Pigment-Based Recoloring of Watercolor Paintings

Elad Aharoni-Mack Yakov Shambik Dani Lischinski
The Hebrew University of Jerusalem

Figure 1: Pigment-based watercolor painting recoloring. Left: the original painting with an automatically extracted three-color pigment-based palette (below). Center, Right: two recolored paintings using the modified palettes shown below. Original watercolor painting © Olya Kamieshkova.

ABSTRACT

The color palette used by an artist when creating a painting is an important tool for expressing emotion, directing attention, and more. However, choosing a palette is an intricate task that requires considerable skill and experience. In this work, we introduce a new tool designed to allow artists to experiment with alternative color palettes for existing watercolor paintings. This could be useful for generating alternative renditions for an existing painting, or for aiding in the selection of a palette for a new painting, related to an existing one. Our tool first estimates the original pigment-based color palette used to create the painting, and then decomposes the painting into a collection of pigment channels, each corresponding to a single palette color. In both of these tasks, we employ a version of the Kubelka-Munk model, which predicts the reflectance of a given mixture of pigments. Each channel in the decomposition is a piecewise-smooth map that specifies the concentration of one of the colors in the palette across the image. Another estimated map specifies the total thickness of the pigments across the image. The mixture of these pigment channels, also according to the Kubelka-Munk model, reconstructs the original painting. The artist is then able to manipulate the individual palette colors, obtaining results by remixing the pigment channels at interactive rates.

NPAR'17, July 28-29, 2017, Los Angeles, CA, USA

ACM ISBN 978-1-4503-5081-5/17/07…$15.00
https://doi.org/10.1145/3092919.3092926

CCS CONCEPTS

• **Computing methodologies** → **Non-photorealistic rendering**; **Image manipulation**;

KEYWORDS

pigments, watercolor, decomposition, Kubelka-Munk, recoloring, color-transfer

ACM Reference format:
Elad Aharoni-Mack, Yakov Shambik, and Dani Lischinski. 2017. Pigment-Based Recoloring of Watercolor Paintings. In *Proceedings of NPAR'17, Los Angeles, CA, USA, July 28-29, 2017,* 11 pages.
https://doi.org/10.1145/3092919.3092926

1 INTRODUCTION

> "Getting to know as much as possible about pigments and their personalities is important to any artist, and even more so to a watercolorist."
>
> — Jeanne Dobie, *Making Color Sing*

Color is one of the most important elements available to artists in order to express emotion and to convey mood. Selecting a color palette is one of the very first steps in the process of creating a painting, and has a huge impact over the result. Different color palettes could result in a different atmosphere, tell a different story, change the focus of the painting, and more. Unfortunately, selecting the proper palette requires considerable skill and experience, and once the selection has been done and the painting has been created, the result is impossible to change.

In particular, in watercolor paintings, color arises from light reflected from the paper and through the layers of pigments. Clear and pure color pigments lead to transparent, light and vivid colors,

while more opaque pigments and non-pure paint additives lead to muddy results. MacKenzie [1999] provides a good overview of the nature and characters of the different pigments, and lays out a number of rules for artists to follow when selecting a palette:

Vivid (Saturated) Colors have a powerful impact, catch the attention and should be used for center of interest. Thus, there should be a good balance between dulled and vivid colors.

Temperature of color is a psychological interpretation of the warmth or coolness of color as perceived by a human observer. Warm hues, such as red and orange attract attention and may be associated with a feeling of action, as opposed to cool colors, such as green or blue.

Contrast of colors is relative to their environment. A color may be emphasized by placing it next to a color with opposite characteristics, such as hue, temperature, and/or purity.

Limited Palette is usually key to achieving a unified result. It is considered good practice to let colors appear at their purest form, while letting them mix during the painting process to create interesting transitions.

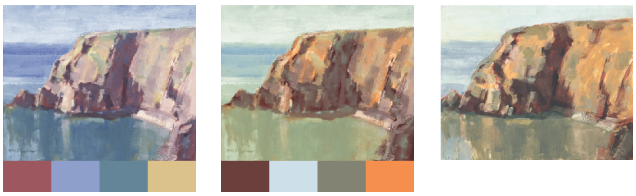
Color Harmony is an effect created by combining two or more colors in a palette. A variety of interesting harmonies may be achieved by using neighboring or contrasting colors.

As we can see from the above list, selecting the color palette is indeed a crucial, yet intricate task. In this work, we introduce a new tool designed to allow artists (and particularly watercolorists) to explore alternative color palettes for paintings that have already been created. This could be useful for creating a variety of different digital renditions for an existing painting (as shown in Figure 1), as well as for selecting a color palette for a new painting related to an existing one, as demonstrated in Figure 2.

A number of painterly rendering methods have been proposed over the past two decades [Gooch and Gooch 2001], some of which specifically target watercolor, e.g., [Curtis et al. 1997]. Most of these methods, however, are designed to digitally create painterly images from scratch, to render painterly images from 3D models, or to create such images by processing existing photographs and videos. Much less research has been done on re-editing and re-coloring existing paintings.

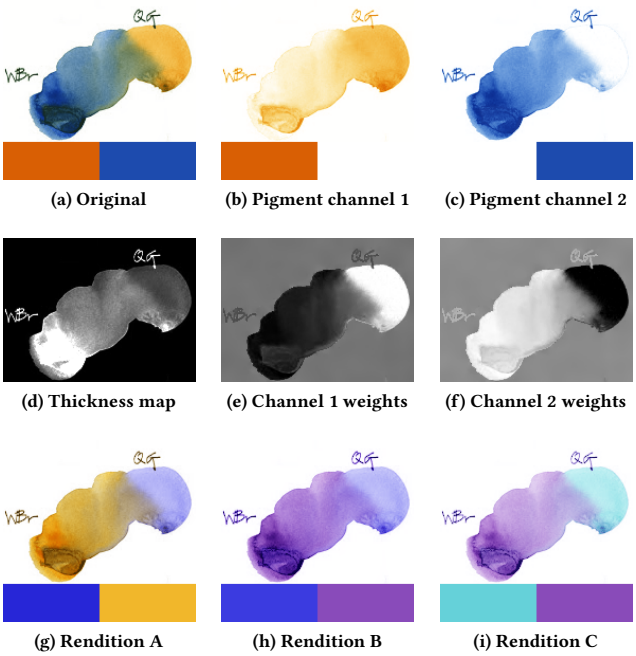
There are also many techniques for manipulating color in general images, including color transfer ([Reinhard et al. 2001] and many follow ups) and interactive colorization [Levin et al. 2004]. Techniques have also been proposed specifically for manipulating color schemes and palettes, e.g., [Chang et al. 2015; Shapira et al. 2009; Tan et al. 2016; Wang et al. 2010]. Although these tools may provide good solutions for editing general images, as we show in this work, they are less well suited for the task of color manipulation in paintings, and particularly watercolor paintings.

When modifying the color palette of a watercolor painting, care must be taken to ensure that the result still looks like a painting consisting of spatially coherent brush strokes and color washes created using colors from the palette. In this work, we achieve this by first estimating the pigment-based color palette that was used to create the painting, and then decomposing the painting into a collection of pigment channels, each corresponding to a single palette color. In both of these tasks, we employ a simplified, but



(a) Original (b) Recolored (c) Reference

Figure 2: The palette of the oil painting in (a) is modified, using our tool, to create the recolored result in (b). The new palette is designed using the advanced editing UI of our tool, so as to match the reference painting in (c). Original oil paintings © Michael Chesley Johnson; images used by permission of The Artist's Magazine, F+W Media.



