All pixels should be used in the kernel creation since every pixel can contribute to the color characteristic of the image it belongs to. Reducing the number of pixels involved in kernel creation can speed up the process. Sampling pixels or removing duplicated pixels in this step will cause a lower quality of the result.

In case of the high resolution images, it is inevitable to trade off the lower computation and less memory usage with some qualities in the result. Greedy KPCA [48], [49] is a technique to speed up the kernel creation. The technique works with a smaller subset of data which is used to approximate a whole data set, and thus requires less memory and computation. The algorithm iteratively searches for the best smaller set of independent data in the feature space so that the rest of data can be derived by a linear combination of this smaller set.

To create a kernel of an image with size of 600×450 requires an $n \times n$ kernel matrix where n is the number of pixels, i.e., $n = 600 \times 450 = 270,000$. It is beyond a capacity of memory in a current hardware system. Fig. 3.3 compares two techniques to speed up the kernel creation in color matching. Because of the high number of pixels, both source and target images are sampled. Fig. 3.3 (c) shows the result when using samples directly in normal KPCA while (d) shows the result from the greedy KPCA. Using samples directly in KPCA reduces some crucial data contribution to the kernel. In other words, the kernel insufficiently capture the color characteristic by simply using



(a) (b)

(c) (d)

Figure 3.3: Comparison two methods that speed up color matching: (a) source, (b) target, (c) color matching by KPCA with sampled data, (d) color matching by greedyKPCA

sampled pixels directly. On the other hand, the greedy KPCA algorithm uses a reduced set of pixels that should suffice to represent a whole image. Hence, the greedy KPCA algorithm can speed up the computation time, reduce memory usage, and return a decent result.

3.5 Result Evaluation

The aim of color transfer is to modify the color appearance of a source image to look similar to that of a target image. Human usually has a personal impression to what is perceived, or has a subjective evaluation for looks and feels of an image

appearance. Therefore, the evaluation of a color transfer technique often is subjective. However, to evaluate the result of the color transfer process objectively as in Xiang et al. work [26], colorfulness similarity can be used to measure the color appearance between the result and target images. Hasler and Susstrunk [50] proposed the colorfulness metric in the RGB color space. The metric is an effective method for evaluating colorfulness between the color-transferred result and target images. The colorfulness metric C is defined as

$$C = \sqrt{\sigma_{rg}^2 + \sigma_{yb}^2} + 0.3\sqrt{\mu_{rg}^2 + \mu_{yb}^2} \tag{3.29}$$

where μ and σ are the mean and standard deviations of the pixels in the opponent spaces: Red-Green and Yellow-Blue. The subscripts rg and yb refer to the opponent spaces Red-Green and Yellow-Blue, respectively. The pixel values in these opponent spaces can be derived by:

$$\begin{aligned} rg &= R - G \\ yb &= 0.5(R + G) - B \end{aligned}$$

With the colorfulness of the result image, C_r, and the colorfulness of the target image, C_t, the colorfulness similarity, CS, between two images can be derived by

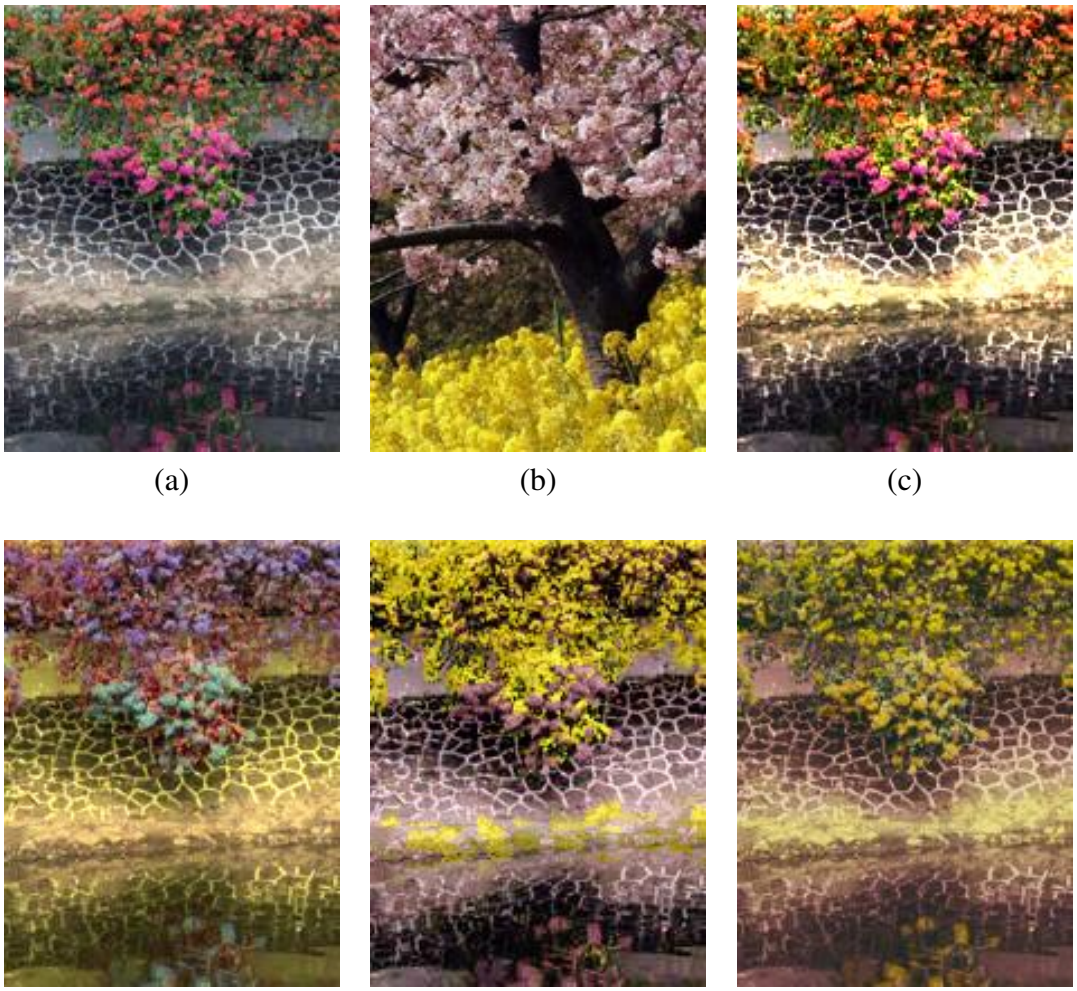
$$CS(r, t) = |C_r - C_t| \tag{3.30}$$

A low value of $CS(r, t)$ indicates close similarity in colorfulness between the result and target images.

The evaluation comparison among color transfer techniques will be discussed in Section 3.6.

3.6 Results and Discussion

In this section, the proposed algorithm was compared with other color transfer techniques (Reinhard et al. [3], PCA-based [17], and Nd-PDF [2]) based on assumption that all techniques aimed for global color transfer; hence, the experiments were set without user interaction such as swatches.



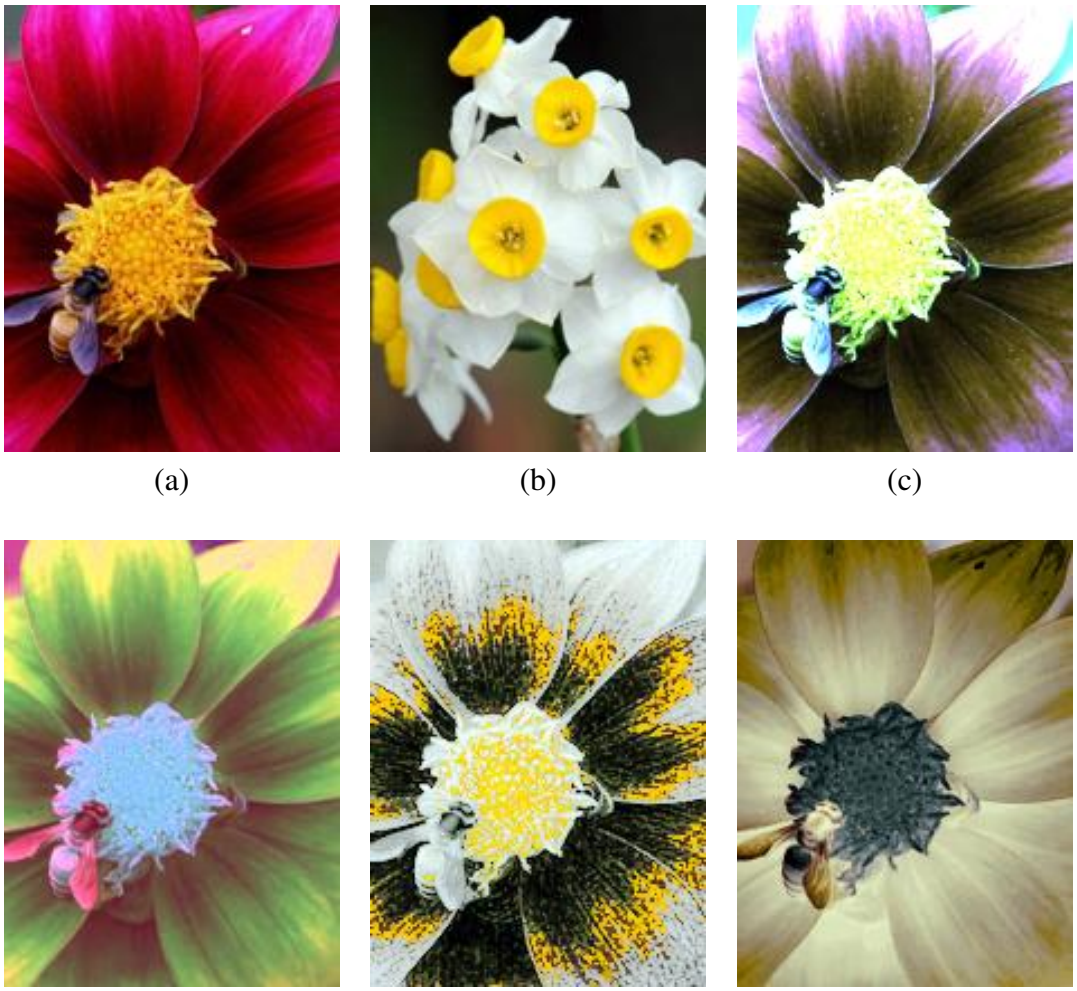
(a) (b) (c)

(d) (e) (f)

Figure 3.4: Color transfer comparison: (a) source, (b) target, (c) Reinhard, (d) PCA-based, (e) Nd-PDF, (f) color matching by KPCA

Because of linear transform used in Reinhard and PCA-based techniques, the result colors may be not close to the target colors and some colors from the source image are left in the result, as shown in (c) and (d) of Fig. 3.4, Fig. 3.5 and Fig. 3.6. These effects can be noticeable, particularly when either the source image or the target image or both have non-Gaussian color distributions. The effects can be emphasized if the dynamic range or scaling factors between the source and the target images are particularly different.

The results of Nd-PDF had the color appearance similar to that of the target image. However, there is an inconsistency problem in colors apparently shown in some areas. Especially, in Fig. 3.5, the pink areas of the source image are separately



(a) (b) (c)

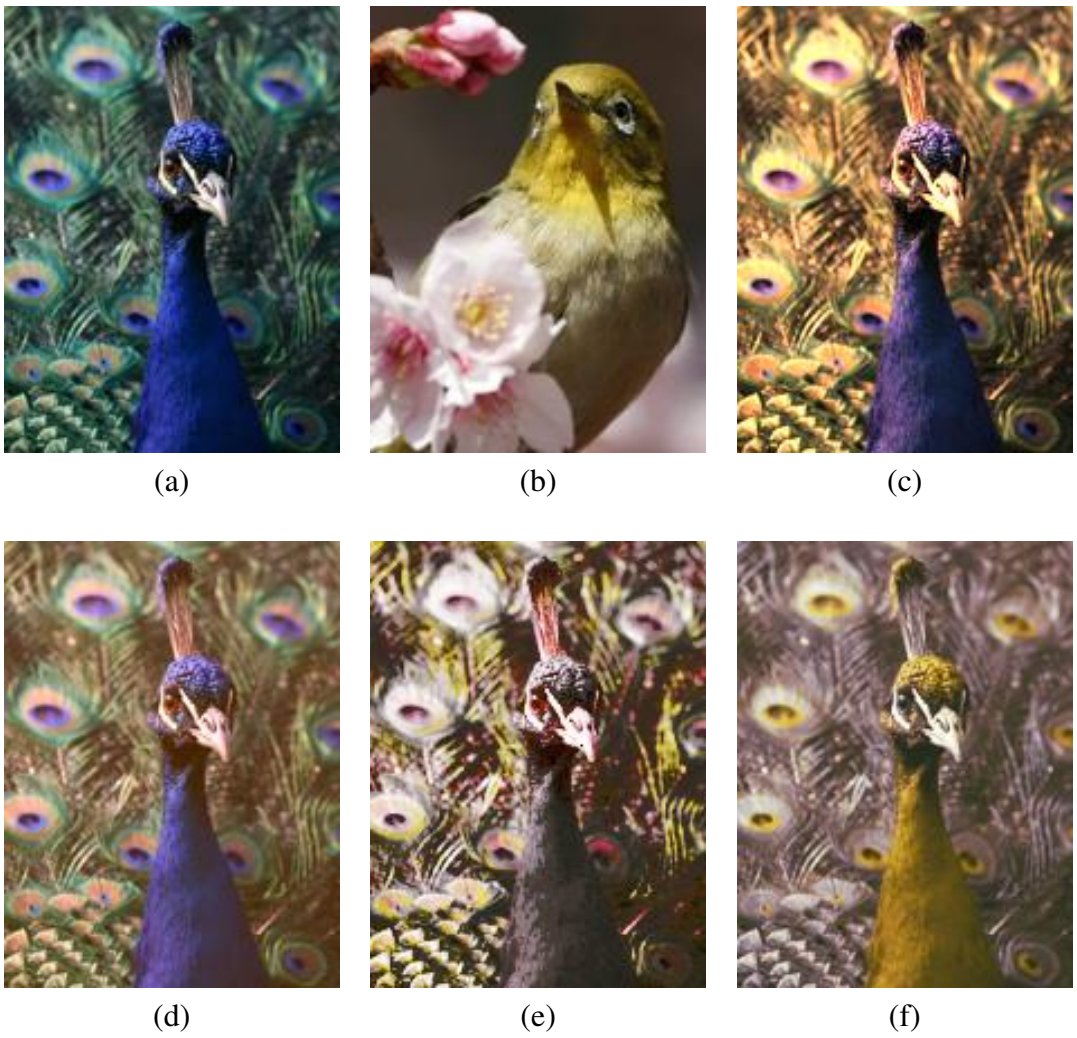
(d) (e) (f)

Figure 3.5: Color transfer comparison: (a) source, (b) target, (c) Reinhard, (d) PCA-based, (e) Nd-PDF, (f) color matching by KPCA

transferred to white, black and yellow areas in the result as shown in Fig. 3.5.(e). The inconsistency problem usually occurs when an image has a large difference in color distributions among different areas. The majority color can then be separated and transferred into different colors.

