Optimization-based Gradient Mesh Color Transfer

Yi Xiao[1,2], Liang Wan[*,3], Chi Sing Leung[2], Yu-Kun Lai[4], Tien-Tsin Wong[5]

[1] Hunan University, Changsha, China, yixiao1984@gmail.com
[2] City University of Hong Kong, Hong Kong SAR, China, eeleungc@cityu.edu.hk
[3] Tianjin University, Tianjin, China, lwan@tju.edu.cn (*, corresponding author)
[4] Cardiff University, Cardiff CF10 3AT, U.K., yukun.lai@cs.cardiff.ac.uk
[5] The Chinese University of Hong Kong; CUHK Shenzhen Research Institute, China, ttwong@cse.cuhk.edu.hk

Abstract
In vector graphics, gradient meshes represent an image object by one or more regularly connected grids. Every grid point has attributes as the position, color, and gradients of these quantities specified. Editing the attributes of an existing gradient mesh (such as the color gradients) is not only nonintuitive but also time-consuming. To facilitate user-friendly color editing, we develop an optimization based color transfer method for gradient meshes. The key idea is built on the fact that we can approximate a color transfer operation on gradient meshes with a linear transfer function. In this paper, we formulate the approximation as an optimization problem, which aims to minimize the color distribution of the example image and the transferred gradient mesh. By adding proper constraints, i.e. image gradients, to the optimization problem, the details of the gradient meshes can be better preserved. With the linear transfer function, we are able to edit the colors and color gradients *of the mesh points automatically, while preserving the structure of the gradient mesh. The experimental results show that our method can generate pleasing recolored gradient meshes.*

Keywords: Gradient mesh, example-based color transfer, linear operator, optimization.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Picture/Image Generation—

1. Introduction

In vector graphics, the gradient mesh, offered by Adobe Illustrator and Corel Coreldraw, is a popular representation, which is suitable for representing multi-colored objects with smoothly varying colors. It is used to create photo-realistic vector art by many artists. Gradient meshes represent an image object by one or more regularly connected grids. Every grid point (also called a control point) has attributes including the position, color, and gradients of these quantities defined in the parametric domain. The image represented by gradient meshes is obtained by bicubic interpolation of the specified grid information.

To create gradient meshes, artists have to manually specify mesh grids and manipulate the associated attributes, which is labor intensive. The problem has gained attention such that several methods for (semi-)automatic generation of gradient meshes [SLWS07] [LHM09] have been developed. Good results can be obtained with the aid of some user assistance. Although gradient meshes can be obtained automatically, how to *efficiently edit* gradient meshes remains problematic. Due to the large number of attributes associated with each control point, manually editing the attributes (such as the color gradients) of an *existing gradient mesh*, which can be obtained by manual operation or automatic generation, is not only nonintuitive but also time-consuming. When the gradient mesh contains a large number of mesh points, or the vector art consists of multiple gradient meshes, it becomes especially important to have more convenient and efficient ways for gradient mesh editing.

A straightforward way to relieve the color editing problem is to borrow the color editing methods for raster images, such as color transfer [RAGS01]. However, these methods cannot directly deal with gradient meshes, which have more attributes per element than raster images. People may consider performing color transfer on the rasterized gradient mesh and then regenerating the gradient meshes from the recolored rasterized gradient mesh using the methods in [SLWS07] [LHM09]. Nevertheless, it should be pointed out that using the generation method in [SLWS07] or [LHM09] is an approximating process, which will result in loss of detail information in the recolored image. While consuming extra computing time, the regeneration may also change the gradient mesh structure, which is important in some scenarios where the structure was originally obtained through labor-intensive manual creation or complicated optimization.

Considering these issues, in our previous work [XWL*13], we proposed a principal component analysis (PCA) based linear color transfer method, which changes the color style of a gradient mesh by borrowing the color statistics of an example image. This method investigates a nice property of a gradient mesh that applying a linear operator on the color component is equivalent to applying the linear operator on the colors and color gradients of the control points, respectively. The linear operator is defined as a fused version of two basic PCA based color transfer algorithms [XM06, AK07]. We call this method PCA fusion [XWL*13]. Although this method can obtain pleasing results in most cases, there still exist some disadvantages that can be further improved. The PCA fusion method implicitly assumes that the colors of the example image and the gradient mesh are Gaussian distributed, and only matches their means and variances in the de-correlated color space. Unfortunately, since the mean and variance are global properties, matching only them may result in out-of-gamut artifacts, which means some colors in the recolored gradient mesh may not be present in the example image. Moreover, the PCA fusion method does not consider preserving the detail information represented in the gradient mesh such as highlights and textures, which may be destroyed after color transfer. Finally, the PCA fusion method may incur color inconsistency artifacts because of the fusion operation.