(a) Original (b) Pigment channel 1 (c) Pigment channel 2

(d) Thickness map (e) Channel 1 weights (f) Channel 2 weights

(g) Rendition A (h) Rendition B (i) Rendition C

Figure 3: Decomposition and recomposition of a simple watercolor painting created with a bi-color palette. Top row: original painting and the two extracted pigment channels, rendered separately. Second row: the corresponding thickness map and mixture weights. Bottom row: alternate renditions with modified palettes. White colors typically decompose arbitrarily, uniformly and with thickness 0. Original watercolor painting © Liz Steel.

sufficiently powerful, version of the Kubelka-Munk model, which predicts the reflectance of a given mixture of pigments. Specifically, each of the palette colors is assumed to have been created by mixing together a set of base pigments, whose spectral parameters are known to us. Each pigment channel in the decomposition is a piecewise-smooth map that specifies the concentration of one of the colors in the palette across the image (see Figure 3). A thickness map is estimated as well, which specifies the total thickness of the

pigments across the image. The mixture of these pigment channels, also according to the Kubelka-Munk model, reconstructs the original painting. The artist is then able to manipulate the individual palette colors, and see the result created by remixing the pigment channels at interactive rates.

We compare our tool to a number of alternatives, and show its effectiveness for editing the color palette of a watercolor painting. Although we have not yet carried out a user study, we believe that the resulting tool is useful even for novice users with little or no experience in painting or knoweldge of pigments.

2 RELATED WORK

2.1 Digital Simulation of Artistic Media

Much of the research in the area of non-photorealitic rendering has focused on digitally simulating traditional artistic media, targeting oil and watercolor paintings, as well as pen-and-ink and pencil drawings [Gooch and Gooch 2001]. Models have been proposed for paints, brushes, and substrates, as well as algorithms for the artwork creation processes (drawing and painting).

Curtis et al. [1997] describe an elaborate system for computer generated watercolor. They discuss a variety of watercolor effects and introduce a physically-based model for simulating them. The distribution of pigments across the paper is simulated using the physics of shallow water fluid flows, while the resulting reflectances across the painting are predicted using the Kubelka-Munk (KM) model [Kubelka 1948]. Their work focuses on creating watercolor-like artwork from scratch, as well as "watercolorizing" existing images. In contrast, the goal of our work is to decompose a watercolor painting in a manner that would enable experimenting with its color palette. We also rely on the Kubelka-Munk model for modeling the reflectance of watercolor pigment mixtures, and describe the relevant equations in Section 3.

Baxter et al. [2004] describe a viscous paint model targeting interactive applications and painting in particular. Their work complements [Curtis et al. 1997] by providing a complete pipeline for viscous paints, such as acrylic and oil paints, as opposed to watercolor. Again, the focus here is on the interactive creation of paintings, and on applying the Kubelka-Munk model in real-time rendering. Painting decomposition is not addressed.

The Kubelka-Munk model has also been used in computer graphics for modeling pigmented materials [Haase and Meyer 1992] and metallic patinas [Dorsey and Hanrahan 1996].

2.2 Painting Decomposition

One of the use cases described in [Curtis et al. 1997] is an automatic image "watercolorization". Given an ordered set of pigments, they compute the corresponding layer thicknesses that would reproduce the image according to the Kubelka-Munk layer compositing model. The resulting thicknesses are then used to determine the brushstrokes in each layer. Since our work does not involve generating and simulating brushstrokes, we use the simpler, single layer, version of the Kubelka-Munk model, rather than their layer compositing model. However, we enforce piecwise smoothness when performing our decomposition, and thus the resulting channels appear similar to ones that might have been painted by an artist.

Tan et al. [2015] describe a full pipeline for decomposing a time-lapse video of a painting process into a set of translucent strokes or images applied at each step by utilizing time and space assumptions. They also use the Kubelka-Munk model in order to recover the layers corresponding to successive steps of the painting process. Our approach does not assume that the creation history is available, and attempts to compute a decomposition into pigments given only the final painting.

Another recent work by Tan et al. [2016] presents a model for decomposing digital paintings into layers via RGB-space geometry. Their method first recovers the palette of colors used in the painting based on convex-hull analysis and assuming alpha blending of colors. Next, based on a user's ordering of RGB colors, the painting is decomposed into layers using optimization with spatial constraints. This work targets digital paintings, which are indeed created using blending, in contrast to paintings created using real pigments. Furthermore, the spatial term they use penalizes edges in the resulting layers, in contrast to the edge-aware spatial term used in our approach. This allows the decomposition produced by our approach to better capture brush strokes. We compare our approach with alpha-based decomposition in Section 5.1.

2.3 Color Transfer and Recoloring

Over the years, there has been much work on the transfer of color from one image to another. The pioneering work of Reinhard et al. [2001] transfers colors by globally matching color statistics, an approach that was later improved by several follow-up works, e.g., [Pitié et al. 2007]. However, global transfer of color statistics may produce unexpected results and artifacts. HaCohen et al. [2011] utilize dense correspondences between images to derive a parametric color transformation, based on content shared by the two images. Yoo et al. [2013] find local region correspondences between two images by exploiting their dominant colors in order to apply a statistical transfer. All of the these methods require a reference image as input, in addition to the image being manipulated.

Various interactive techniques for recoloring were explored as well. A number of methods allow users to recolor images via a scribble-based interface, e.g., [An and Pellacini 2008; Chen et al. 2012; Levin et al. 2004; Qu et al. 2006; Xu et al. 2009]. These methods focus on how to properly propagate the edits indicated by a set of local scribbles to the entire image.

More closely related to this work is the palette-based recoloring approach of Chang et al. [2015], which extracts a color palette from an image using a variant of the k-means clustering algorithm. Given a target color palette, the new luminance L and chroma ab of each pixel are computed independently. The new chroma is obtained via a linear blend of transfer functions, using radial basis functions in the ab color plane. Although suitable for general images, we found their method to be less well suited for watercolor paintings. Since their approach uses linear blending of palette colors, rather than the Kubelka-Munk model, their palette colors do not mix in the same manner as real pigments. Consequently, regions in the image often reach saturation when the palette is manipulated, which is undesirable for painting recoloring. Also, their approach does not compute a decomposition into piecewise smooth channels, as we do,

which may result in noisy results when the palette is manipulated. We demonstrate these differences in a comparison in Section 5.2.

3 THE KUBELKA-MUNK PIGMENT MODEL

The Kubelka-Munk theory of reflectance is a commonly used model that predicts the reflectance of a homogeneous, isotropic pigment layer on top of a substrate, whose reflectance is known. It was first introduced by Kubelka and Munk in their work [1931] in order to predict the color of a painted substrate or predict a thickness of paint needed to obscure the substance. Later, Kubelka [1948] provided a simplified analysis of the interaction of incoming light with a layer of material such as a layer of paint under the following conditions: the material is assumed to be uniform, isotropic, non-fluorescent, non-glossy and the sample has to be illuminated by diffuse, monochromatic light.