Color matching by KPCA had no inconsistency problem and gave the result colors from the target image without color artifacts. However, the color dynamic range of the result may be lower that that of the target image because of the search mechanism. Moreover, some colors shown in the target image may not be seen in the result. If the major colors of the target image get lost in the result as in Fig. 3.5.(f), the overall color appearance of the result may not be convincing as compared with Fig. 3.4.(f) and

Fig. 3.6.(f).



(a) (b) (c)

(d) (e) (f)

Figure 3.6: Color transfer comparison: (a) source, (b) target, (c) Reinhard, (d) PCA-based, (e) Nd-PDF, (f) color matching by KPCA

The results shown in Fig. 3.4, Fig. 3.5, and Fig. 3.6 from each color transfer technique, i.e., Reinhard et al technique, PCA-based technique, Nd-PDF technique, and our technique, color matching by KPCA are evaluated here using the evaluation method of colorfulness similarity explained in Section 3.5. The result evaluation are shown in Table 3.2.

As mentioned before, a low value of $CS(r,t)$ indicates close similarity in colorfulness between the result and target images. From the Table 3.2, the Nd-PDF method gives the best evaluation in Fig. 3.4 while the PCA-based method gives the best evaluation in Fig. 3.5 and Fig. 3.6.

Table 3.2: Evaluation comparison among color transfer techniques

Colorfulness Similarity, $CS(r,t)$	**Reinhard**	**PCA-based**	**Nd-PDF**	**CM by KPCA**
Figure 3.4	16.3	26	9.2	26.4
Figure 3.5	15.1	1	8.8	5.2
Figure 3.6	30.6	9.5	15.4	41.5

The evaluation in some cases may conflict with the evaluation by eyes. As shown in Fig. 3.4, the techniques of Nd-PDF and our color matching gave the results with similar color appearances but their evaluation values are quite different. The evaluation of colorfulness similarity does not consider the spatial information of an image. Figure 3.6 show this conflict obviously when the evaluation of the PCA-based method is the best whereas the evaluation of our color matching is the worst.

Compared with the subjective evaluation by eyes, the evaluation of colorfulness similarity is still not sufficient to give the result evaluation of a color transfer method in numbers. The evaluation of color transfer methods is an on-going research. Our future work will include the technique enhancing the quality of the evaluation method.

3.7 Conclusion

This chapter has proposed the global color transfer technique. The technique bases on searching colors in the high dimensional space and matching to the RGB color space. Our method aims to be automatic and to handle complex color distributions which usually found in natural images.

The experimental results show some problems in the previous techniques when handling with the complex color distributions; for example, color artifacts according to the interpolation in a linear color transform, color inconsistency according to the large differences of color distributions in nonlinear histogram transform. Our method has not shown such problems.

A linear transformation is simple and fast, but limited to simple Gaussian color distributions. Our method can cope with complex color distributions but its process is also complex and takes much time and memory usage. Since our method uses the

searching technique for the closest pairs of source and target pixels, the derived results shows a low quality in the color dynamic range.

Even though the color matching technique can give a result whose color appearance is close to the target image, it gives no control to a user. Chapter 4 will focus on an interactive color transfer technique which offers some controls to a user to manipulate smaller regions where colors will be transferred.

CHAPTER IV
INTERACTIVE COLOR TRANSFER BY REGION EXPLORATION

Color transfer between images with complex color distributions can be solved by a nonlinear method as demonstrated in Chapter 3. Another method is local color transfer which considers color transfer in smaller scale, that is, color transfer between local regions. The local method simplify the complex color distribution of an image into simpler color distributions of several regions. The local color transfer can be either automatic as in [24], [26], or interactive as in [51] [30], [52], [6], [36], [53], [37]. However, the interactive approach can offer more benefits besides handling with complex color distributions as described in following scenarios:

Scenario 1. One source versus one target

The purpose for this scenario is similar to global color transfer, i.e., to transfer a color impression of a target image to a source image as a whole. When considering color transfer in a smaller scale with local regions, a user needs tools to manipulate the color transfer process. This leads to an interactive approach that lets a user define regions locally, control which pairs of source and target regions will be performed color transfer, or which source region will be left unchanged.

Scenario 2. One source versus many targets

At times, a user may require looks and feels from different target images to create a mixed feeling into a source image. The opportunities to select particular areas from different target images increase a new dimension in color transfer. It does not intend to simulate a feeling of one specific target image; instead, bring up each outstanding feel of particular areas from several target images and put together into the source image.

This scenario arises with the increasing number of professional images available from the internet, e.g. www.flickr.com, www.msnbc.msn.com, and www.sxc.hu to name a few. Color transfer can take advantages of a variety of professional images as reference images.

Scenario 3. Specific source regions versus a collection of specific target regions

It is not unusual that only specific regions of a source image need to change their color content. A particular type of images can be collected by defining specific regions of each image and tagging them by a type name; for example, "SKY" is named for images with sky regions. Color transfer between specific source regions against a tag name will transfer color between pre-defined regions of each target image and the source regions in question. This scenario can return a number of results according to the image collection with the tag name.

4.1 Introduction

In general, interactive color transfer consists of four major parts; namely, region definition, region extraction, region mapping, and region color transfer. Region definition lets a user define local regions of an image. Regions can have soft or hard boundaries according to segmentation techniques. Users can provide guidelines to the segmentation process using interactive tools such as swatches, scribbles, strokes, or brushes. Another method is defining regions by modeling such as Gaussian mixture models. This method usually is automatic but can receive the pre-defined number of regions from a user.

Region extraction decomposes an image into several regions by image segmentation. There is a number of image segmentation techniques usually categorized to soft or hard segmentation. The hard segmentation often introduces color artifacts along the boundary when used later in some applications such as color transfer [30], Drag-and-Drop pasting [54], or image cloning [55]. A wildly used solution for the boundary problem is the gradient smoothing method [35]. On the other hand, the soft segmentation returns weights for pixels to represent their membership among regions. Normally,

pixels in the centered regions have high weights for the region they belong to as major memberships where as pixels around the region boundaries are usually not clear which regions they belong to. However, the weights along the boundary are used to decompose regions with smoother boundaries which prohibit the boundary problem once the regions are recomposed again.

Once the regions are extracted, there is an issue about which source regions should be mapped which target regions. With a number of regions of both source and target images, there is a large number of possible mapped pairs of source and target regions. However, the interactive approach lets a user manually map pairs of source and target regions via the interactive tools.

Linear color transfer by Reinhard et al. [3] is fast and good enough when handling with regions that can be modeled by Gaussian models. Color transfer between regions depends on the segmentation types. Regions extracted by hard segmentation can be applied by the linear color transfer directly but need a post-processing to smooth the boundary areas. Soft segmentation uses pixel weights to decompose regions. The weights are also used in the modified version of the linear color transfer as in [24].

In the previous techniques of interactive color transfer, manually manipulating regions gives a user more freedom and options to control region definition in interactive color transfer. However, this approach is a error prone for a less skilled user who might put strokes in the wrong place and then generate disqualified regions for color transfer.

A user does not only define local regions in the user interface, but also define region mappings between source and target regions. Color transfer is then performed according to pre-defined region mapped pairs. The recent techniques [52], [37] show tightly couple between region definition and region mapping in the user interface, i.e., the strokes determined by a user are tied to region extraction and the stroke colors used for region mapping are coupled with the color transfer process in a later step. Such technique are not responsive to changes because of tightly coupled processes. If a user is not satisfied with the result and needs some changes, such techniques will take more time to recompute some parts or overall process.

In this chapter, we propose interactive color transfer by region exploration.

This is a semi-automatic method that separates region definition from region color transfer. Using exploration approach, the program will generate intermediate results automatically and then lets a user interact with the user interface of the program in order to navigate, evaluate, and select the results. Such method can reduces the error introduced by a user and also reflects readiness to change and responsiveness in the user interface.

4.2 Related Work

Reinhard et al. [3] introduced global color transfer between images. Their automatic method define an images by a Gaussian model in the independent color space, $\ell\alpha\beta$. Linear color transformation is then performed on each channel of $\ell\alpha\beta$. The algorithm is simple but effective with a Gaussian model. However, in case of a complex color distribution which can not be modeled by a simple Gaussian model, the result usually shows color artifacts. Using swatches is one method to solve this problem as described in their paper and also in the others.

Tai et al. [24] proposed local color transfer that uses Gaussian Mixture models (GMM) to model an image. Using GMM, color distributions of local regions are simplified and then modeled by simple Gaussian models. Once having region models, source and target regions can be extracted. Regions are mapped by rules, and then performed color transfer. Xiang et al. [26] continued the Tai's idea with more target images. They also introduced additional rules for region mapping and some evaluation criteria.

Interactive color transfer techniques give a user more control to manipulate color transfer on local regions instead of the whole image. These techniques add a region definition and region extraction processes in a variety of methods such as matting [53], rough and hard segmentation [30], cellular automata [6], [37], graph-cut [52]. Similar to local color transfer, interactive approach already includes local consideration in the processes of region definition and extraction. That means the interactive color transfer can also solve the complex color distribution problem as the same as local color transfer does.

Swatches is one of interactive tools that used in many work such as [3], [5]. However, many techniques recently uses a kind of strokes as an interactive tool. The stroke tools give a user convenient to locate the desired regions much easier than

swatches. Regions are then extracted according to initial stroke locations. Moreover, the stroke's colors provide a way to map regions of source and target in order to perform color transfer too.

Since each region is usually modeled by a simple Gaussian model, color transfer between regions can simply use a linear and statistical transformation as in [3]. However, there are obvious color artifacts around the region boundaries as expected. The solutions depend on the types of region extraction or region segmentation. The soft segmentation techniques used in [53], [6], [37], [52] provide contribution weights for all pixels which can be used in a modified version of the linear color transfer. By using weights, pixels along the boundaries are blended smoothly without color artifacts. On the other hand, the hard segmentation has no weights and need additional method to solve the boundary problem such as the gradient smoothing method used in [30].

Another interactive approach introduced by Shapira et al. [56] gave another point of view of interactive image editing. As mentioned before, editing an image by parameter tuning is a difficult task and usually need a lot of skills to make a good editing. The exploration approach hides the tuning part from a user and lets a user explore some results with various parameter settings. A user gives a response back as a guideline for further exploring. The exploration is a high dimensions in the search space. It gives a user a freedom to search his or her desired results without exactly tuning parameters. However, this technique is not suitable to search for a specific target as in the example-based techniques.

4.3 Region Exploration Approach

The exploration approach is a kind of interactive approaches. Instead of giving a user full controls, the exploration method is a semi-automatic and iterative method that generates results for a user as choices. Hence, a user can navigate, evaluate, and select a desired result. The algorithm can respond to the user selection by generating a new set of results due to the current selection. A user can select the result repeatedly until satisfied with the final result.

The region exploration approach can be illustrated by analogy with the Google search. In the Google search, a user can type a keyword; for example "sky" in order to

search for sky images. Google will return a number of sky images for the user to explore as shown in Fig. 4.1. The user can then navigate, evaluate, and select the returned results.

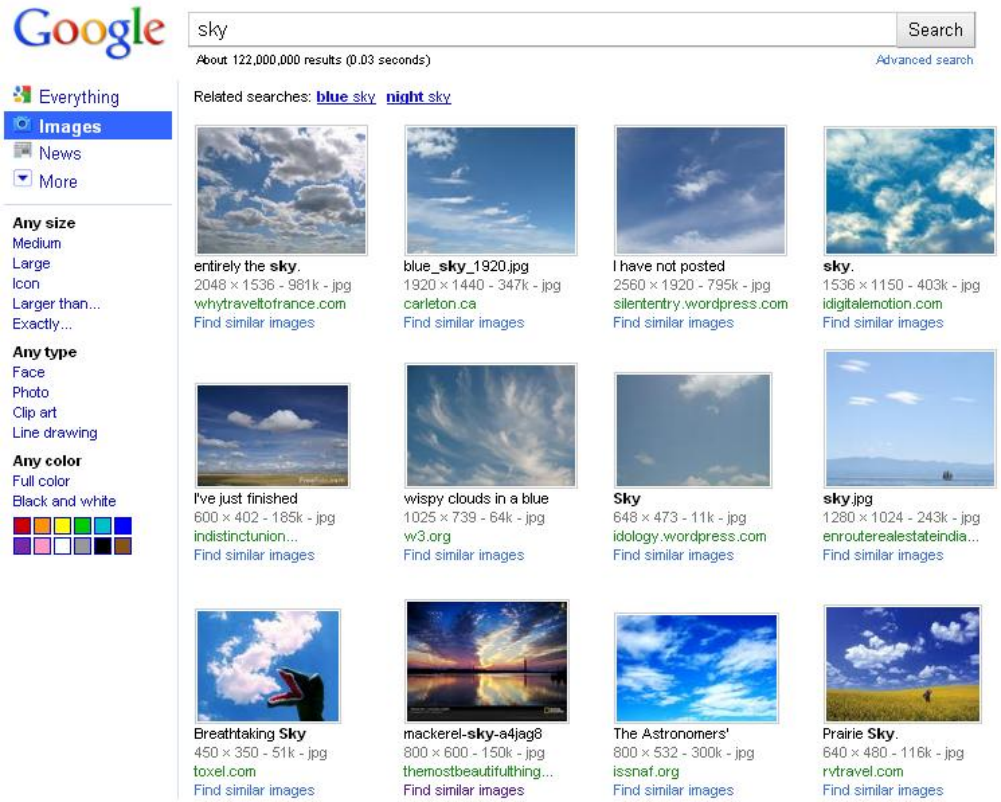


Figure 4.1: Exploration approach in the Google search

Without a user selection, the user can navigate back and forth to another page in order to explore more results. With a user selection, Google will return another set of sky images according to a new criteria from the user selection. Figure 4.2 shown a new set of sky images from the Google search when a user clicked on the link “Find similar images” under the second image from the last row shown in Fig. 4.1.