In this paper, we propose a novel framework for gradient mesh color transfer to edit gradient meshes. The key insight of our framework is to approximate a color transfer operator with an *optimal* linear transfer function. The approximation can be well formulated as a classical minimization problem, which equivalently minimizes the differences between the color distribution of the linear transferred gradient mesh and that of an example image. Although our current method also uses a linear transform, we use a totally different framework. In the previous work, the linear transform is defined by PCA, which implicitly assumes the color distributions of the example image and the gradient mesh are Gaussian. In our current work, the transform is defined by directly minimizing the difference of color distributions, which does not have such an assumption. Therefore, our current method handles images with non-Gaussian color distributions, and also better matches the colors, so that the out-of-gamut colors are avoided. Moreover, in our current method, image gradients can be explicitly added to the minimization model as a constraint, which can control the influence of the transfer and preserve the details of the gradient mesh. Also, since the linear transform is solved directly from the minimization problem, artifacts due to the fusion operation in the previous work are avoided. Experimental results and a user study show that our method avoids abnormal colors caused by the unmatched distributions, and also better preserves the details of the gradient meshes. Figure 1 shows an example of the results obtained by our method.

2. Related Work

Color transfer, first introduced by Reinhard et al. [RAGS01], refers to a category of methods which modify the color distribution of a target image according to that of a reference image. Since its introduction, a large number of methods have been developed, and a recent survey can be found in [FPC*14]. Reinhard et al. [RAGS01] modeled the color distribution of an image using the mean and standard deviation of color values in $l\alpha\beta$ space, due to the color correlation in RGB space. The mean and standard deviation of both the target image and the reference image are then used for shifting and scaling the colors of the pixels in the target image. This method was then extended to produce an image sequence based on multiple reference images and user input parameters [WH04]. Abadpour and Kasaei [AK07] de-correlated the colors in RGB space using principal component analysis (PCA) and then matched the means and standard deviations in the de-correlated space. The color distribution is transferred to the target image by applying PCA-based transformation on each pixel. Similar ideas were also reported in [XM06] [Kot05].

Instead of matching the mean and standard deviation, some global color mapping techniques directly match the histograms [WSM99] [MS03] [GD05a] [NN05]. Recently, Pouli and Reinhard [PR11] presented a histogram reshaping technique for images of arbitrary dynamic range. In these histogram matching methods, the colors are transformed to a color space which is assumed to be uncorrelated, such as CIELab. The histogram of each color channel is then matched separately. Alternatively, Pitie et al. [PKD07] cast a three-dimensional color distribution matching problem into a sequence of one dimensional matching problems via the Radon transform. This method is suitable for any color spaces. The impact of color spaces on color transfer effects was recently exploited by Reinhard and Pouli [RP11].

Greenfield and House [GH03] performed image segmentation, and extracted a color palette by choosing representative colors from the segments. The color mapping between the palettes of the reference and target images is then computed. Rather than binary segmentation, Tai et

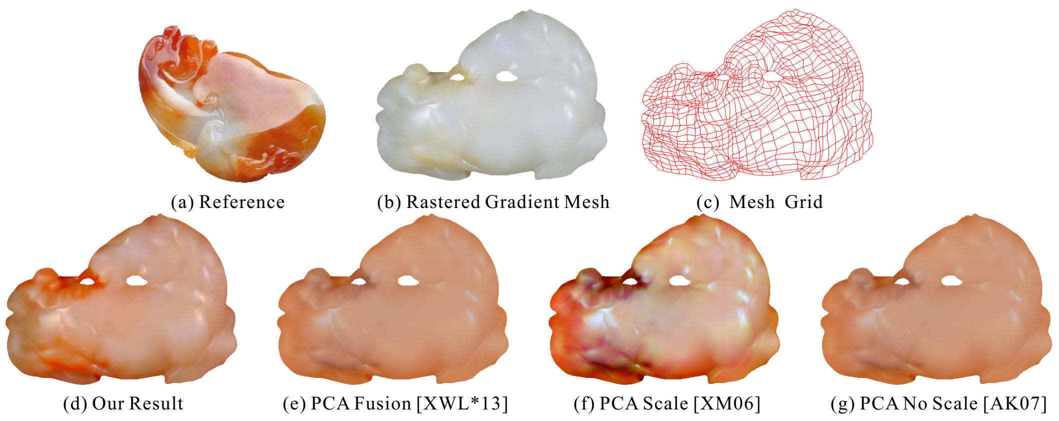


Figure 1: *Results of the jade example. Our method faithfully captures the color style of the example image and better preserves the details of the gradient meshes.* λ *is set to 1 in this example.*