The widely used and accepted method for estimating the reflectance R of a pigment layer, introduced by Kubelka and Munk [1948; 1931], is known as the Kubelka-Munk equation:

$$R = R_{KM}(K,S,\xi,h) = \frac{1-\xi(a-b\coth(bSh))}{a-\xi+b\coth(bSh)}, \quad (1)$$
$$\text{where} \quad a = 1+\frac{K}{S} \quad \text{and} \quad b=\sqrt{a^2-1}.$$

Here K and S are the pigment's absorbtion and scattering coefficients, respectively, ξ is the substrate reflectance, and h is the thickness of the pigment layer. All quantities are per-wavelength, therefore a pigment is modeled as pairs of absorbtion and scattering coefficients over a set of wavelengths. Thus, to determine the reflectance of a pigment layer over a substrate with some known reflectance in the RGB color space, this model requires a total of 7 parameters (6 absorption and scattering coefficients, as well as the layer thickness h).

In this work, we use the Kubelka-Munk equation for predicting the reflectance of a mixture of pigments, rather than a single pigment. In this case, the absorbtion and scattering coefficients of the mixed pigment layer can be obtained simply as the linear combination of the corresponding coefficients of the different pigments in the mixture [Duncan 1940; Glassner 1994]. We also use this property later, when optimizing pigments and palette colors mixture weights, in order to reproduce the RGB values observed in the image.

In this work, our goal is to allow users to easily specify pigment-based color palettes. Requiring users to directly specify the absorbtion and scattering coefficients of each pigment is less intuitive than asking them to perform a visual selection, e.g., using an RGB color picker. Curtis et al. [1997] request the user to specify the K and S coefficients interactively, by picking two RGB colors: R_w, specifying the appearance of a layer of the pigment over a white substrate, and R_b, over a black substrate. The pigment layer is assumed to be of unit thickness. Curtis et al. show that given R_w and R_b, the K and S coefficients can be obtained by inverting the Kubelka-Munk equations, using the following equations:

$$S = \frac{1}{b}\coth^{-1}\left(\frac{b^2-(a-R_w)(a-1)}{b(1-R_w)}\right) \quad (2)$$
$$K = S(a-1), \quad (3)$$

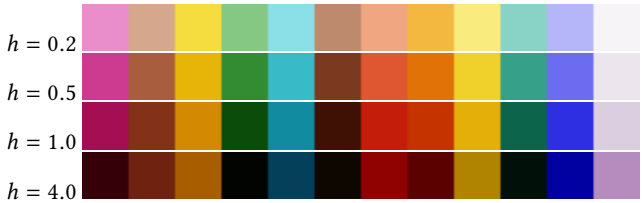


Figure 4: Base pigments taken from [Curtis et al. 1997] are rendered with different thicknesses $h = 0.2, 0.5, 1, 4$. These pigments are used to mix palette colors, which, in turn, are used to mix and generate the painting's colors.

where

$$a = \frac{1}{2}\left(R_w + \frac{R_b - R_w + 1}{R_b}\right), \quad b = \sqrt{a^2-1} \quad (4)$$

In the results shown in this paper, we use their set of 12 base pigments, and the above method may be used to add additional base pigments to this set. However, our user interface also allows the user to indicate a pigment via a single RGB color selection, in which case we approximate the selected RGB color by fitting to it a mixture of base pigments, as described later in Section 4.1.

4 METHOD

The goal of the proposed approach is to decompose a watercolor painting into several channels, each corresponding to a pigment from a palette, in a manner that would enable an artist user to experiment with different palettes, as demonstrated in Figure 1.

The first challenge is, thus, to estimate the color palette used in the original painting. This is done using an automatic method described in Section 4.2. The next challenge is to use the estimated palette colors in order to perform a decomposition of the painting into a set of spatially smooth pigment channels, and a thickness map, as described in Section 4.3. Figure 3 shows such a decomposition for a simple watercolor painting created using a bi-color palette.

Although we do not explicitly recover the original brush strokes used by the artist to create the painting, our decomposition attempts to capture these brush strokes by including an edge-aware spatial smoothness term into the optimization process described in Section 4.3. Once a decomposition into pigment channels is available, it is possible to modify the palette colors and recompose the channels to obtain an alternative rendition of the painting, as shown in the bottom row of Figure 3.

The palette pigment estimation, as well as the pigment channel decomposition and recomposition processes all use the well known and time tested Kubelka-Munk equations, as described in Section 4.1. This is key in order for the recomposed results to retain the appearance of a watercolor painting, and to remain faithful to the painting's original style, despite the change in palette.

In section 4.4 we combine the palette extraction and channel decomposition into a single optimization in order to improve decomposition and palette extraction accuracy. Finally, in Section 4.5, we describe our interactive UI for palette manipulation.

4.1 Pigment Mixture Model

When preparing a color palette for a painting, artists may use one of a set of pure base pigments in their possession, or a mixture of several of such base pigments. In this work, we use the Kubelka-Munk mixture model, described earlier in Section 3. We assume that the artist uses a set of 12 base pigments, shown in Figure 4, whose absorbtion and scattering parameters are taken from Curtis et al. [1997]. This is an arbitrarily chosen set of base pigments, and any other set of pigments with a sufficiently wide gamut may be used instead.

Thus, our approach assumes two levels of pigment mixtures. First, each color in the artist's palette is produced by a mixture of the 12 base pigments, and next each color in the painting is produced by a mixture of the palette colors. Below we describe how the mixture weights w are determined to obtain K, S, h given an RGB color.

Given a palette color c in the RGB color space, we compute the corresponding K and S values by looking for a set of weights w, such that a weighted linear mixture of our base pigments will reproduce the target color c by applying a layer with an arbitrary thickness over a substrate with reflectance $\xi = 1$. Specifically, we assume that the K and S coefficients are given by

$$K_w = \sum_{i=1}^{N} w_i K_i, \quad S_w = \sum_{i=1}^{N} w_i S_i, \tag{5}$$

where the K_i and S_i coefficients describe the base pigments ($N = 12$). Modeling a mixture of pigments by a linear combination of their spectral coefficients was experimentally validated by Duncan [1940].

Finding the mixture that matches a target color c is done by solving the following optimization problem:

$$\begin{aligned} \underset{w,t}{\operatorname{argmin}} \|R_{KM}(K_w, S_w, \xi = 1, h = t) - c\|, \\ \text{subject to } \sum_{i=1}^{N} w_i = 1, \end{aligned} \tag{6}$$

where t is the thickness of the mixture applied, and equation (5) is used to obtain the K_w and S_w values. We use L-BFGS-B [Nocedal 1980] in order to solve the optimization (as well as all subsequent optimizations in our method).

We denote the set of optimized spectral properties of the palette colors with K^*, S^* to distinguish them from the base pigments. For the second mixture level we denote by $L = \{L_1, \ldots, L_k\}$ the set of pigment channels and by T the thickness map used to produce the painting given the K^*, S^* values of the k palette colors. Each channel L_i is a mixture weight map for palette color i, and $(L)_p = \{w_1, \ldots, w_k\}$ is an operator indicating the mixture weights for all palette colors at pixel p. T_p indicates the total pigment thickness t applied at pixel p.