4.4 Interactive Color Transfer by Region Exploration

The proposed technique of interactive color transfer has two major parts: region definition and region exploration. The region definition is a semi-automatic method that generates regions automatically and then lets a user refine the regions if desired. Region exploration is an iterative process that offers a user a freedom to navigate, evaluate,

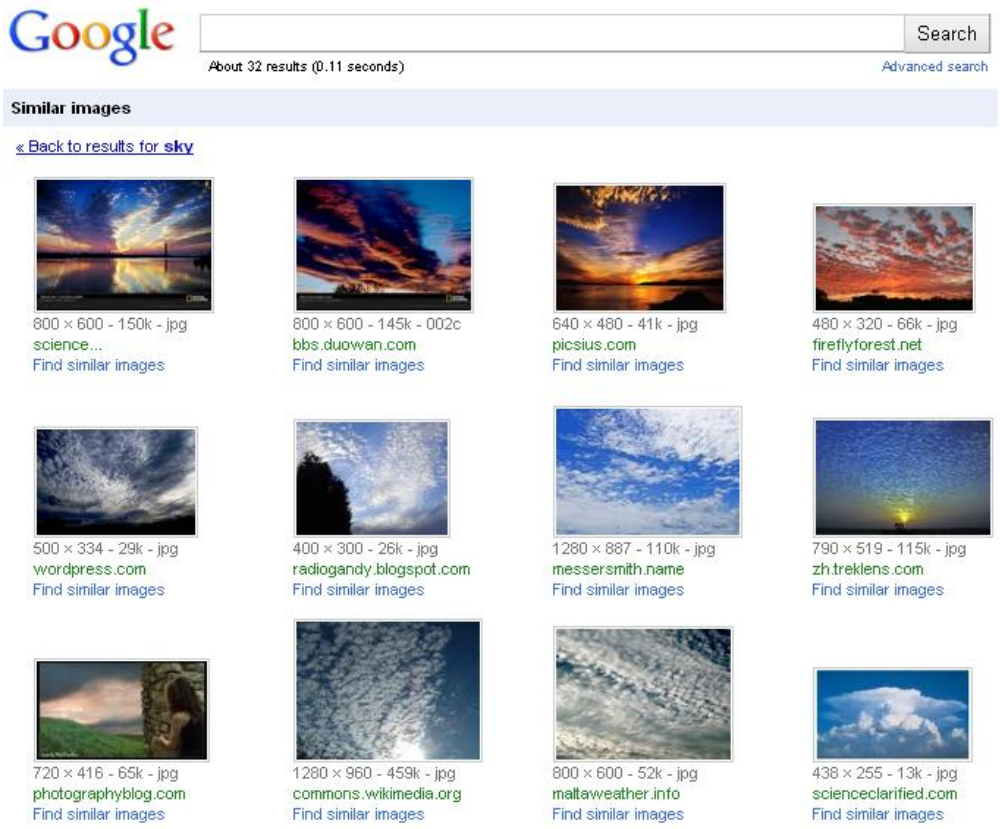


Figure 4.2: The Google search response to the user selection

and select the results generated automatically from pre-defined region mapped pairs.

The framework of the proposed method is illustrated in Fig. 4.3. The framework has 2 parts. First, region definition begins with modeling an image automatically by Gaussian mixture models. Each region is modeled by each corresponding Gaussian model derived by GMM. The regions can be extracted according to probabilities of the Gaussian models. The regions and their models are then saved for the later use in region color transfer.

The second part is region exploration which needs no pre-defined region mapping from a user but provides all possibilities of region mapped pairs between source and target regions derived by region definition. Region exploration is an interactive and iterative process. Each iteration, color transfer between one source region and each target regions is performed and generate intermediate results that can be evaluated by a user. The user can select a desired intermediate result or keep exploring to the next iteration. The current source region will be updated according to the user selection. The

region exploration process is so responsive that the user can freely navigate back and forth in the exploration loop, and can make any change as desired.

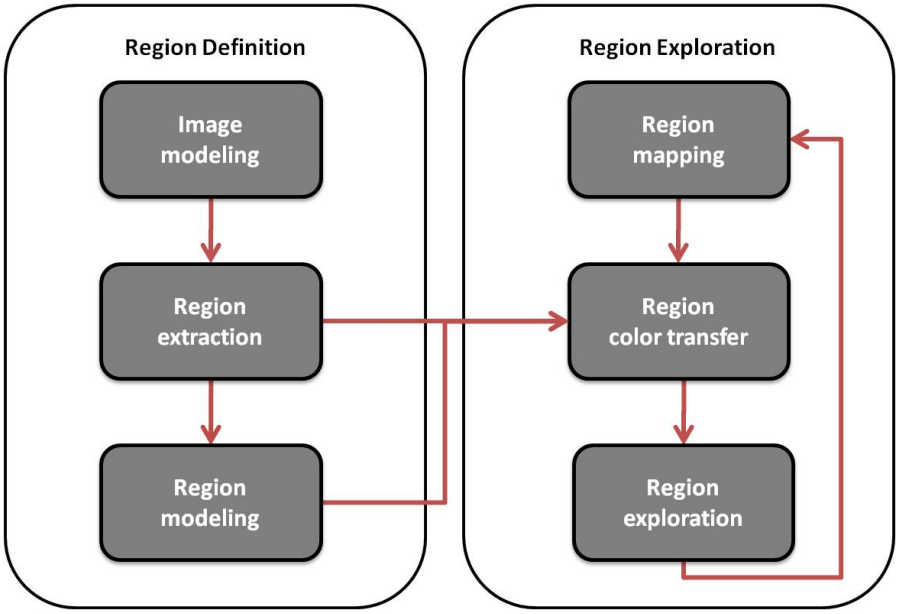


Figure 4.3: Interactive color transfer by region exploration diagram

Next sections will explain more details of each part in the framework of interactive color transfer by region exploration.

4.4.1 Region Definition

Interactive color transfer approach considers partial color transfer between smaller regions from a source and a target images. A user can interactively select which region should change its colors, or just keep unchanged. Regions instead of a whole image are considered and needs to be defined first. Breaking an image down into several regions does not only serve the interactive approach, but also simplify a complex color model into several simple color models. There are a number of techniques to define region including soft and hard image segmentation.

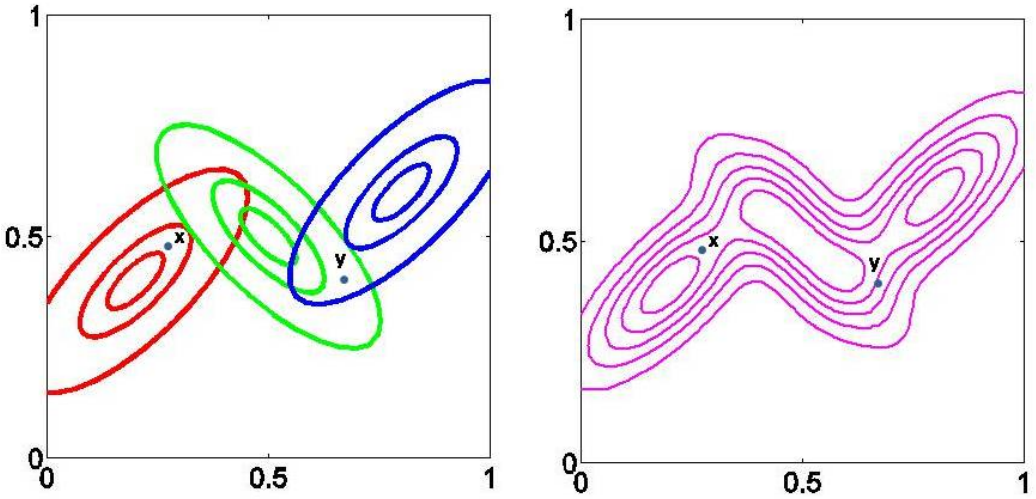
Existing techniques such as [30], [52], [6], [37] use the stroke tools to define regions by letting a user put strokes across desired regions. This approach may return regions that serve the user's desire but stil have complex color distributions which are not suitable with linear color transfer. Our method defines regions from Gaussian mixture

models. The extracted regions from GMM have soft boundaries and the color transfer process can use them directly and efficiently in the statistical transformation as in [3].

Image Modeling by GMM

GMM is a parametric statistical model which assume that the data originates from a weighted sum of several Gaussian sources. Figure 4.4 shows a simple concept behind GMM. Any data can be generated from 3 individual Gaussian models as shown explicitly in red, green, and blue ellipses, see in (a), while (b) shows a nonlinear model derived by mixing 3 Gaussian models from (a).

As shown in the figure, x and y are generated or mixed from these 3 Gaussian models in different proportions. For example, x is generated by 3 models with probabilities: $P_{red} = 0.79$, $P_{green} = 0.2$, and $P_{blue} = 0.01$ whereas probabilities of y are $P_{red} = 0.01$, $P_{green} = 0.54$, and $P_{blue} = 0.45$.



(a) (b)

Figure 4.4: A simple concept of 3 Gaussian mixture models

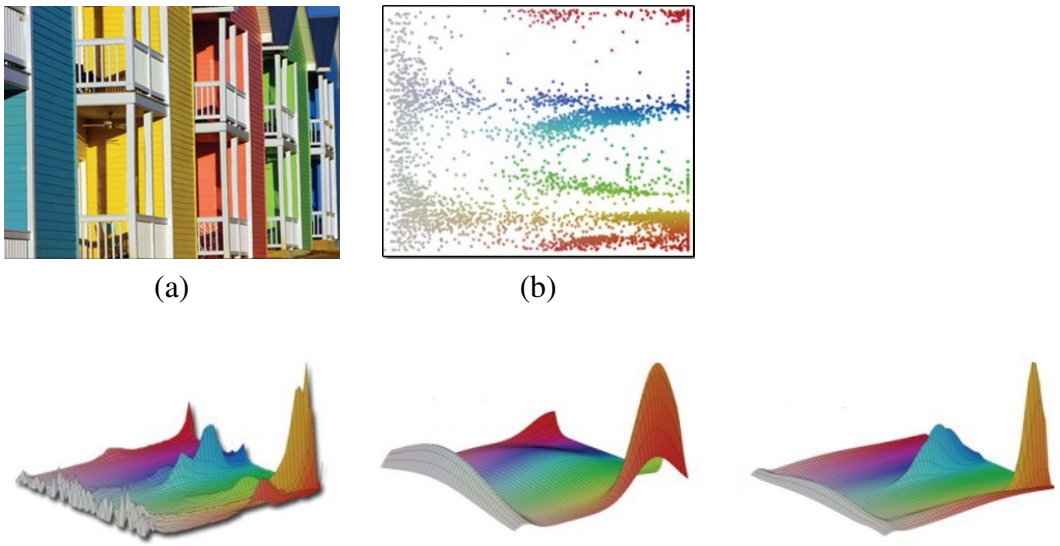
Similarly, an image pixels can be derived from Gaussian mixture models. Each pixel can belong to every model but in a different proportions. Suppose that x and y are pixels in an image modeled by GMM as shown in Fig. 4.4. For each pixel, three probabilistic values must be summed up to one, i.e. $P_{red} + P_{green} + P_{blue} = 1$. The higher probability of the pixel x, namely P_{red}, indicates that x is most likely influenced

by the red model and probably locates inward the region generated by the red model. In addition, x is also partially influenced from the green model but rarely from the blue model.

In a color image, the joint color distribution of a whole image is modeled by a mixture of M Gaussian models which can be calculated as:

$$p(x|\Theta) = \sum_{i=1}^{M} a_i p(x|\lambda_i) \tag{4.1}$$

where x is a pixel ($x \in R^3$) and the parameters $\Theta = a_1, \ldots, a_M, \lambda_1, \ldots, \lambda_M$. a_i denotes the weight of each Gaussian, λ_i its respective Gaussian parameters. The prior weight a_i must satisfy $a_i \geq 0, i = 1, \ldots, M$ and $\sum_{i=1}^{M} a_i = 1$. Each $p(x|\lambda_i)$ is a probability density function parameterized by λ_i.



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

Figure 4.5: GMM as the color model of an image

To illustrate the concept of using GMM to model the image color as shown in [56], Fig. 4.5 shows an image (a) with its color distribution in the Hue and Saturation space (b). The color distribution or the histogram of pixels in HS space (c) is modeled by GMM as shown in (d) for three Gaussian components and (e) for five Gaussian components.

Considering an image that has M Gaussian components $G(\lambda_i), i = 1, \ldots, M$

mixed together with M mixing coefficients $a_i, i = 1, \ldots, M$, a Gaussian component i can contribute to each pixel x of an image by the following probability density function:

$$p(x|\lambda_i) = \frac{1}{(2\pi)^{3/2}|\Sigma_i|^{1/2}} \exp(-\frac{1}{2}(x-\mu_i)^T\Sigma_i^{-1}(x-\mu_i)) \tag{4.2}$$

Here, λ_i is the parameter set of $G(\lambda_i)$ that includes the mean vector μ_i and the covariance matrix Σ_i (which contains the standard deviation vector σ_i).

Expectation-maximization (EM) [57] , [58] is a widely used method for estimating the parameter set (Θ) of the models in GMM using unlabeled data (pixels). The algorithm of EM iterates between two steps until converged. That is,

- **E-step** calculates the expectation of the log-likelihood over all possible assignments of pixels.
- **M-step** maximizes the expectation by differentiating according to the current parameters.

The spatial information is used in many applications such as [59], [29] in order to keep changes consistent inside neighborhoods. When applying EM with the image data, the spatial consistency is a critical issue since nearby pixels in the same area are usually similar to each other where as nearby pixels from different areas are different and separated along the boundary line. The modified version of EM with spatial consideration as in [24] is applied for an image as follows:

E-step: Let N represent the total number of pixels, and pixel $x \in \{x_n\}, n = 1, \ldots, N$. The probability $p^{i,x}$ that a pixel x belongs to the i^{th} Gaussian, $G(\lambda_i)$, is calculated as:

$$p^{i,x} = \frac{a_i p(x|\lambda_i)}{\sum_{i=1}^{M} a_i p(x|\lambda_i)} \tag{4.3}$$

The modified version of EM when considering spatial information, subject to $\sum_{i=1}^{M} p_t^{i,x} = 1$, can be derived by:

$$p_t^{i,x} = p_{t-1}^{i,x} + p_{t-1}^{i,\Omega_\epsilon} \tag{4.4}$$

where $p_0^{i,x}$ is initialized by Eq. 4.3 where all parameters come from the k-mean algorithm. The symbol p_t^{i,Ω_ϵ} represents the probability that the neighborhood Ω_ϵ of the pixel x belongs to region i at time t.

$$p_t^{i,\Omega_\epsilon} = \frac{1}{Z} \sum_{x_\epsilon \in \Omega_\epsilon} D(x, x_\epsilon) p_t^{i,x_\epsilon} \tag{4.5}$$

where $x_\epsilon \in \Omega_\epsilon$ is an arbitrary neighbor pixel in the neighborhood that has the pixel x as a center pixel, $Z = \sum_i \sum_{x_\epsilon \in \Omega_\epsilon} D(x, x_\epsilon) p_t^{i,x_\epsilon}$ is the normalization factor. p_0^{i,Ω_ϵ} is also derived from Eq. 4.3 where all parameters come from the k-mean algorithm.