al. [TJT05] applied probabilistic segmentation that produces soft region boundaries. Chang et al. [CSN03] categorized each pixel as one of the eleven basic perceptual color categories. Then, color transformation was applied within the same basic color category. This method has been extended to video data [CSN07]. Dong et al. [DBZP10] defined a one-to-one mapping between two dominant color sets, which are extracted from both the target image and reference image using probabilistic segmentation. The mapped dominant colors are then used for transforming the pixels in the target image. Wu et al. [WDM*11] adopted this idea and they further proposed to preserve the spatial distribution of the dominant colors. Su et al. [SDYL12] decomposed the target image into the base and detail layers. The colors of base layers are mapped by matching the distribution of base layers and that of a reference image. The final result is produced by combining the mapped base layers with the detail layers.

To avoid the spatially inconsistent artifacts caused by some per pixel operations like histogram matching, Pitie et al. [PKD07] suggested recovering the details of the target image after the per pixel operation. This is done by preserving the original gradients in the target image. Similar ideas can also be found in [LWX07,WHCO08,XM09,DBZP10], although expressed in different formulations. Instead of preserving gradient, Grundland et al. proposed mapping reference color gradients to corresponding target color gradients [GD05b]. In our new method, by comparison, we use the gradient term as a constraint in the optimization process.

The aforementioned methods can produce pleasing results when the color distributions of the reference and target images are homogeneous. Otherwise, they can generate unnatural looking results. To address this problem, some researchers suggested relying on user input swatches pairs to classify the colors [RAGS01,WAM02,XM06,AK07,AP10]. The color transfer is then locally performed within each swatch pair. The final color is the weighted sum of the results from all swatch pairs. In our method, we adopt this idea. We define a linear operator for each pair of swatches, and combine the linear transformed colors and color derivatives of all pairs.

Inspired by Reinhard et al.'s work [RAGS01], Welsh et al. [WAM02] colorized a grayscale image by borrowing the color characteristics of a color reference image. Instead of using a reference image, Levin et al. [LLW04] allowed users to specify the reference color distributions by drawing color strokes over a grayscale image. They then proposed a global optimization method to diffuse color strokes across the entire grayscale image. There have been many attempts to improve the computational efficiency of this process [HK05] [HTC*05] [ICOL05] [YS06] [LWCO*07]. In addition, automatic grayscale image colorization methods have been developed [JLWT04] [CHS08] [LHZ08] [MTN09].

3. Gradient Mesh

We now describe the mathematical representation of gradient meshes and discuss its linear property.

3.1. Gradient Mesh Representation

As defined in [SLWS07] [LHM09], a gradient mesh is a topologically planar rectangular grid, whose every four nearby control points (vertices) define a primitive component called a Ferguson patch [Fer64]. Figure 2 shows a gradient mesh with four Ferguson patches. For each control point in the grid, three types of information are specified: position $\mathbf{p} = (x, y)$, color $\mathbf{c} = (r, g, b)$ and their gradients $\mathbf{p}_u, \mathbf{p}_v, \mathbf{c}_u, \mathbf{c}_v$. Note that the gradients are defined in parametric coordinate space u, v, where $0 \leq u, v \leq 1$. Each Ferguson patch in the gradient mesh is rendered via bicubic Hermite interpolation, given by

$$m(u,v) = \mathbf{u}CQC^T\mathbf{v}^T, \tag{1}$$

where

$$Q = \begin{bmatrix} m^0 & m^2 & m_v^0 & m_v^2 \\ m^1 & m^3 & m_v^1 & m_v^3 \\ m_u^0 & m_u^2 & 0 & 0 \\ m_u^1 & m_u^3 & 0 & 0 \end{bmatrix},$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -3 & 3 & -2 & -1 \\ 2 & -2 & 1 & 1 \end{bmatrix},$$

m can be any of x, y, r, g, b, the superscript of m indicates one of the four control points of the Ferguson patch, $\mathbf{u} = [1\ u\ u^2\ u^3]$, $\mathbf{v} = [1\ v\ v^2\ v^3]$, and the parameters have the range of $0 \leq u, v \leq 1$. Given a parameter pair (u, v), its position and color are calculated using the above equation.