4.2 Extracting a Color Palette

Our method supports either an automatic or a manual process of palette extraction. We assume that for each color in the palette, the painting contains at least one region, where this color appears in its pure form, i.e., not mixed with any of the other colors in the palette. To identify these pure palette colors automatically, we perform convex hull optimization in the a^*b^* plane of the CIE $L^*a^*b^*$ color space. To avoid noise, we first discard the brightest and the darkest colors, and compute the convex hull of the remaining colors in the image. We then greedily iterate in order to simplify the convex hull polygon by pruning its vertices until we are left with k vertices, where k is the desired palette size. Pruning is done similarly to the Douglas-Peucker algorithm [1973], where at each iteration we remove the vertex whose distance from the line connecting its neighbors is the smallest. We use the original L value of the projected vertex in order to transform back to RGB color space.

Alternatively, the user can indicate one or more of the pure palette colors by clicking on the image, or by selecting a color using a color picker.

After identifying the RGB colors of the palette, we represent each palette color as a mixture of base pigments, as described in the previous section. Later, in Section 4.4 we will show how the palette extraction phase could be combined with the decomposition into pigment channels in order to increase accuracy.

4.3 Painting Decomposition

Given the K^*, S^* values for each of the k palette colors, we decompose each pixel into a set of k mixture weights and a thickness indicator scalar t. The result is a set of $k + 1$ scalars specifying mixture and applied thickness per pixel. Separating between thickness and mixture weights is helpful for maintaining local and global luminance monotonicity, while manipulating the spectral coefficients of the palette's pigments. This is similar to approaches such as [Chang et al. 2015], which treat luminance and chroma separately.

In order to enforce spatial smoothness of each of the pigment channels and exploit spatial cues, as in [An and Pellacini 2008; Chen et al. 2012; Wang et al. 2010], our method incorporates an edge-aware spatial term based on that of the WLS filter [Farbman et al. 2008]. Since our primary application is recoloring and color transfer, we enforce the smoothness only for the mixture weight maps and not for the thickness layer. The weight maps are thus obtained by optimizing

$$\underset{L,T}{\operatorname{argmin}} \left(E_{data} + \lambda E_{spatial} \right) \tag{7}$$

The data term E_{data} ensures that the colors of the pixels I_p may be reconstructed as a mixture of the pigments in the palette:

$$E_{data} = \sum_p \left(R_{KM}(K^*_{(L)_p}, S^*_{(L)_p}, \xi = 1, h = T_p) - I_p \right)^2 \tag{8}$$

For each pixel this is the same optimization as equation (6), except that here we use the K^*, S^* coefficients of the palette's colors, instead of those of the base pigments.

The spatial term enforces edge-aware smoothness of channels:

$$E_{spatial} = \sum_{i=1}^{k} \left(\sum_p \left(a_x(p) \left(\frac{\partial L_i}{\partial x}(p) \right)^2 + a_y(p) \left(\frac{\partial L_i}{\partial y}(p) \right)^2 \right) \right) \tag{9}$$

$$a_x(p) = \left(\left| \frac{\partial \ell}{\partial x}(p) \right|^{\alpha} + \epsilon \right)^{-1}, \quad a_y(p) = \left(\left| \frac{\partial \ell}{\partial y}(p) \right|^{\alpha} + \epsilon \right)^{-1},$$

where L_i is the i-th channel (mixture weights of the i-th palette color), and ℓ is the log-luminance of the input painting. α and λ

are used in a similar manner to WLS to control sensitivity to local edges and to balance between the data and smoothness terms.

Using the edge-aware spatial term above we are able to reduce in-channel noise and to produce spatially coherent decompositions, while capturing the significant chroma gradients. Thus, although we do not explicitly recover the individual brush strokes, the combined effect of the strokes done using a particular palette color is usually captured by the corresponding channel. The effect of the spatial term is demonstrated in Figures 5 and 6. Without spatial smoothing the decomposition is noisy, which may be visible after palette modification. Non edge-aware smoothing suppresses noise, but fails to separate the colors correctly, while edge-aware smoothing achieves both of these objectives.

Figure 5: Decomposition to pigments with and without edge-aware spatial smoothing. Each row shows the three pigment channels on the left and the thickness map on the right, for the painting in Figure 1. Top: no smoothing ($\lambda = 0$); Middle: using non edge-aware affinities ($\alpha = 0, \lambda = 0.5$); Bottom: using edge-aware affinities ($\alpha = 1.6, \lambda = 0.5$).

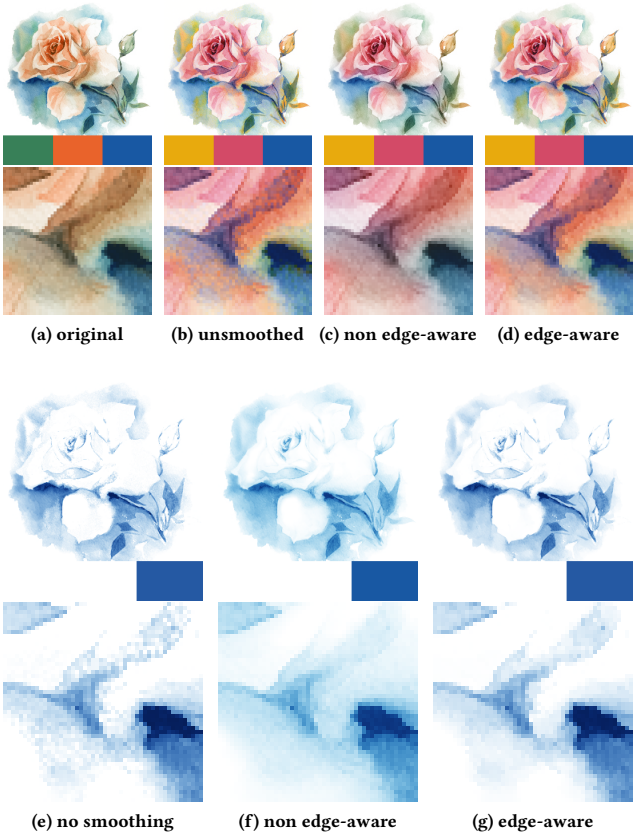


Figure 6: Effect of different smoothing methods on recoloring. Top row shows the combined result of three channels, while the bottom row shows only the blue pigment channel. Comparison includes: no smoothing, non edge-aware smoothing, and edge-aware smoothing. Without smoothing, the noise in the decomposition becomes visible after palette modification. Non edge-aware smoothing suppresses noise, but fails to separate the colors correctly, while edge-aware smoothing achieves both of these objectives.

4.4 Joint Palette and Channel Optimization

If the palette extraction process described in Section 4.2 succeeds in accurately estimating the palette's colors, the process ends with the channel decomposition described in the previous section. However, this might not be the case, for example, when the individual palette colors are never visible on their own in the painting. In this scenario, the recomposition of the painting from the pigment channels may fail to reproduce the original image.