$D(x, x_\epsilon)$ is used to control the color and spatial smoothness in the neighborhood Ω_ϵ. Bilateral filter [60] is adopted to perform both color and spatial smoothness with two parameters δ_c and δ_s. To be clear with the symbols, let $x(u, v) \in R^3$ be a pixel x at the position (u, v) and $x_\epsilon(u_\epsilon, v_\epsilon)$ be a neighbor pixel x_ϵ at the position (u_ϵ, v_ϵ). The smoothness control in the neighborhood can be derived as:

$$D(x, x_\epsilon) = \exp\left(-\frac{(u-u_\epsilon)^2 + (v-v_\epsilon)^2}{\delta_s}\right) \exp\left(-\frac{|x - x_\epsilon|^2}{\delta_c}\right) \tag{4.6}$$

The size of the neighborhood can affect the smoothness as described in [24]. It the size is too large, the region boundaries are over-smoothed, On the other hand, if the size is too small, the results will show color artifacts around the boundary due to the binary boundary as usually happened in the hard segmentation.

M-step: To maximize the expectation, the parameters are updated for the next iteration. For region i or Gaussian $G(\lambda_i)$, the new values of the weight $\tilde{a_i}$, the mean $\tilde{\mu_i}$ and the standard deviation $\tilde{\Sigma_i}$ are re-estimated as:

$$\tilde{a_i} = \frac{\sum_{n=1}^{N} p^{i,x_n}}{N} \tag{4.7}$$

$$\tilde{\mu_i} = \frac{\sum_{n=1}^{N} x_n p^{i,x_n}}{\tilde{a_i} N} \tag{4.8}$$

$$\tilde{\Sigma_i} = \frac{\sum_{n=1}^{N} p^{i,x_n} (x_n - \tilde{\mu_i})(x_n - \tilde{\mu_i})^T}{\tilde{a_i} N} \tag{4.9}$$

To apply EM for the image color model, additional tasks are added to reach the optimal solution of GMM.

Usually, the initial values of a_i and λ_i (including μ_i and Σ_i), are calculated by *k-means* clustering algorithm. The k-mean clustering requires the number of cluster k at the beginning step and returns k clusters as regions of the image. The k-means clustering technique can be used as hard segmentation for an image. A cluster i is a region i which can calculate its mean μ_i and standard deviation Σ_i from its member pixels. The number of members of region i, $|R_i|$, is also calculated as the prior weight a_i for region i by:

$$a_i = \frac{|R_i|}{\sum_{i=1}^{k} |R_i|}$$

Since $R_i \bigcap R_j = \emptyset, i \neq j$ in k-means clustering, $\sum_{i=1}^{k} |R_i| = N$, where N is the number of pixels. Hence, a_i can be rewritten as: $a_i = |R_i|/N$.

In experiment, k is the initial number of the image regions in the initial step of EM. It tends to decline later on. Therefore, the k should be set a bit high, approximately $6 - 8$. Along the course of EM, regions can be merged if their means are close to one another enough. Eventually, the number of the regions will be reduced to the optimal number at the final step. Following is the algorithm of EM with spatial consideration.

Algorithm of EM with spatial consideration

Input: an image I
Initialization:

- At time $t = 0$, initialize values of a_i^0, μ_i^0, and Σ_i^0 by k-means algorithm.
- Set a guess number of components k to a high number and perform k-mean several times to give a better initialization and also avoid sub-optimal results.

EM Loop $\mathbf{t \geq 1}$:

Step 1. Expectation
Given the initial parameters from the k-mean clustering, the probability density function of each Gaussian can be derived by Eq. 4.2. Then, the probability distribution with spatial consideration is estimated according to Eq. 4.4. In addition to the general EM, the spatial consideration takes the neighbor pixels into account as in Eqs. 4.5 and 4.6.

Step 2. Maximization
For each Gaussian component i, re-estimate a_i^t, μ_i^t, and Σ_i^t using Eqs. 4.7, 4.8, and 4.9, respectively.

Step 3. Refine Models
Given the new estimation of Gaussian parameters from the previous step, for every pair of Gaussian components in the same image, e.g., $G(\lambda_i^t)$ and $G(\lambda_j^t)$, if $|\mu_i^t - \mu_j^t| < \delta$ for small δ, they are merged and parameters are then re-estimated by:

$$\tilde{a_i^t} = a_i^t + a_j^t \tag{4.10}$$

$$\tilde{\mu_i^t} = \frac{\mu_i^t + \mu_j^t}{2} \tag{4.11}$$

$$\tilde{\Sigma_i^t} = \sum_{n=1}^{N} (x_n - \tilde{\mu_i^t})(x_n - \tilde{\mu_i^t})^T \tag{4.12}$$

Note that the merged Gaussian is assigned to the Gaussian i where as the Gaussian j is crossed out. If there are more than one pair fitting in this criteria, only pair that has the minimum difference gets a chance to be merged.

Step 4. Stopping Criteria
If the difference of means and variances between all corresponding pairs of Gaussian components between iterations t and $t - 1$ is small enough, and there is no Gaussian merging in current iteration t, stop. Otherwise, go back to step 1 and begin iteration $t + 1$.

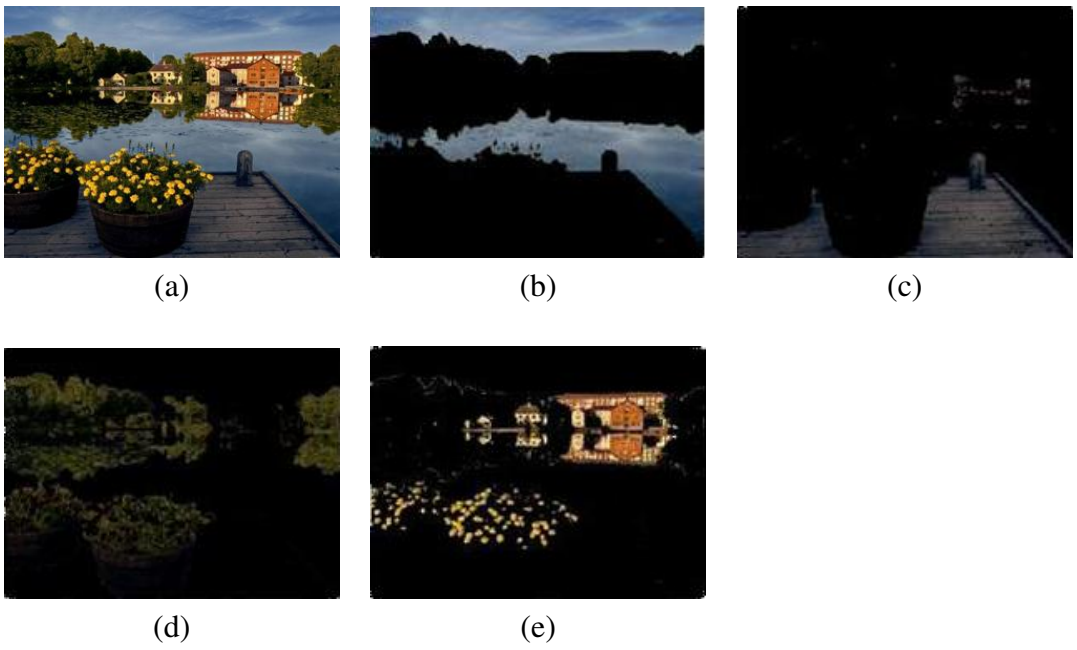
Return: the optimal number of Guassians

Region Extraction

Once the image models are defined, regions can be extracted and kept or saved for the region exploration. Given an image I with N pixels and its GMM, a region i, (R_i), is derived by multiplying region probabilities p^{i,x_n} (as alpha values) with an image pixels x_n where $x_n \in I$ as the following equation:

$$R_i = x_n p^{i,x_n} \quad n = 1, \ldots, N \tag{4.13}$$

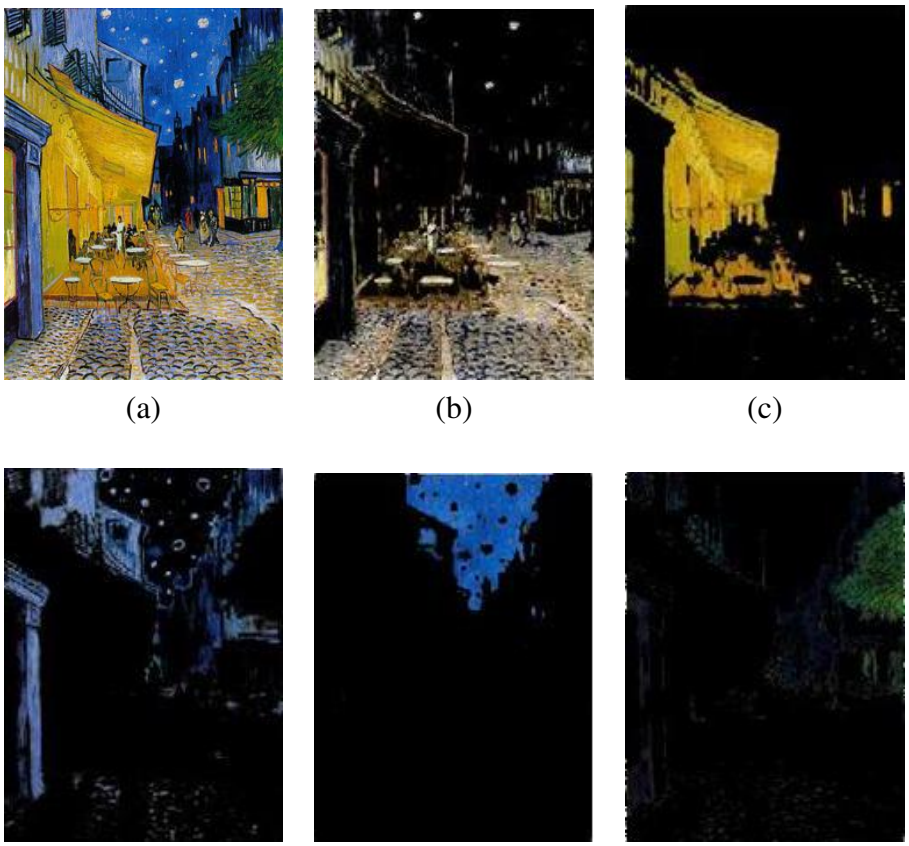
To illustrate the concept of the proposed technique, one example of interactive color transfer by region exploration is shown along the process. The process begins with the region definition by GMM of the source image shown in Fig. 4.6.(a). The regions are extracted by Eq. 4.13 according to the derived source models. The four extracted regions are shown in Fig. 4.6.(b)-(e). Similarly, Fig. 4.7 also shows the target image (a) and its five regions (b)-(f). Note that the source image comes from [61] for experiments only. The target image is the painting from Van Gogh, "The Café Terrace on the Place du Forum, Arles, at Night."



(a) (b) (c)

(d) (e)

Figure 4.6: Source image and its regions



(a) (b) (c)

(d) (e) (f)

Figure 4.7: Target image and its regions

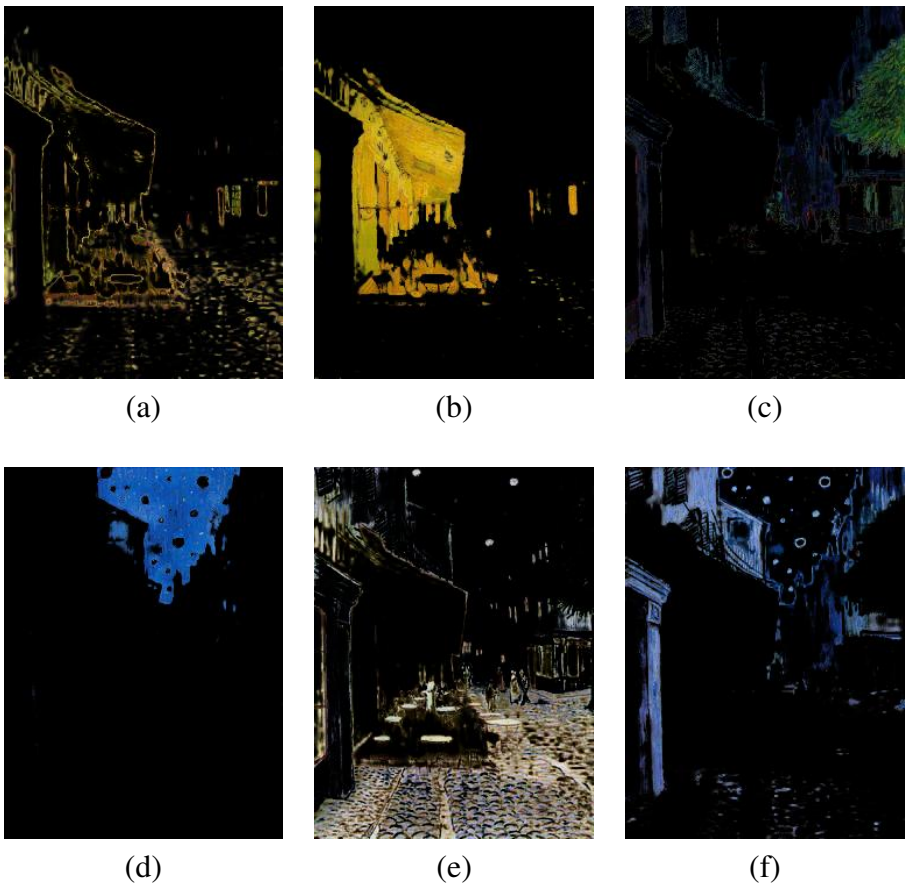
Defining Meaningful Regions

Regions derived from GMM are considered only color and spatial information which can not guarantee regions will have meaning to a user. For example, a sky region may consist of two Gaussian models representing the white cloud area and the blue empty space. We might either merge these two models into one for the sky region, or mark them as the region of sky. Sometimes, two objects with similar colors may be included into the same model. Separating them apart will make regions more meaningful. For example, orange flowers which are near an orange wall of the building should be in a different model from the one of the building.

Since a human has a better judgement defining meaningful regions, our interactive approach let a user send feedbacks back to the algorithm in order to increase meaningfulness of the regions derived from the EM algorithm. The user feedbacks can be achieved by using region merging and region marking.

Region merging According to the EM technique, the number of regions is initiated by k-mean clustering. The number is usually set a bit high in order to cover all possible regions in the image. The EM technique then evolves and tries to optimize the number of regions by merging a pair of regions that close to each other using color means as criteria. By this method, the regions may have no meaning if the merged threshold is set too high and causes too many regions. The suitable merged threshold is vary and depends on the characteristic of an individual image.

Figure 4.8 show an example in case of setting the merged threshold too high which causes too many regions.