Since the position $\mathbf{p} = (x, y)$ and the color component $\mathbf{c} = (r, g, b)$ are given in parametric space, (1) implicitly defines a function $\mathbf{c} = \mathbf{g}(x, y, \mathbf{p^i}, \mathbf{p_u^i}, \mathbf{p_v^i}, \mathbf{c^i}, \mathbf{c_u^i}, \mathbf{c_v^i}, i \in \Gamma)$, where i is the index of a control point, Γ is the set which contains all the indices of the control points. Since $\mathbf{p^i}, \mathbf{p_u^i}, \mathbf{p_v^i}, \mathbf{c^i}, \mathbf{c_u^i}, \mathbf{c_v^i}, i \in \Gamma$ are constant, $\mathbf{g}(\cdot)$ implicitly defines the mapping between position (x, y) and color (r, g, b). Due to the complexity of (1), $\mathbf{g}(\cdot)$ can not be expressed explicitly. To get a raster gradient mesh, we sample in the parametric domain (u, v) to get a raster image of the gradient mesh. This sampling process is called rasterization. Denote the rasterized gradient mesh as $\mathbf{c} = \mathbf{s}(x, y)$, the process of rasterization is denoted by $Ras(\cdot)$. Then, we have $\mathbf{s}(x, y) = Ras(\mathbf{g}(x, y, \mathbf{p^i}, \mathbf{p_u^i}, \mathbf{p_v^i}, \mathbf{c^i}, \mathbf{c_u^i}, \mathbf{c_v^i}, i \in \Gamma))$.

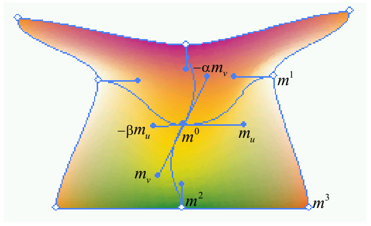


Figure 2: *The gradient mesh contains four Ferguson patches. The gradient information is illustrated for point m^0 in the bottom-right patch.*

3.2. The Linear Property of Gradient Mesh

As pointed out in [XWL*13], gradient meshes are defined in a parametric domain and have curvilinear grid structures. Since the grid structures should be preserved during the transfer, we aim to perform color editing in the parametric domain. In other words, we try to tackle colors and color gradients of control points.

In [XWL*13], Xiao et al. investigated the linear property of the gradient mesh. They found that applying a linear operator on the color component is equivalent to applying the linear operator on the colors and color gradients of the control points. We summarize the linear property of the gradient mesh in matrix form. When applying a linear operator $L(\cdot)$ represented by a 3×3 matrix T and a 3×1 translation vector $\mathbf{b}$ on $\mathbf{m}(u, v)$, we get

$$T\mathbf{m}(u,v) + \mathbf{b} \rightarrow \begin{cases} T\mathbf{m}^i + \mathbf{b}, \text{ for } i = 0,1,2,3, \\ T\mathbf{m}_u^i, \text{ for } i = 0,1,2,3, \\ T\mathbf{m}_v^i, \text{ for } i = 0,1,2,3, \end{cases} \tag{2}$$

where $\mathbf{m}(u,v) = [m_r(u,v), m_g(u,v), m_b(u,v)]^T$ is the color vector at the parametric coordinate (u, v), $\mathbf{m}^i = [m_r^i, m_g^i, m_b^i]^T$ is the color vector of the i-th control point, and $\mathbf{m}_u^i$, $\mathbf{m}_v^i$ are the color derivatives of the i-th control point. From the above equation, we can see that a linear operator is transparent to parametric interpolation. That is, if we want to linearly transform the colors inside a patch, we just need to perform the linear transformation on the colors and color gradients of the four control points, and re-rasterize the gradient mesh. The linear property can also be applied to the rasterized gradient mesh, which is given by