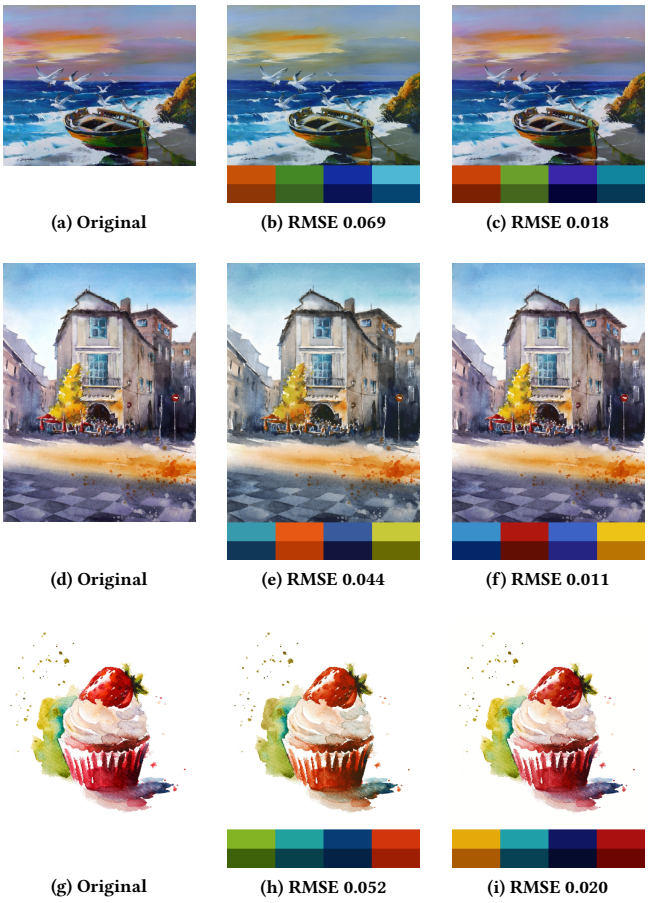
In order to achieve low decomposition error when minimizing equation (7), the derived spectral coefficients for the selected color palette must span the painting's gamut well in K, S domain. In cases where the painting's gamut is not well spanned either due to incorrect selection of the palette's RGB colors, or their conversion to the K, S parameters, we employ a joint optimization that adjusts the mixture of palette colors from base pigments in order to achieve a lower decomposition error. We feed the originally estimated spectral values as an initial guess and allow the optimization process to refine these values simultaneously with channel optimization.

Specifically, we modify equation (7) to:

$$\underset{L,T,W}{\operatorname{argmin}} \left(E_{data} + \lambda E_{spatial}\right)$$
$$s.t. \sum_{i=1}^{N} W_{i,j} = 1, \quad K_j^* = \sum_{i=1}^{N} W_{i,j} K_i, \quad S_j^* = \sum_{i=1}^{N} W_{i,j} S_i. \tag{10}$$

Figure 7 shows examples where the decomposition error is reduced and the visual accuracy is increased, when using the joint optimization above. In order to prevent the process from diverging, we start with the palette pigment mixtures fixed to their initial guess during the first c iterations ($c = 20$ in our current implementation), and then let the optimization modify the palette with a step size attenuated by 0.99^{i-c}, where i is the iteration number.

In practice, we use the joint optimization only if the decomposition of the painting with the palette computed as described in



(a) Original (b) RMSE 0.069 (c) RMSE 0.018

(d) Original (e) RMSE 0.044 (f) RMSE 0.011

(g) Original (h) RMSE 0.052 (i) RMSE 0.020

Figure 7: Automatic palette extraction with and without joint palette and channel optimization. In each row from left to right: original image, reconstructed image before palette refinement, reconstructed image after palette refinement. The reconstruction error is indicated under each image. In this and the following figures our pigment-based palettes are visualized using 2 thicknesses corresponding to x1 and x5 times the thickness estimated as described in Section 4.1. Oil painting in 1st row © Christian Jequel; Watercolor paintings in 2nd and 3rd rows © Olya Kamieshkova.

Section 4.2 fails to achieve a sufficientlyy low error ($RMSE < 0.3$). This is so, since the joint optimization process is significantly slower, by a factor of x5–x10 for an image of size 400×400, and since it does not guarantee that the selected paint in fact appears in the painting in its pure form. We also found that the joint optimization sometimes results in hue shifts (a change in hue as thickness increases) when using only a small number of palette colors. This is discussed further in Section 5.4.

4.5 Recomposition

Recompositing a recolored painting given a new palette of pigments is straightforward. The result of the decomposition process, either if done jointly or separately, are the base-pigments-to-palette mixture weights W, the per pixel palette-to-painting mixture weights L, and the per pixel applied thicknesses T. For each pixel p, we use equation (5) to calculate its K, S values and then its reflectance, using equation (1), by applying a layer with thickness T_p over a substrate assumed to have reflectance $\xi = 1$.

To provide an interactive experience we propose a user interface shown in Figure 8 that allows basic and advanced palette editing. Similar to other palette-based approaches, such as [Chang et al. 2015], our approach aspires for the same principles of simplicity, expressiveness, intuitivity and responsiveness.

The UI works as follows: the user specifies the number of colors in the palette and a palette is automatically extracted from the input painting. The user may then modify each of the palette's colors using several different ways, such as:

- For the source (decomposition) palette, by clicking on the original painting to indicate palette colors.
- For the target (recomposition) palette, by clicking on a reference painting to indicate palette colors. This mode is useful for color transfer between paintings (see Figure 2).
- A standard color picker dialog through which the user may indicate the reflectance of an arbitrary thickness layer over white background.
- By directly editing the spectral coefficients of each palette color (advanced mode).

Any change of pigment properties invokes a rendering operation for palette visualization. For advanced editing the UI provides 7 sliders for the currently selected palette color (see Figure 8). The first 6 are used to edit the K, S values, 2 per RGB channel. The 7th slider provides the ability to adjust the relative weight of the corresponding pigment channel. The target palette in Figure 2 was designed using this advanced editing UI.

5 RESULTS

We have implemented our method using Python and SciPy optimization toolbox with UI implemented using PyQt. Decomposition takes between 10 seconds and 1 minute, depending on the size of the palette, for an image of size 400×400 (on a single core 4.00GHz i7-4790K CPU), while recomposition takes approximately 1 second using an unoptimized CPU implementation, which already enables interactive palette exploration. A GPU implementation, as in [Baxter et al. 2004], would enable real-time response.

Figure 9 shows several color palette manipulation results obtained using our method.

We have also compared our method against several existing alternatives, for which we were able to obtain or reproduce their implementation. Specifically, we experimented with the alpha-compositing based decomposition of Tan et al. [2016], and with the palette-based recoloring of Chang et al. [2015]. We also experimented with the scribble-based interactive recoloring approach of Levin et al. [2004], and with the image appearance exploration tool of Shapira et al. [2009]. Although the latter tool is able to suggest a variety of alternative appearances for a given input image, we found that the resulting alternative appearances did not look like paintings produced using different palettes, and failed to obtain the results that we wanted in a controllable fashion.

Figure 8: Advanced editing UI. Left: original image with the decomposition palette. Right: recolored image with the modified palette. The row of color patches below each image shows the palette colors selected using an RGB color picker, or using our automatic palette extraction. When a color is selected, the 7 sliders below enable editing the 6 absorption and scattering coefficients and the relative weight of the channel. The two bottom rows of color patches visualize the pigment layer at two different thicknesses, after manipulating the coefficients. Visualization of two thicknesses allows better control and understanding. Watercolor painting © Olya Kamieshkova.