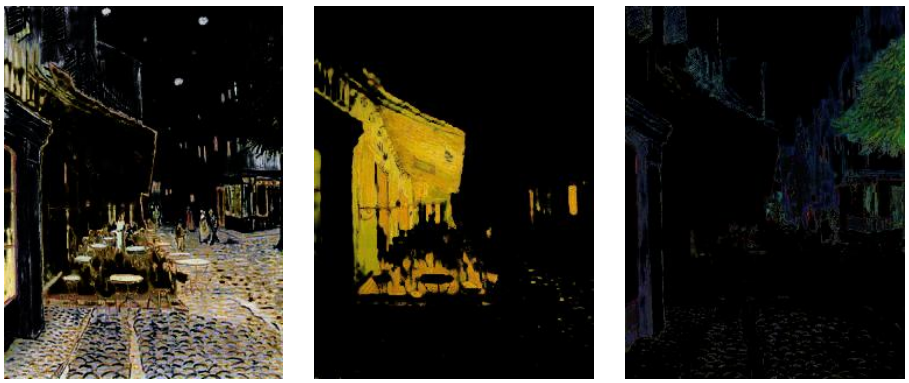
(a) (b) (c)

(d) (e) (f)

Figure 4.8: Too many regions by EM

Our method lets a user force regions to be merged by a user's judgement. A user can evaluate the results once the EM converged and then select a pair of regions to be merged. The merging process is responsive with small calculation as explained in the EM algorithm above. The user can continue merging regions until satisfied.

Figure 4.9 shows the region merging process according to the regions in Fig. 4.8. Suppose that a user needs to merge regions in Fig. 4.8.(a) and (e), the user simply selects both of them and forces them to be merged. The merged region is shown in Fig. 4.9.(a) and the rest regions (b)-(e) are kept unchanged. The model parameters and region probabilities of the merged region can be computed from Eqs. 4.11, 4.12, and 4.10.



(a) (b) (c)

(d) (e)

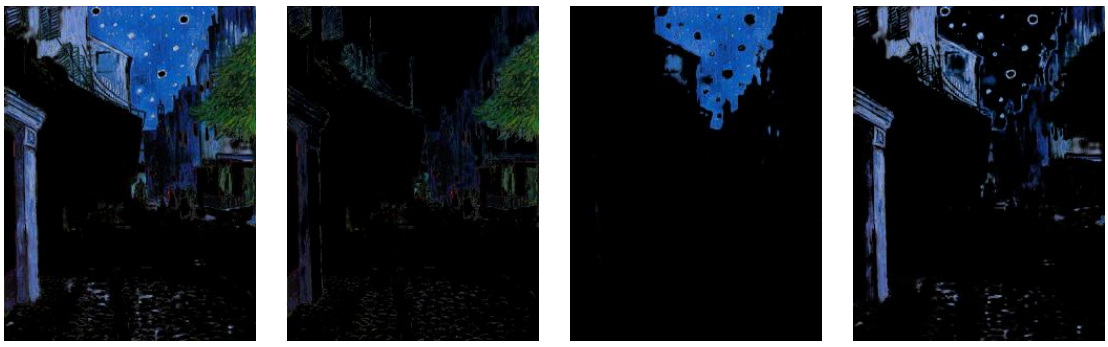
Figure 4.9: Regions are merged by the merging feature

Region marking At the first glance, GMM can reduce the complexity of color distribution for each region. However, regions may turn out to be fragments without meanings at all. Region merging can solve the problem; however, it might increase the complexity of the merged region again if overused. As a result, color transfer between merged regions may still causes color artifacts.

It seems like this tradeoff needs a good balance to achieve both objectives.

The alternative method called region marking is introduced. Regions are marked and grouped together to represent a meaningful region without merging models. The marked region is a subset or a group of regions and does not have its own model. The regions belong to the marked region still have their own models intact. In addition, the marked regions can be explored in the same way as a normal case, but in the smaller number of iteration because of the smaller number of marked regions.

Figure 4.10 shows the region marking process according to the regions in Fig. 4.8. Suppose that a user needs to mark regions in Fig. 4.8.(c), (d) and (f), the user simply selects them to be marked. The appearance of the marked region shown in Fig. 4.10.(a) looks like the region derived by region merging. However, the marked region is just a subset of regions which consists of regions shown in Fig. 4.10.(b), (c) and (d).



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

Figure 4.10: Regions are marked by the marking feature

4.4.2 Region Exploration

Region exploration is an iterative process. Each iteration considers only one source region versus each target region. The flow of the exploration is straightforward, e.g., exploring just one round from the first source region until the last one. However, the exploration process is not limited only in this scenario. It can be repeated by navigating back and forth along source regions many times until a user is satisfied.

For each iteration, a user can evaluate the color-transferred results and select one of them, or just keep exploring to the next region without any change. If a user selects one of the results, colors of the considered source region will be transferred

according to the selection, and the source image will be updated for the next iteration. A user can make any change at any stage of the exploration process, and then get any result as many as desired from any stage while exploring.

Using the exploration approach, a user has more options to explore around all result possibilities and evaluate the would-be results while exploring. This process should be responsive in order to keep attention from a user. Hence, the exploration part is separated from the region definition part. There are many sub-tasks in each iteration of the exploration process which need to be done as followings.

Region Mapping

The number of extracted regions is not large in general cases. However, the number of region-mapping possibilities can be huge. For example, if a source image has n regions and a target image has m regions. All possibilities of region mapping P between source and target regions are:

$$P = (m + 1)^n \tag{4.14}$$

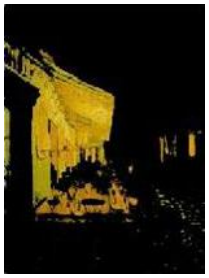
1 is added because a source region is allowed to keep unchanged or not to map to any target regions. To limit the number, the automatic methods in [24], [26] used rules for region mapping. The rules are expected to produce a good and correct result. Nevertheless, the result sometimes can not guarantee a user's satisfaction. The result evaluation is still subjective even though Xiang et al. [26] pushed efforts to define some criteria for objective evaluations.

The region exploration approach gives a user more control over a region-mapping process. A user may not intend to change colors all of the regions, but may need options to decide where to change colors or where to keep unchanged. The approach focus on a user's desire which is not limited by the rules. In the region exploration approach, there are no rules at all for region mapping. A user can explore and evaluate all result possibilities from each mapping and then make a decision for selection.

The region mappings between a source region and every target region occur every iteration of the region exploration without rules at all. Figure 4.11 shows examples of region mappings between a source region (a) from Fig. 4.6.(b) and all target regions

(a)

(b)



(c)

(d)

(e)

(f)

Figure 4.11: Region mapping

(b)-(f) from Fig. 4.7.(b)-(f).

The no-rules-at-all approach for region mappings causes a number of possibilities of the results. For example, the source image from Fig. 4.6.(a) which has 4 regions and the target image from Fig. 4.7.(a) which has 5 regions will have all possibilities in region mappings equal to 1296 according to Eq. 4.14 where $m = 5$ and $n = 4$. The number of possibilities is so huge that all the results can not fit in the user interface at once. Even all of the results can fit it, a user can not see them carefully, let alone evaluate them equally.

If the results from all possibilities are shown at once, a user will be overwhelmed by them to see and evaluate. Instead, the exploration approach lets a user navigate around source regions, one-by-one. By navigating one source region at a time, a user can focus only the source region in question; and the number of results is reduced to the number of target regions. It is easier for a user to handle with this number of results. A user is comfortable to compare the results in a small number for each stage and can save any result if satisfied. At last, a user has a chance to compare the saved results again.

Region Color Transfer

Since each region is represented by its own Gaussian model, color transfer between regions can be performed by a following equation which is a simple statistical transformation in $\ell\alpha\beta$ color space [15].

$$\tilde{x_s} = \frac{\sigma_{R_t}}{\sigma_{R_s}}(x_s - \mu_{R_s}) + \mu_{R_t} \tag{4.15}$$

where x_s is an arbitrary source pixel in a source region, R_s while R_t is a target region. Similar to the color transfer equation by Reinhard et al. [3], the resultant pixel $\tilde{x_s}$ is derived separately from each channel of $\ell\alpha\beta$.

To represent color transfer between the current source region Fig. 4.11.(a) and each target region Fig. 4.11.(b)-(f), there are a number of sub-tasks before showing the results to a user. First, the current source region is first decomposed from the source image and leave the background region for a later use when re-composition is needed. Each mapping between the current source region and each target region is then performed color transfer as in Eq. 4.15. Each color-transferred region and the source background are then re-composed to show a user the complete image results. Figure 4.12 shows the results of region color transfer from each region mapped pair according to Fig. 4.11. Notice that each image in Fig. 4.12 differs to each other only in the sky area and the water area according to the current source region, Fig. 4.11.(a).

Interactive User Interface

The user interface for the region exploration should be simple and responsive. A user can load the source and target images including their models which are saved from the region definition process. The user interface will show results from one iteration of the exploration process at a time.

The region exploration process will pick one of the source regions at the beginning stage, usually the first source region. The source region is decomposed from the source image as a foreground, and the rest region becomes a background. The source region are mapped to every target regions in order to perform color transfer. The outcomes are then recomposed with the background to produce the complete images which are shown to a user for consideration.

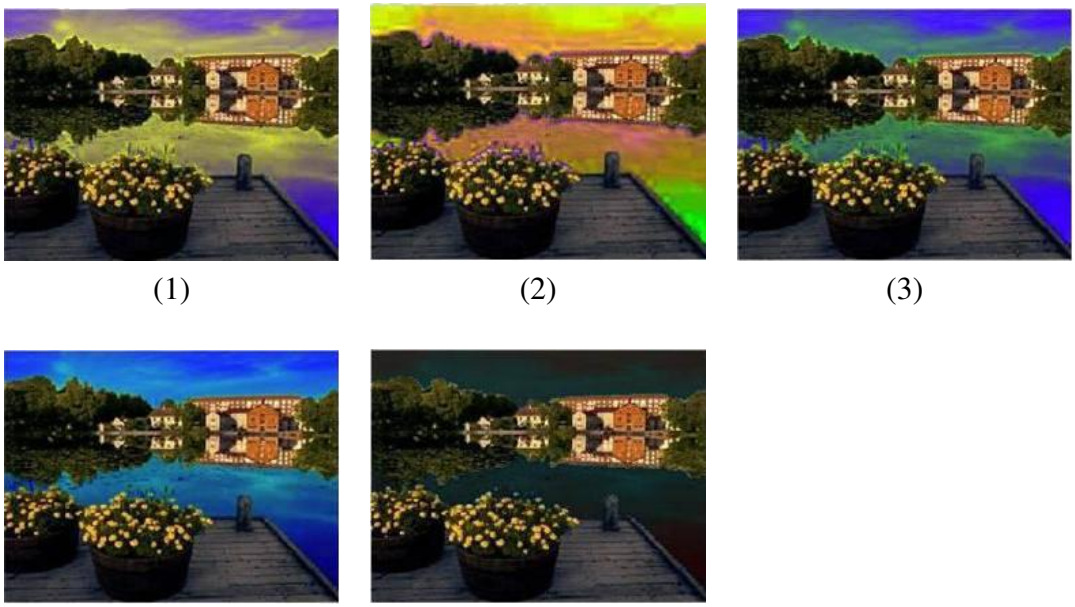


Figure 4.12: Color transfer between regions

Hence, the user will continue the exploration process by interacting with the user interface whether to select one of the results or explore more to the next region. If the user selects one of them, that result will be a new source image for the next iteration. The user also has a chance to save the selected result and get it back at the end of the exploration process. The user can explore more to the next source region without any selection to see more results. In this case, there is no effects to the current source region and also the current source image.

Figure 4.12 shows the results of region color transfer between the first source region Fig. 4.11.(a) and each target regions Fig. 4.11.(b)-(f). Hence, a user can see and compare the results from each region mapped pair. Assume that the user selects the result number 4. The current source region will be updated according to the user selection.

Figure 4.13 shows the consequence of the user selection from Fig. 4.12. Notice that the first source region, Fig. 4.6.(b) is now changed according to the user selection (4). Now, the user can evaluate the results for the next source region, Fig. 4.6.(c). Suppose that the user selects the result number 2, the same effects will be applied for the current selection and forwarded to the next iteration.

To summarize all steps involved in the region exploration process, below

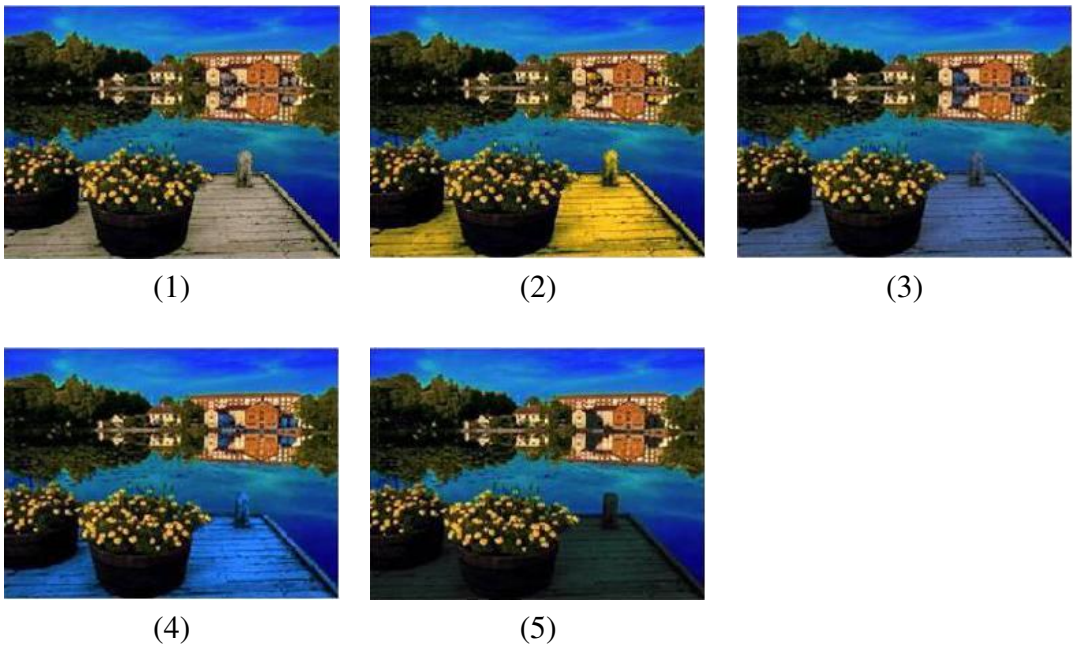


Figure 4.13: Region exploration response to the user selection

demonstrates the algorithm of the region exploration. The region definition of the source and target images is assumed to be accomplished before the region exploration begins.

Region Exploration Algorithm

Input: a source image I^s and source models M^s,
a target image I^t and target models M^t

Initialization:

- The current source image $I_c^s \leftarrow$ The original source image I^s
- Extract target regions R_j^t by: $R_j^t = p^{j,I^t} * I^t$

Exploration Loop: For each source region i:

Step 1. Decompose the current source image I_c^s into a source region R_i^s and a source background B_i^s. According to Eq. 4.13, the source region is extracted as a foreground by:

$$R_i^s = p^{i,I^s} * I_c^s$$

where p^{i,I^s} is probabilities for a region i of the source image I^s. The background B_i^s is derived by subtracting the current source image with the foreground as a following equation.

$$B_i^s = I_c^s - R_i^s = (1 - p^{i,I^s}) * I_c^s$$

Step 2. Perform region color transfer for each mapping between the source region R_i^s and each target region R_j^t by Eq. 4.15.

Step 3. Compose each transferred result R_{ij}^s with the source background B_i^s. To get the complete images I_{ij}^s, simply add them together as:

$$I_{ij}^s = R_{ij}^s + B_i^s$$

Step 4. Represent all results for a user to consider. The user has many options to respond such as:

(a) select a desired result which will make a program to update the current source image for the next loop, i.e., $I_c^s \leftarrow I_{ij}^s$;

(b) save any result I_k^r if the user is satisfied with it; that is, $I_k^r \leftarrow I_{ij}^s$;

(c) or explore more as much as desired by moving forward to the next source region or moving backward to the previous one if possible. Only exploring without any selection will not affect to the current source region.

Return: the saved results $I_k^r, k \geq 0$

4.5 Results and Discussion

The first scenario has one source and one target. This scenario shows the interactive color transfer between the source and target regions in one-round exploration, starting from the first source region until the last source region. This experiment uses the source image and source regions from Fig. 4.6 and the target images and target regions from Fig. 4.7.

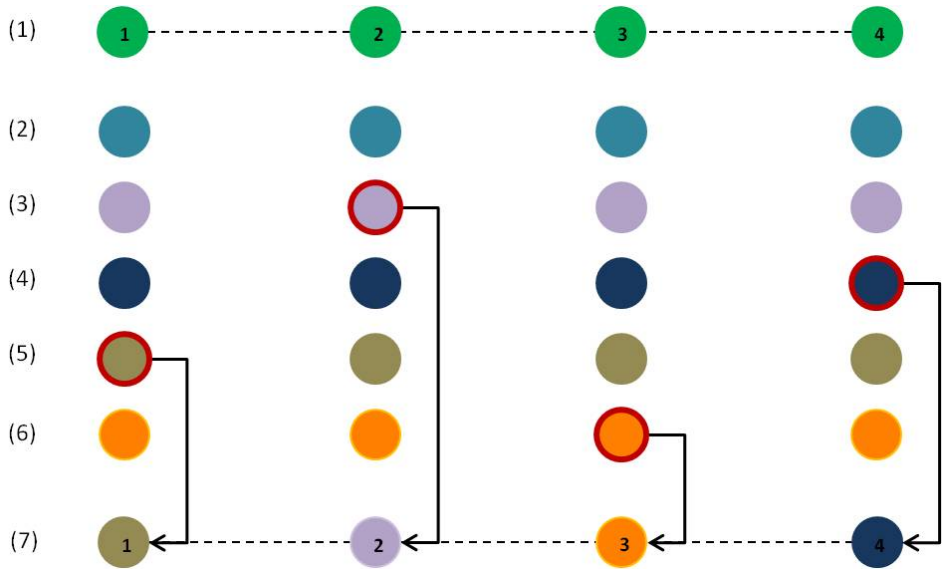


Figure 4.14: Illustration of the user interaction flow in the one-round-exploration scenario

To follow this experiment, the diagram in Fig. 4.14 shows the one-round exploration along the source regions starting from the first region and finishing at the last region. The green circles at the first row represent four source regions with their own original colors. The numbers inside them indicate the corresponding numbers of the source regions. The dash line connecting the first source region through the last one together illustrates the one-round exploration flow.

In each iteration (each column) of the exploration show the results from all mapped pairs between the current source region and each target region. The number of the results equals to five which is the number of the target regions (row 2-6). The results from different target regions are shown in different color circles. The results circled in red indicate user selections. The source regions will be updated according to

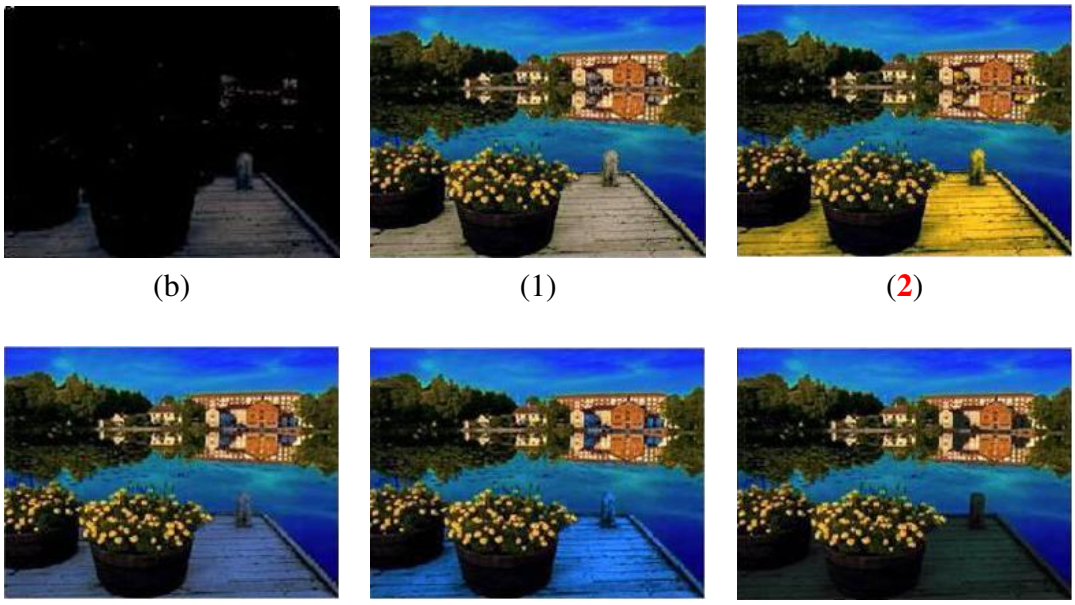
the selections. The last row shows the updated source regions whose colors are changed to the selected result colors. The color updates are illustrated by the arrow lines.

Figures 4.15 – 4.18 show the user interface of the same experiment in the scenario illustrated in the diagram in Fig. 4.14. The figure shows the results and the user selections in the sequence from the first source region (a) to the last one (d). The red and bold numbers under the results of each source region denote the user selections. Notice that the current source region is updated according to the user selection and apparently shown in the next iteration.

(a) (1) (2)

(3) (**4**) (5)

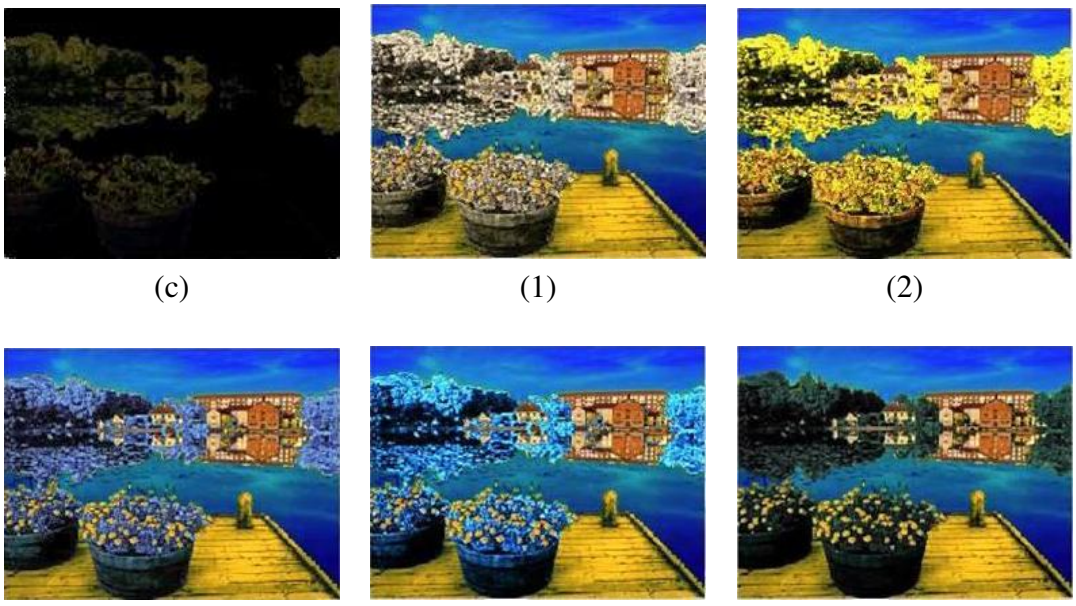
Figure 4.15: User selection for the first source region



(b) (1) (2)

(3) (4) (5)

Figure 4.16: User selection for the second source region



(c) (1) (2)

(3) (4) (5)

Figure 4.17: User selection for the third source region

(d) (1) (2)

(3) (4) (5)

Figure 4.18: User selection for the fourth source region

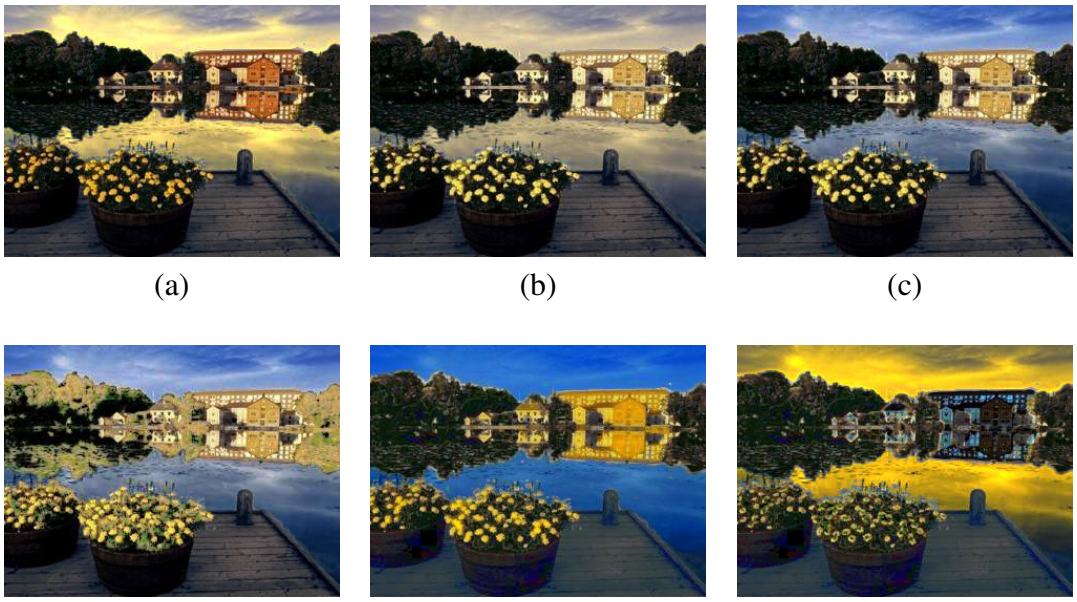
Figure 4.19 shows the final result according to the user selections shown in Fig. 4.15 – 4.18. In this scenario, the user has selected the result numbers 4, 2, 5, and 3 for the first region to the fourth region, respectively.

(a) Source image (b) Target image (c) Result image

Figure 4.19: The final result in the one-round-exploration scenario

During region exploring, a user can get more results as desired. Using the same source and target images as the previous scenario, Fig. 4.20 shows some possible results the user may get while exploring. Each result is constructed according a combination of the user selections.

Evaluation in color transfer is hard to measure since human eyes cannot



(a) (b) (c)

(d) (e) (f)

Figure 4.20: More results a user can get while exploring

detect small changes in colors, especially for individual pixel. Moreover, users usually have their subjective impression when they see the overall image. In the automatic approach of color transfer, a user may be not satisfied with the result even whose color appearance is close to the color appearance of the target image. Sometimes the color tones of the result are right, but the color locations might be wrong.

In the exploration approach, the color transfer method gives a user more controls to evaluate and adjust the desired result along the course of the exploration process. A user also has opportunities to repeat evaluating and adjusting the results many times until satisfied.

4.6 Conclusion

The interactive color transfer method using the exploration approach is proposed. The method gives a user more controls to define regions meaningfully and explore the results of all possible region mapped pairs. The region exploration gives an intuitive method to navigate, evaluate, and select the intermediate results. The final result is eventually elaborated by the user selections in the exploration process. The exploration technique is so responsive that the user can extensively refine the result until satisfied. While exploring, a user can easily get more desired results.

CHAPTER V
CONCLUSION

The first part of this thesis gives literature reviews of color transfer techniques proposed in recent years. We analyzed and categorized the techniques in three groups, which are global, local, and interactive color transfer. The thesis extends the state of the art of color transfer by presenting new algorithms for handling complex color distributions usually found in natural images. Our first method, global color transfer in the high dimensional space, uses nonlinear analysis to manipulate a nonlinear structure found in a complex color distribution. Without user intervention, our experiments showed no color artifacts in the results usually found in the existing global techniques. Our second method, interactive color transfer by region exploration, creates a new dimension in the interactive approach using the interactive exploration method. The proposed method gives users controls over the color transfer process that enhances the performance while reducing the human errors.

We review the contributions of our two color transfer techniques in the section 5.1. In section 5.2, we describes several possible extensions for the two techniques.

5.1 Contributions

The main contributions of this thesis are development and implementation of two color transfer techniques that enable (1) handling of complex color distributions and (2) more user interaction with the color transfer program. The contributions from the two proposed methods are listed below.

1. **Global color transfer in the high dimensional space**

   The technique of global color transfer in the high dimensional space computes the kernel matrices that capture nonlinear color structures of source and target pixels into the high dimensional space, the image space. The technique achieved the

color transfer goal by projecting both source and target pixels into their image spaces and then searching for the closest pairs between source and target pixels in the image spaces.

To the best of my knowledge, the work described in this thesis is the first occurrence that the nonlinear analysis in the high dimensional space is put to use in color transfer arena. Global color transfer in the high dimensional space can solve the complex color distribution problem without requirement of additional inputs from a user. Although the complex color distribution problem can be solved in many ways including techniques such as swatches determined by a user or local regions with simplified color distributions, the technique implemented in this thesis is advantageous in being global and automatic.

2. **Interactive color transfer by region exploration**

   Region exploration implemented in my research work took a different viewpoint from previous interactive color transfer techniques which usually tie the user interaction together with the region definition and region mapping processes. This particular technique is designed in two separated parts, called region definition and region exploration.

   The region definition is done automatically by modeling regions with GMM. Additional features such as region merging and region marking are added to the region definition in order to let users to refine regions more meaningfully. This approach reduces human errors with trials and errors in order to manually define regions with interactive tools.

   The region exploration explores all possibilities of color-transferred results from region mapped pairs between source and target regions instead of using criteria for region mapping or being forced mapping by a user. The region exploration also cooperates with the user evaluation while exploring and lets users get more desirable results. Being separated from the region definition, the exploration process focuses on region color transfer and navigation. This approach allow users to revise the results easier and make the exploration process responsive and effective with simply clicking on the desired results.