5.1 Comparison with α-based decomposition

As described in Section 2.2, Tan et al. [2016] perform convex hull simplification in 3D RGB color space to select a decomposition palette and decompose the image into layers that are composited using alpha blending. We found that the resulting decompositions typically do not resemble a decomposition into pigment layers, as demonstrated below.

Figure 10 shows two simple examples where two overlapping watercolor strokes are decomposed using our method and that of Tan et al. [2016] (using their implementation). These examples clearly demonstrate that our pigment-based decomposition is more faithful to the underlying stroke structure of the paintings. Our method is also able to more accurately reconstruct the original paintings than Tan et al. There are visible differences between the originals and the α-recomposed results (g,q), and the RMSE errors are higher for Tan et al. [2016] than they are for our method.

Figure 11 shows several comparisons of painting recoloring using our method vs. those of Chang et al. [2015] and Tan et al. [2016]. We discuss this comparison in more detail in the next section.

5.2 Comparison with palette-based recoloring

We also compare our method against the palette-based recoloring approach of Chang et al. [2015]. Compared to our method, theirs provides a fully interactive process, since their palette creation takes place in real-time. In order to ensure that the white background

Figure 9: Examples of pigment-based palette manipulation results, demonstrating the ability of our method to produce alternative renditions of watercolor paintings. In each row the original image (along with the extracted palette) is on the left, followed by two palette manipulation results. Watercolor paintings in 1st, 2nd and 3rd rows © Olya Kamieshkova; Watercolor painting in 4th row © Maria Stezhko.

remains white when using their method, we add white as an extra color to their palette.

We found that when applying the method of Chang et al. to paintings, their automatic palette rarely corresponds to the pigments present in the painting. Manipulating the palette entries with the intent to modify a particular color often introduces unexpected effects on other colors in the painting, making recoloring of painting less intuitive. Significant color changes often result in an image that no longer looks like a watercolor painting.

In Figure 11 we show several examples of painting recoloring using our method as well as those of Chang et al. [2015] and Tan et al. [2016]. In the top three rows the goal is to recolor an original painting (leftmost column) to a color scheme of a reference painting (rightmost color). Despite our best efforts, the results we were able to achieve with the other two methods are typically muddier and less vibrant than our results, and the colors often reach saturation,

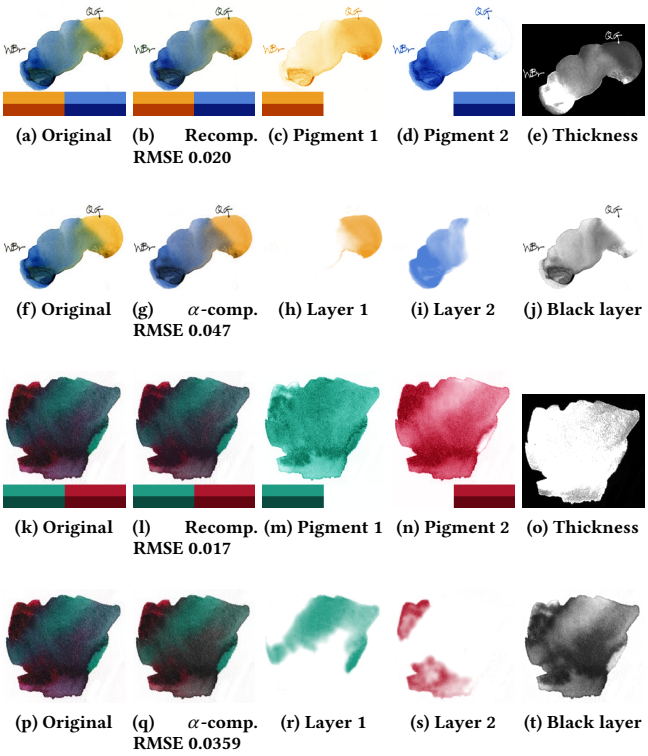


Figure 10: Comparison of pigment-based decomposition (our method, top and 3rd rows) vs. alpha-based decomposition [Tan et al. 2016] in 2nd and 4th rows. Our method uses two pigment channels and a thickness map, while alpha-based decomposition uses the same two-color decomposition palette along with a black layer. Watercolor paintings in 1st row © Liz Steel; Watercolor painting in 3rd row © Jane Blundell.

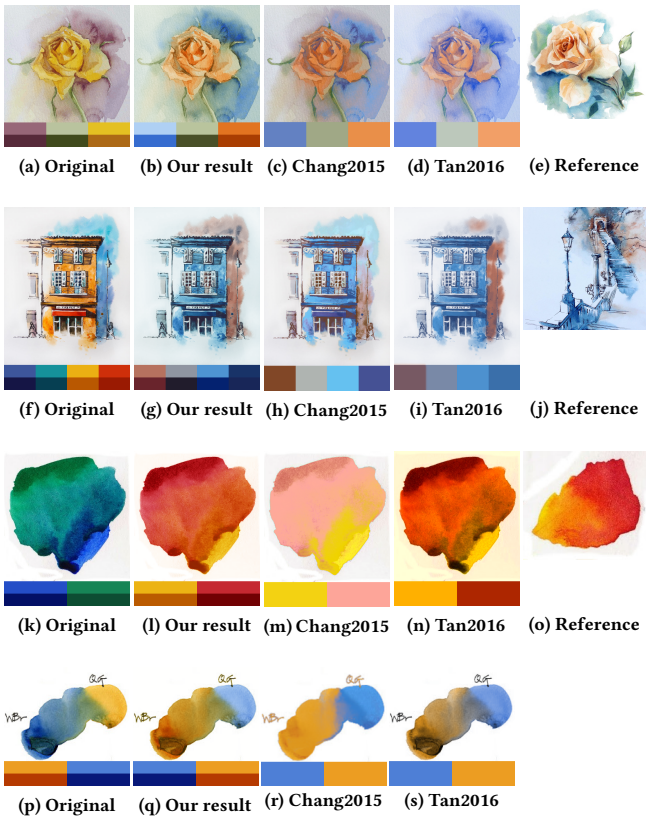


Figure 11: A comparison of palette manipulation results. From left to right, the columns correspond to: original painting, recolored using our method, recolored using Chang et al. [2015], recolored using Tan et al. [2016]. Tan's method does not use a palette; the palette shown under the Tan2016 results demonstrates the transformation that our palette colors undergo using their manipulation. Watercolor paintings in 1st and 2nd rows © Olya Kamieshkova; Watercolor painting in 3rd row © Jane Blundell; Watercolor painting in 4th row © Liz Steel.

thereby eliminating the delicate watercolor textures. In the bottom row, we simply swap the two palette entries of the original painting. Again, the colors, the transitions between them, and the textures are less well preserved with the methods of Chang et al. and Tan et al. Thus, we conclude that these two methods are not well-suited for recoloring watercolor paintings.

5.3 Comparison with edit propagation

Interactive editing methods such as [An and Pellacini 2008; Chen et al. 2012; Levin et al. 2004] propagate edits from a set of sparse scribbles to the entire image. We compare our approach to the method of Levin et al. [2004], since implementations of other methods are not available. A comparison is shown in Figure 12, where we show the scribbles that were used to recolor the painting. The diffusion of colors between the user's scribbles results in a different color transition than the one that may be observed in the original painting. This color transition is better captured by our method.