5.2 Future Work

1. **Global color transfer in the high dimensional space**

   Our results showed no color artifacts or colors which are not shown in the target image. However, the searching method can cause a low color dynamic range in the results. Also, our technique requires much of computation time and memory usage. For a large image size, it might not be feasible to perform color transfer by this technique. Some sampling techniques can be alleviated with the quality dropped as expected. We are going to explore more speed-up KPCA techniques in order to handle the high resolution images.

2. **Interactive color transfer by region exploration**

   One issue in this method is there is a large number of resultant choices a user need to explore, especially when source and target images have too many regions. Some of result qualities may not reach to a user's preferences, but a user still has to explore them without a chance to avoid exploring them.

   According to the issue of a large number of resultant choices, we think of some criteria to limit the number of choices. This might be some heuristic or objective rules that apply to all possible results in order to show a smaller set of possible results. A user may opt for this rules or not depends on a user's desire.

   We expect that the region definition can be used further as a tool to define an object meaningfully. Existing techniques such as hard segmentation or matting are used to define and extract an object. The extracted object may consist of several regions which still have a complex color distribution which can cause color artifacts when applied with linear color transfer. Instead, our technique can break an object into separated regions which can be performed color transfer between specific regions.

   There are not many work applying color transfer to video images. The consistency among adjacent frame images is a major problem. However, color transfer techniques can help reducing cost in the film production. Particularly, object color transfer seems to be feasible and interesting in a movie application.

REFERENCES

[1] Jeschke, E. R. (2010). Converting color images to b&w. http://www.gimp.org/tutorials/Color2BW/.

[2] Pitié, F., Kokaram, A. C., and Dahyot, R. (2005). N-dimensional probability density function transfer and its application to colour transfer. *ICCV*, pp. 1434–1439.

[3] Reinhard, E., Adhikhmin, M., Gooch, B., and Shirley, P. (2001). Color transfer between images. *IEEE Computer Graphics & Applications*, 21(5), 34–41.

[4] Neumann, A. and Neumann, L. (2005). Color style transfer techniques using hue, lightness and saturation histogram matching. *Workshop on Computational Aesthetics*, pp. 111–122.

[5] Welsh, T., Ashikhmin, M., and Mueller, K. (2002). Transferring color to greyscale images. *SIGGRAPH 2002, Computer Graphics Proceedings*, pp. 277–280.

[6] Konushin, V. and Vezhnevets, V. (2006). Interactive image colorization and recoloring based on coupled map lattices. *Graphicon2006, International conference on computer graphics & vision*.

[7] Efros, A. A. and Leung, T. K. (1999). Texture synthesis by non-parametric sampling. *International Conference on Computer Vision*, pp. 1033–1038.

[8] Efros, A. A. and Freeman, W. T. (2001). Image quilting for texture synthesis and transfer. *SIGGRAPH 2001, Computer Graphics Proceedings*, pp. 341–346.

[9] Wei, L. Y. and Levoy, M. (2000). Fast texture synthesis using tree-structured vector quantization. *SIGGRAPH 2000: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pp. 479–488.

[10] Hertzmann, A., Jacobs, C. E., Oliver, N., Curless, B., and Salesin, D. H. (2001). Image analogies. *SIGGRAPH 2001, Computer Graphics Proceedings*, pp. 327–340.

[11] Ashikhmin, M. (2001). Synthesizing natural textures. *I3D 2001: Proceedings of the 2001 symposium on Interactive 3D graphics*, pp. 217–226.

[12] Ashikhmin, M. (2003). Fast texture transfer. *IEEE Computer Graphics and Applications*, 23(4), 38–43.

[13] Chen, T., Yu, J., and Peng, Q. (2006). Style conversion of cartoon animation. *Proceedings of Technologies for E-Learning and Digital Entertainment, Edutainment 2006*, Lecture Notes in Computer Science, 3942, pp. 1074–1079.

[14] Xie, X., Tian, F., and Seah, H. S. (2007). Feature guided texture synthesis (fgts) for artistic style transfer. *DIMEA 2007: Proceedings of the 2nd international conference on Digital interactive media in entertainment and arts*, pp. 44–49.

[15] Ruderman, D. L., Cronin, T. W., and Chiao, C. C. (1998). Statistics of cone responses to natural images: implications for visual coding. *JOSA*, 15(8), 2036–2045.

[16] Abadpour, A. and Kasaei, S. (2004). A fast and efficient fuzzy color transfer method. *Signal Processing and Information Technology, 2004. Proceedings of the Fourth IEEE International Symposium on*, pp. 491–494.

[17] Abadpour, A. and Kasaei, S. (2007). An efficient pca-based color transfer method. *Journal of Visual Communication and Image Representation*, 18(1), 15–34.

[18] Kotera, H. (2005). A scene-referred color transfer for pleasant imaging on display. *Image Processing, 2005. ICIP 2005. IEEE International Conference on*, Sept., pp. II–5–8.

[19] Xiao, X. and Ma, L. (2006). Color transfer in correlated color space. *VRCIA '06: Proceedings of the 2006 ACM international conference on Virtual reality continuum and its applications*, pp. 305–309.

[20] Grundland, M. and Dodgson, N. A. (2005). Color histogram specification by histogram warping. *Proceedings of SPIE*, pp. 610–621.

[21] Gonzalez, R. C. and Woods, R. E. (2002). *Digital Image Processing*, 2 edition. Prentice Hall, Upper Saddle River.

[22] Pitié, F. (2006). Statistical Signal Processing Techniques for Visual Post-Production. PhD thesis University of Dublin, Trinity College.

[23] Pitié, F., Kokaram, A., and Dahyot, R. (2007). Automated colour grading using colour distribution transfer. *Journal of Computer Vision and Image Understanding*, 107(1-2), 123–137.

[24] Tai, Y. W., Jia, J., and Tang, C. K. (2005). Local color transfer via probabilistic segmentation by expectation-maximization. *CVPR 2005: Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 1*, pp. 747–754.

[25] Tai, Y.-W., Jia, J., and Tang, C.-K. (2007). Soft color segmentation and its applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29, 1520–1537.

[26] Xiang, Y., Zou, B., and Li, H. (2009). Selective color transfer with multi-source images. *Pattern Recognition Letters*, 30(7), 682–689.

[27] Chang, Y., Saito, S., and Nakajima, M. (2003). A framework for transfer colors based on the basic color categories. *CGI 2003: Proceedings of the Computer Graphics International*.

[28] Greenfield, G. R. and House, D. H. (2003). Image recoloring induced by palette color associations. *WSCG*, 11(1), 3–7.

[29] Wang, J. and Cohen, M. (2005). An iterative optimization approach for unified image segmentation and matting. *ICCV*.

[30] Luan, Q., Wen, F., and Xu, Y. Q. (2007). Color transfer brush. *PG 2007: Proceedings of the 15th Pacific Conference on Computer Graphics and Applications*, pp. 465–468.

[31] Levin, A., Lischinski, D., and Weiss, Y. (2006). A closed form solution to natural image matting. *CVPR*.

[32] Agarwala, A., Dontcheva, M., Agrawala, M., Drucker, S., Colburn, A., Curless, B., Salesin, D., and Cohen, M. (2004). Interactive digital photomontage. *SIGGRAPH 2004, Computer Graphics Proceedings*.

[33] Sun, J., Jia, J., Tang, C. K., and Shum, H. Y. (2004). Poisson matting. *ACM Trans. Graph.*, 23(3), 315–321.

[34] Porter, T. and Duff, T. (1984). Compositing digital images. *SIGGRAPH 1984, Computer Graphics Proceedings*, pp. 253–259.

[35] Pérez, P., Gangnet, M., and Blake, A. (2003). Poisson image editing. *ACM Trans. Graph.*, 22(3), 313–318.

[36] Maslennikova, A. and Vezhnevets, V. (2007). Interactive local color transfer between images. *Graphicon2007, International conference on computer graphics & vision*.

[37] Dong, Y. and Xu, D. (2009). Interactive local color transfer based on coupled map lattices. *IEEE CAD/Graphics*.

[38] Vezhnevets, V. and Konouchine, V. (2005). Growcut–interactive multi-label n-d image segmentation by cellular automata. *Graphicon2005, International conference on computer graphics & vision*.

[39] Schölkopf, B., Smola, A., and Müller, K. R. (1998). Nonlinear component analysis as a kernel eigenvalue problem. *Neural Comput.*, 10(5), 1299–1319.

[40] Schölkopf, B., Burges, C. J. C., and Smola, A. J. (eds.) (1999). *Advances in kernel methods: support vector learning*. MIT Press, Cambridge, MA, USA.

[41] Kim, K. I., Franz, M. O., and Scholköpf, B. (2005). Iterative kernel principal component analysis for image modeling. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(9), 1351–1366.

[42] Cheng, S.-C. and Hsia, S.-C. (2003). Fast algorithms for color image processing by principal component analysis. *Journal of Visual Communication and Image Representation*, 14(2), 184 – 203.

[43] Schölkopf, B., Mika, S., Smola, A., Rätsch, G., and Müller, K.-R. (1998). Kernel pca pattern reconstruction via approximate pre-images.

[44] Mika, S., Schölkopf, B., Smola, A. J., Müller, K.-R., Scholz, M., and Rätsch, G. (1999). Kernel PCA and de–noising in feature spaces. In Kearns, M. S., Solla, S. A., and Cohn, D. A. (eds.), *Advances in Neural Information Processing Systems 11*.

[45] Kwok, J. T. and Tsang, I. W. (2004). The pre-image problem in kernel methods. *IEEE Transactions on Neural Networks*, 15(6), 1517 – 1525.

[46] Arias, P., Randall, G., and Sapiro, G. (2007). Connecting the out-of-sample and pre-image problems in kernel methods. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 18-23 jun.

[47] Barros, J. E., French, J. C., Martin, W. N., Kelly, P. M., and Cannon, T. M. (1996). Using the triangle inequality to reduce the number of comparisons required for similarity-based retrieval. In Sethi, I. K. and Jain, R. C. (eds.), *Proc. of SPIE*, pp. 392–403.

[48] Franc, V. and Hlavc, V. (2006). Greedy kernel principal component analysis. *Cognitive Vision Systems*, Lecture Notes in Computer Science, 3948, pp. 87–105.

[49] Chin, T. J. and Suter, D. (2006). Improving the speed of kernel pca on large scale datasets. *AVSS '06: Proceedings of the IEEE International Conference on Video and Signal Based Surveillance* , p. 41.

[50] Hasler, S. and Susstrunk, S. (2003). Measuring colorfulness in real images. *Proc SPIE Eletronic Imaging*, pp. 87–95.

[51] Grundland, M. and Dodgson, N. A. (2005). Color search and replace. *Computational Aesthetics in Graphics, Visualization and Imaging*, pp. 101–109.

[52] Wen, C.-L., Hsieh, C.-H., Chen, B.-Y., and Ouhyoung, M. (2008). Example-based multiple local color transfer by strokes. *Comput. Graph. Forum*, 27(7), 1765–1772.

[53] Xiao, X., Ma, L., and Kunze, M. (2006). Object-based image recoloring using alpha matte and color histogram specification. *VSMM*, pp. 194–203.

[54] Jia, J., Sun, J., Tang, C.-K., and Shum, H.-Y. (2006). Drag-and-drop pasting. *ACM Transactions on Graphics (SIGGRAPH)*, 25(3), 631–637.

[55] Fu, X., Guo, H., Wang, Y., Liu, T., and Li, H. (2008). Signal Processing for Image Enhancement and Multimedia Processing. Springer US.

[56] Shapira, L., Shamir, A., and Cohen-Or, D. (2009). Image appearance exploration by model-based navigation. *Comput. Graph. Forum*, 28(2), 629–638.

[57] Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society, series B*, 39(1), 1–38.

[58] Bilmes, J. (1998). A gentle tutorial of the em algorithm and its application to parameter estimation for gaussian mixture and hidden markov models. Technical report. University of California, Berkeley.

[59] Levin, A., Lischinski, D., and Weiss, Y. (2004). Colorization using optimization. *ACM Transactions on Graphics*, 23(3), 689–694.

[60] Tomasi, C. and Manduchi, R. (1998). Bilateral filtering for gray and color images. *ICCV '98: Proceedings of the Sixth International Conference on Computer Vision*, pp. 839–846.

[61] Luan, Q., Wen, F., Or, D. C., Liang, L., Xu, Y. Q., and Shum, H. Y. (2007). Natural image colorization. *SR 2007 Rendering Techniques*, pp. 309–320.

APPENDIX

REGION EXPLORATION USER INTERFACE

Our experimental program for interactive color transfer by region exploration has two parts. The first part is region definition which lets a user load an image in order to define regions and save the regions for later use. The second part is region exploration which lets a user load source and target regions defined in the region definition part, and perform region color transfer by a exploring method.

A.1 Region Definition

At the beginning of the program in Fig A.1, the user interface shows a blank area (a). When clicking on the image icon , a user can select an image in order to define regions. In this case, the user selected the image of cafe terrace. The program run and return the regions as shown in (b).

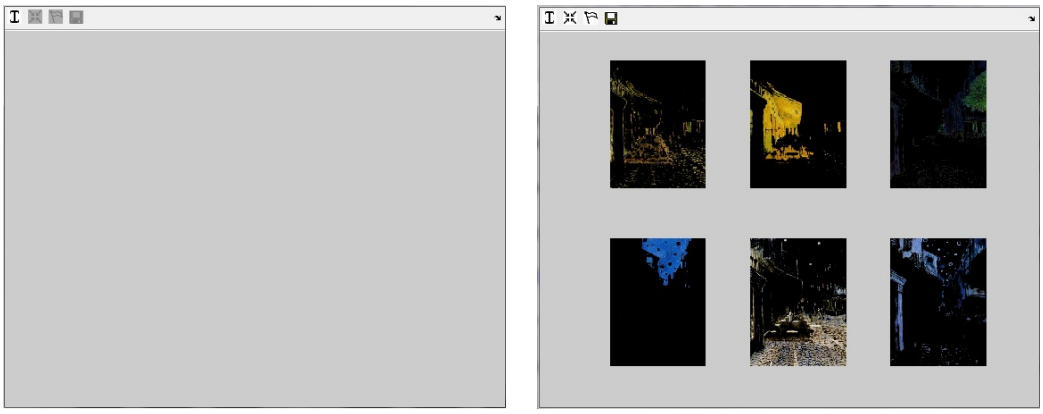
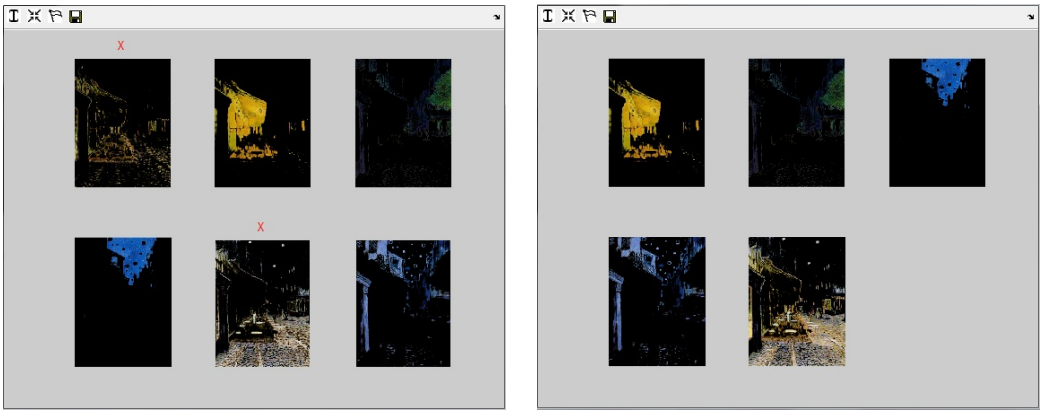


Figure A.1: User interface: region definition

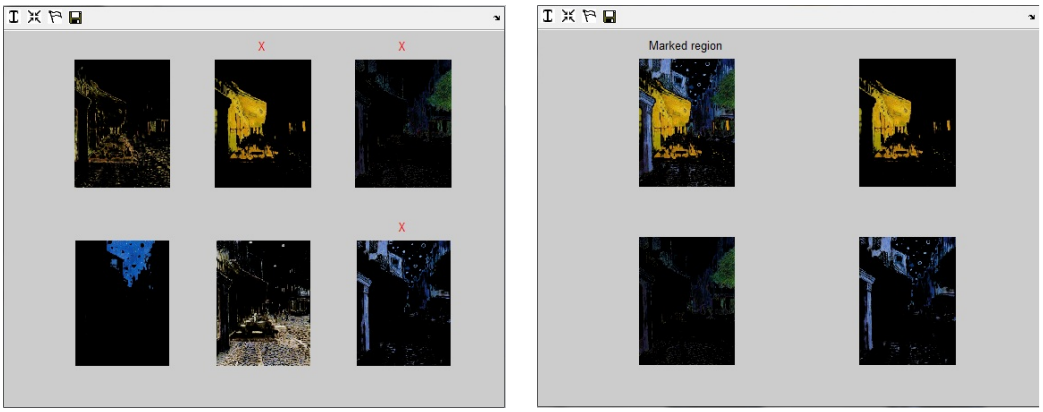
Figure A.2 demonstrates the process of region merging. On the user interface, a user can select the regions of interest. The selected regions will have a red cross (X) over them to indicate the user selections (a). When clicking on the merge icon ,



(a) (b)

Figure A.2: User interface: region merging

Figure A.3 demonstrates the process of region marking. On the user interface, a user can select the regions of interest. Similarly to region merging, the selected regions will have a red cross over them to indicate the user selections (a). When clicking on the mark icon , the program marks the selected regions and return the marked region and its sub-regions as shown in (b).



(a) (b)

Figure A.3: User interface: region marking

When a user finishes region merging or region marking, the current regions and the marked region can be saved by clicking on the save icon . There will be a save dialog showing up and letting a user put a file name.

A.2 Region Exploration

At the beginning of the program in Fig. A.4, the user interface shows a blank area (a) with three scenario buttons on the toolbar. The scenario icon 1:1 is for the one-source-one-target scenario, 1:K for the one-source-many-target scenario, and 1:C for the marked-source-target-collection scenario.

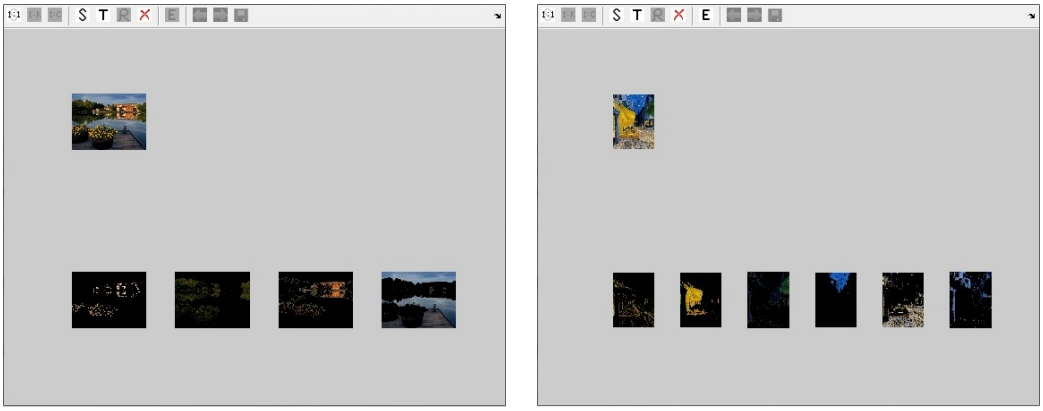
One-source-one-target scenario When selecting the one-source-one-target scenario icon, the source image icon S and the target image icon T are now available to load source and target regions respectively, see (b).

(a) (b)

Figure A.4: User interface: scenario selection

Figure A.5.(a) shows the source image and its regions after clicking on the source image icon and selecting a source file. The delete icon ✕ is now ready to delete the current image, the source image in this case. Similarly, a user can now load target image and its regions shown in (b). Since both source and target regions are loaded up, the explore icon E is available to explore the possible results of region color transfer between pairs of the current source region and all target regions.

Figure A.6 demonstrates how to explore by using the navigator icons: the backward icon ⬅ and the forward icon ➡. A user can select a desired result by clicking on one of the results shown in (a). The red cross (X) over the result indicates the selection. Clicking on navigator icons, a user can move backward or forward along the source regions. If a user select a result of interest and then navigate to the next source region, the transferred colors of the current source region will go along to the next explored loop



(a) (b)

Figure A.5: User interface: one-source-one-target scenario

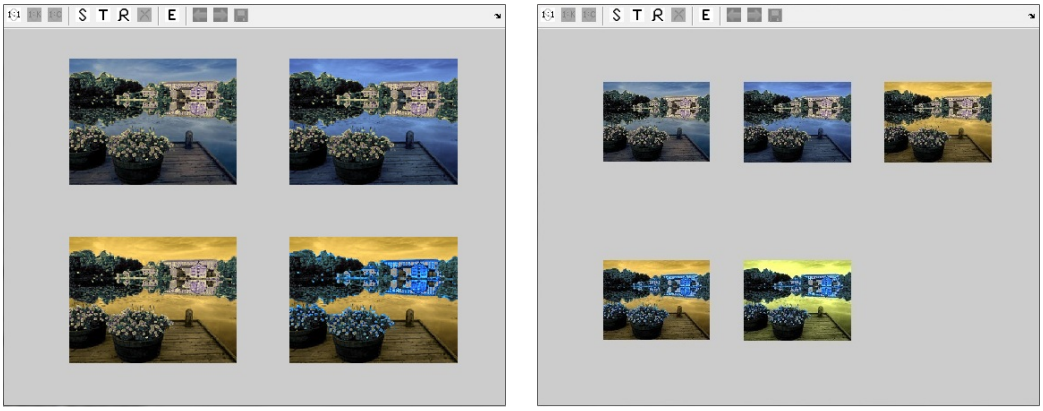
as shown in (b).

(a) (b)

Figure A.6: User interface: region exploration

A user can save any results while exploring by selecting a result and then click the save icon. The saved results can be shown as in Fig. A.7.(a) when clicking on the result icon. Specifically, the selected result as shown in Fig. A.6.(a) can be saved to the result list and shown in Fig. A.7.(b).

One-source-many-target scenario This scenario is similar to the One-source-one-target scenario except that the target regions can come from many target images. Figure A.8 shows the user interface at the beginning of this scenario (a), a loaded source image and its regions (b), the first target image and its regions (c), and the second target image and its regions (d). Notice that the navigator icons are also available for changing

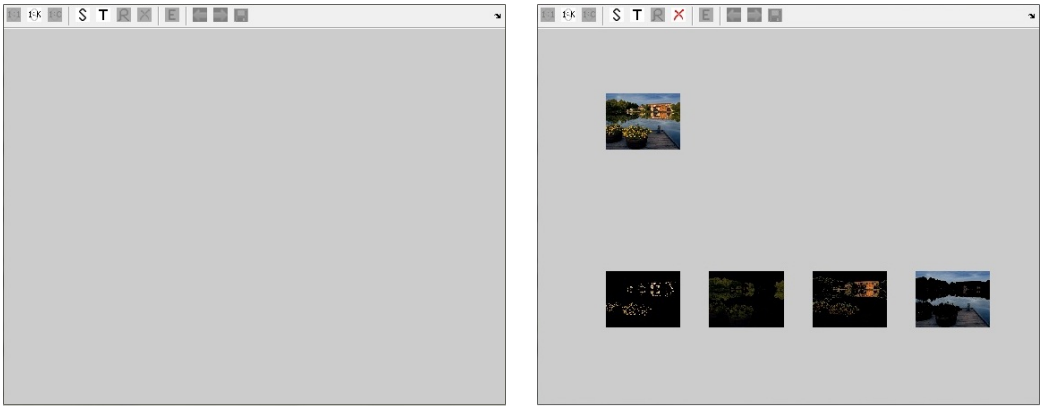


(a) (b)

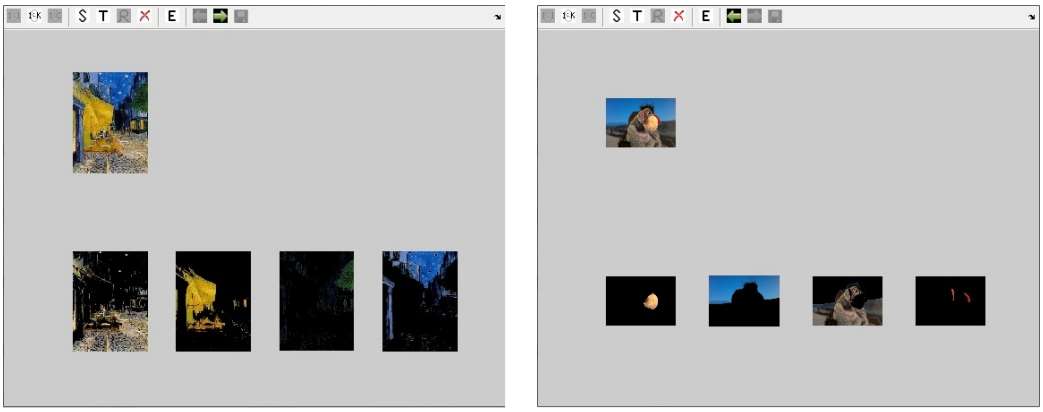
Figure A.7: User interface: save and show results

the current target image. A user can change the current target image any time while exploring. The change of the current target image also change the target regions associated with the source image.

Specific-source-target-region scenario As mentioned in Chapter 4, any image can specify its regions by a tag name using the region marking feature. This scenario expects specific regions or marked regions as inputs. The exploration process is still the same as a normal case, whole images as inputs. Figure A.9 shows the user interface at the beginning of this scenario (a), a source image and its marked regions (b), a target image and its marked regions (c), and the results while exploring the first source region (d). In this scenario, the marked regions of the source image are affected from color transfer. The other regions are always kept intact.

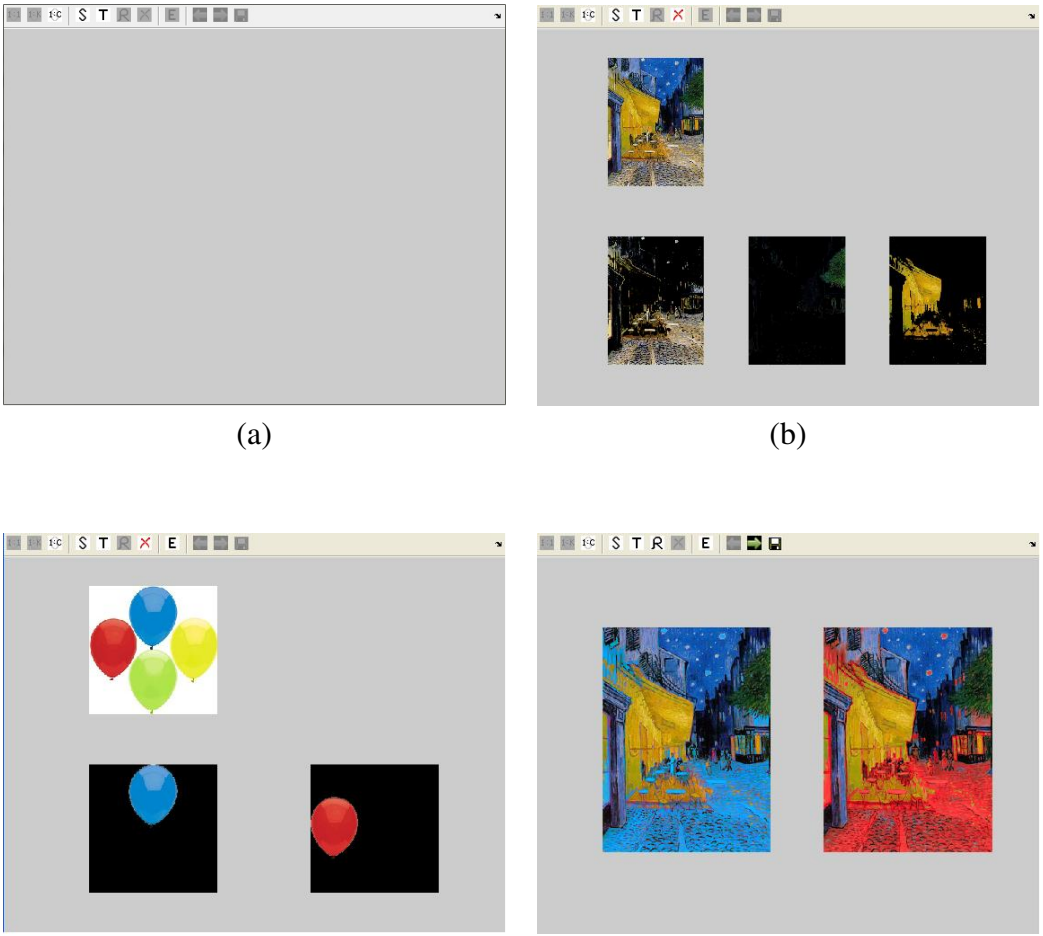


(a) (b)



(c) (d)

Figure A.8: User interface: one-source-many-target scenario



(a) (b)

(c) (d)

Figure A.9: User interface: specific-source-target-region scenario

BIOGRAPHY

NAME	Mr. Somchai Phatthanachuanchom
DATE OF BIRTH	15th April 1973
PLACE OF BIRTH	Tak, Thailand
INSTITUTIONS ATTENDED	Kasetsart University, 1991–1995 Bachelor of Electrical Engineering (Communication Engineering) Offenburg University, 2000–2002 Master of Communication and Media Engineering Mahidol University, 2003–2010 Doctor of Philosophy (Computer Science)
E-MAIL	chaikant@hotmail.com