5.4 Limitations

The ability of a pigment-based palette to express dark colors is dependent on having a black palette color, or at least one pigment that becomes very dark (nearly black) as thickness increases; this requirement is similar to the alpha-blending based decomposition of Tan et al. [2016], which requires a black channel. Another non intuitive property of pigments (and the Kubelka-Munk model) is a change of hue as a function of thickness (hue shift).

These limitations are demonstrated in Figure 13. The top row shows that not having a dark color in the automaically extracted palette results in poor reconstruction and artifacts when the palette's pigments are darkened or lightened.

In the second row of Figure 13, a similar palette is used, but the green pigment was replaced with black. Although this palette yields a much more accurate reconstruction (f), attempts to make

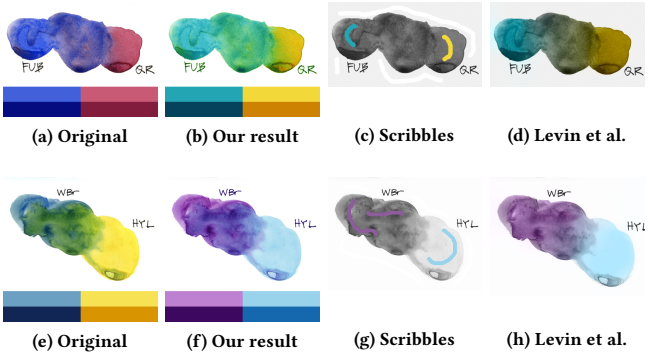


Figure 12: Comparison with Levin et al. [2004]. Watercolor painting © Liz Steel.

the painting lighter, either by manipulating only the non-black pigments (g), or the black pigment as well (h), result in hue changes, which may be undesirable.

The third row uses a palette obtained by the joint optimization of Section 4.4, which has two pigments that significantly darken with thickness (second and third pigments in the palette). The reconstruction is almost perfect visually (i). Replacing these pigments with ones that do not become sufficiently darker might result in failure to reproduce the contrast of the original painting, obtaining saturated non-black areas in regions that were near black in the original painting (j). Manipulating this palette can yield more satisfactory lightened (k) and recolored (l) versions of the original.

5.5 Photo Recoloring

Figure 14 and 15 show examples of pigment-based photo recoloring. Although we did not design our tool for photo recoloring, Figure 14 demonstrates that it can achieve satisfactory results (comparable to ones obtained using Chang et al.'s method). The manipulation was done using automatic palette extraction with 4 colors plus an added black color (a total of 5 colors). The target colors were obtained by clicking the target image to identify target colors, followed by fine tuning these colors via advanced editing of K and S values. For photo editing one might consider to modify the set of base pigments in order better to match real life colors.

Figure 15 shows an example where our method is more effective for recoloring a photo than Chang et al. [2015]. One of the colors in our palette successfully captures the color of the man's shirt, while another captures the color of the hat, allowing their manipulation without obvious artifacts. In contrast, we could not manipulate these colors using Chang's method without introducing artifacts.

6 CONCLUSION

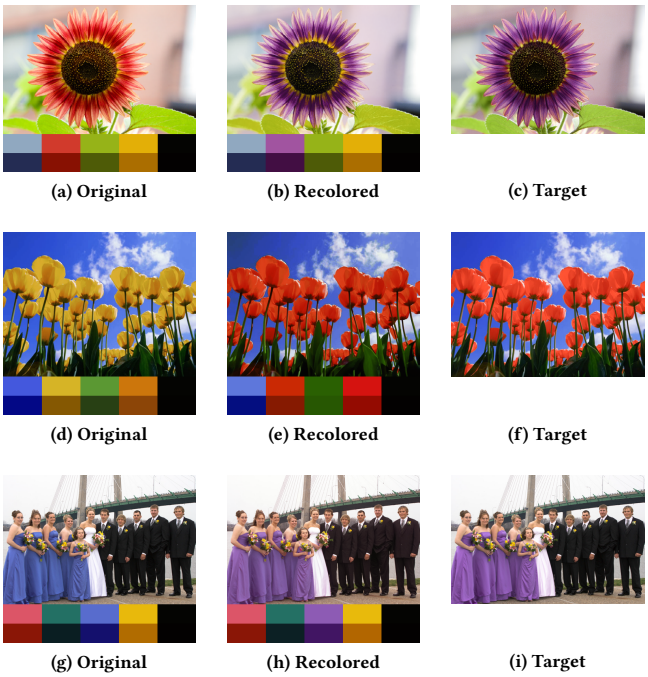
We introduced a new tool designed to allow artists to experiment with alternative color palettes for watercolor paintings. By using the Kubelka-Munk pigment theory along with edge-aware spatial smoothness constraints, we are able to decompose the painting into a set of pigment mixture weight channels, which enable the user to manipulate the color palette while remaining faithful to the original character of color transitions in the painting.

Figure 13: Top row: palette without sufficiently dark pigments suffers from poor reconstruction and unsatisfactory recoloring results. Middle row: replacing one of the pigments with black improves reconstruction, but can create undesirable hue shifts (g) and (h). Bottom row: a better palette with dark but non-black pigments can yield more satisfactory recoloring results (k) and (l). Watercolor painting © Misulbu Atelier.

Possible enhancements left for future work include adding support for the Kubleka-Munk layer composition model. This would increase the ability of our approach to handle effects such as watercolor glazing, and would enable users to modify not only the color palette, but also the ordering of the pigment layers at each pixel.

ACKNOWLEDGMENTS

We would like to thank the following artists for allowing us to use their paintings in our work: Liz Steel, Christian Jequel, Michael Chesley Johnson (and the Artist's Magazine), Maria Stezhko, Jane Blundell, Misulbu Atelier, with special thanks to Olya Kamieshkova. All artists hold the copyrights for their respective original works, which are used herein by their permission. This work was supported in part by the Israel Science Foundation (ISF).

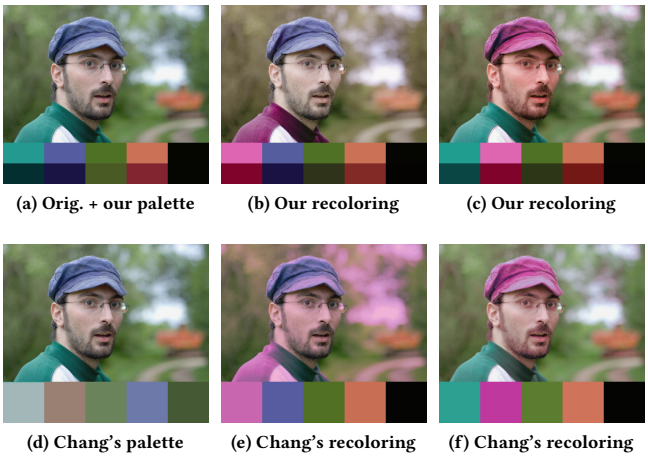


(a) Original (b) Recolored (c) Target

(d) Original (e) Recolored (f) Target

(g) Original (h) Recolored (i) Target

Figure 14: Examples of pigment-based photo recoloring using original and target photos from a user study conducted by Chang et al. [2015] and the recoloring results achieved using our method. All source palettes except for (a) were automatically extracted.



(a) Orig. + our palette (b) Our recoloring (c) Our recoloring

(d) Chang's palette (e) Chang's recoloring (f) Chang's recoloring

Figure 15: More pigment-based photo recoloring. Our automatically extracted color palette (a) enables artifact-free recoloring of the man's shirt and hat (b,c). We were not able to achieve such a recoloring with neither the automatic palette of Chang et al. (d), nor a manually selected one.

REFERENCES

Xiaobo An and Fabio Pellacini. 2008. AppProp: All-pairs Appearance-space Edit Propagation. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, Article 40 (2008), 9 pages. https://doi.org/10.1145/1399504.1360639

William Baxter, Yuanxin Liu, and Ming C. Lin. 2004. A Viscous Paint Model for Interactive Applications. *Computer Animation and Virtual Worlds* 15, 3-4 (July 2004), 433–441. https://doi.org/10.1002/cav.47

Huiwen Chang, Ohad Fried, Yiming Liu, Stephen DiVerdi, and Adam Finkelstein. 2015. Palette-based Photo Recoloring. *ACM Trans. Graph.* 34, 4, Article 139 (July 2015), 11 pages. https://doi.org/10.1145/2766978

Xiaowu Chen, Dongqing Zou, Qinping Zhao, and Ping Tan. 2012. Manifold Preserving Edit Propagation. *ACM Trans. Graph.* 31, 6, Article 132 (Nov. 2012), 7 pages. https://doi.org/10.1145/2366145.2366151

Cassidy J. Curtis, Sean E. Anderson, Joshua E. Seims, Kurt W. Fleischer, and David Salesin. 1997. Computer-Generated Watercolor. In *Proc. SIGGRAPH '97*. 421–430. https://doi.org/10.1145/258734.258896q

Julie Dorsey and Pat Hanrahan. 1996. Modeling and Rendering of Metallic Patinas. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '96)*. ACM, New York, NY, USA, 387–396. https://doi.org/10.1145/237170.237278

David H. Douglas and Thomas K. Peucker. 1973. Algorithms for the Reduction of the Number of Points Required to Represent a Digitized Line or Its Caricature. *Cartographica: The International Journal for Geographic Information and Geovisualization* 10, 2 (1973), 112–122. https://doi.org/10.3138/FM57-6770-U75U-7727 arXiv:http://dx.doi.org/10.3138/FM57-6770-U75U-7727

D R Duncan. 1940. The colour of pigment mixtures. *Proceedings of the Physical Society* 52, 3 (1940), 390. http://stacks.iop.org/0959-5309/52/i=3/a=310

Zeev Farbman, Raanan Fattal, Dani Lischinski, and Richard Szeliski. 2008. Edge-Preserving Decompositions for Multi-Scale Tone and Detail Manipulation. *ACM Transactions on Graphics (Proc. SIGGRAPH 2008)* 27, 3 (Aug. 2008).

Andrew S. Glassner. 1994. *Principles of Digital Image Synthesis.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Bruce Gooch and Amy Gooch. 2001. *Non-Photorealistic Rendering.* AK Peters.

Chet S. Haase and Gary W. Meyer. 1992. Modeling Pigmented Materials for Realistic Image Synthesis. *ACM Trans. Graph.* 11, 4 (Oct. 1992), 305–335. https://doi.org/10.1145/146443.146452

Yoav HaCohen, Eli Shechtman, Dan B. Goldman, and Dani Lischinski. 2011. Non-rigid Dense Correspondence with Applications for Image Enhancement. *ACM Trans. Graph.* 30, 4, Article 70 (July 2011), 10 pages. https://doi.org/10.1145/2010324.1964965

Paul Kubelka. 1948. New Contributions to the Optics of Intensely Light-Scattering Materials. Part I. *J. Opt. Soc. Am.* 38, 5 (May 1948), 448–457. https://doi.org/10.1364/JOSA.38.000448

Paul Kubelka and Franz Munk. 1931. An article on optics of paint layers. *J. Opt. Soc. Am.* 38, 5 (1931), 593–601.

Anat Levin, Dani Lischinski, and Yair Weiss. 2004. Colorization Using Optimization. *ACM Trans. Graph.* 23, 3 (Aug. 2004), 689–694. https://doi.org/10.1145/1015706.1015780

Gordon MacKenzie. 1999. *The Watercolorist's Essential Notebook.* North Light Books.

Jorge Nocedal. 1980. Updating quasi-Newton matrices with limited storage. *Mathematics of computation* 35, 151 (1980), 773–782. https://doi.org/10.1090/S0025-5718-1980-0572855-7

François Pitié, Anil C. Kokaram, and Rozenn Dahyot. 2007. Automated Colour Grading Using Colour Distribution Transfer. *Comput. Vis. Image Underst.* 107, 1-2 (July 2007), 123–137. https://doi.org/10.1016/j.cviu.2006.11.011

Yingge Qu, Tien-Tsin Wong, and Pheng-Ann Heng. 2006. Manga Colorization. *ACM Trans. Graph.* 25, 3 (July 2006), 1214–1220. https://doi.org/10.1145/1141911.1142017

Erik Reinhard, Michael Ashikhmin, Bruce Gooch, and Peter Shirley. 2001. Color Transfer Between Images. *IEEE Comput. Graph. Appl.* 21, 5 (Sept. 2001), 34–41. https://doi.org/10.1109/38.946629

Lior Shapira, Ariel Shamir, and Daniel Cohen-Or. 2009. Image Appearance Exploration by Model-Based Navigation. *Computer Graphics Forum* (2009). https://doi.org/10.1111/j.1467-8659.2009.01403.x

Jianchao Tan, Marek Dvorožňák, Daniel Sýkora, and Yotam Gingold. 2015. Decomposing Time-lapse Paintings into Layers. *ACM Trans. Graph.* 34, 4, Article 61 (July 2015), 10 pages. https://doi.org/10.1145/2766960

Jianchao Tan, Jyh-Ming Lien, and Yotam Gingold. 2016. Decomposing Images into Layers via RGB-Space Geometry. *ACM Trans. Graph.* 36, 1, Article 7 (Nov. 2016), 14 pages. https://doi.org/10.1145/2988229

Baoyuan Wang, Yizhou Yu, Tien-Tsin Wong, Chun Chen, and Ying-Qing Xu. 2010. Data-driven Image Color Theme Enhancement. *ACM Trans. Graph.* 29, 6, Article 146 (Dec. 2010), 10 pages. https://doi.org/10.1145/1882261.1866172

Kun Xu, Yong Li, Tao Ju, Shi-Min Hu, and Tian-Qiang Liu. 2009. Efficient Affinity-based Edit Propagation Using K-D Tree. *ACM Trans. Graph.* 28, 5, Article 118 (Dec. 2009), 6 pages. https://doi.org/10.1145/1618452.1618464

Jae-Doug Yoo, Min-Ki Park, Ji-Ho Cho, and Kwan H. Lee. 2013. Local color transfer between images using dominant colors. *Journal of Electronic Imaging* 22, 3 (2013), 033003–033003. https://doi.org/10.1117/1.JEI.22.3.033003