$$\begin{aligned} L(\mathbf{s}(x,y)) &= L(Ras(\mathbf{g}(x, y, \mathbf{p^i}, \mathbf{p_u^i}, \mathbf{p_v^i}, \mathbf{c^i}, \mathbf{c_u^i}, \mathbf{c_v^i}, i \in \Gamma))) \\ &= Ras(\mathbf{g}(x, y, \mathbf{p^i}, \mathbf{p_u^i}, \mathbf{p_v^i}, L(\mathbf{c^i}), T\mathbf{c_u^i}, T\mathbf{c_v^i}, i \in \Gamma)), \end{aligned} \tag{3}$$

where $L(\mathbf{c}) = T\mathbf{c} + \mathbf{b}$. The advantage of this property is two-fold. First, a linear operator can be defined using the rasterized gradient mesh, whose statistical and other properties can be more easily calculated. Second, the computational cost of the linear transformation only depends on the gradient mesh size, which is much smaller than the resolution of the rasterized gradient mesh in general. Note that the position information $\mathbf{p^i}, \mathbf{p_u^i}, \mathbf{p_v^i}$ are preserved, we ignore them in the following expressions. Meanwhile, we use $\mathbf{g}(x, y, L(\cdot))$ to denote $\mathbf{g}(x, y, L(\mathbf{c^i}), T\mathbf{c_u^i}, T\mathbf{c_v^i}, i \in \Gamma)$ to simplify the expression.

4. Optimization based Color Transfer

4.1. Single-Swatch Color Transfer

The purpose of color transfer is to edit the gradient mesh so that the edited gradient mesh has a similar color distribution to an example image, while preserveing the details of the gradient mesh. Therefore, we need to measure the color distribution function and the details of the gradient mesh. Since the color distribution function can not be analytically derived, we use the cumulative distribution function of the rasterized image obtained from the gradient mesh as in [XWL*13]. The details of the gradient mesh are measured by the image gradients of the rasterized image as in [PKD07, XM09]. We first consider a simple case in which the gradient mesh is to borrow the color distribution from the whole image. Considering the two purposes of color transfer, we formulate the color transfer process as a minimization model including two terms, one to measure the difference of the color distribution functions, and the other to measure the modification in image gradients. The minimization model is given by

$$E = \arg\min_{L} E_1 + \lambda E_2, \tag{4}$$

where

$$\begin{aligned} E_1 &= \int_\Omega \|cdf^{Ras(\mathbf{g}(x,y,L(\cdot)))}(\mathbf{c}) - cdf^{\mathbf{o}(x,y)}(\mathbf{c})\|^2 d\mathbf{c}, \\ E_2 &= \int_\Phi \|\nabla(Ras(\mathbf{g}(x,y,L(\cdot)))) - \nabla(\mathbf{s}(x,y))\|^2 dxdy. \end{aligned} \tag{5}$$

Here, $cdf(\cdot)$ denotes the cumulative distribution function; $\mathbf{o}(x,y)$ denotes the example image; Ω denotes the domain of color space; ∇ denotes the gradient operator; Φ denotes the domain of pixel space; and λ is a parameter to balance the influence of the two terms. The impact of λ is discussed in Section 5.1. Obviously, E_1 denotes the difference between the color distributions of the transferred gradient mesh and the example image, and E_2 is the gradient constraint to preserve the details of the gradient mesh. Using the linear property in (3), E_1 and E_2 become

$$\begin{aligned} E_1 &= \int_\Omega \|cdf^{L(\mathbf{s}(x,y))}(\mathbf{c}) - cdf^{\mathbf{o}(x,y)}(\mathbf{c})\|^2 d\mathbf{c}, \\ E_2 &= \int_\Phi \|\nabla(L(\mathbf{s}(x,y))) - \nabla(\mathbf{s}(x,y))\|^2 dxdy. \end{aligned} \tag{6}$$

(6) means that we can faithfully formulate $L(\cdot)$ on the rasterized gradient mesh without rasterizing the edited gradient mesh multiple times.

Since the $cdf(\cdot)$ functions in (6) are not differentiable and the calculation of the difference of the three dimensional cdf's is computational intensive, previous methods often rely on random search to solve the problem [AP10], which is time-consuming. In fact, (6) can be efficiently approximated by first matching the color distribution of the reference image $\mathbf{o}(x,y)$ and rasterized gradient mesh $\mathbf{s}(x,y)$, and then minimizing the difference of the linearly transferred colors and the matched colors. Since the distribution function of colors is 3-dimensional, we adopt the N-dimensional probability distribution transfer method in [PKD07], which fully transfers the statistics of the reference image to the target images even in RGB space. After the distribution transfer, we obtain a set of reference colors $\mathbf{s}^{\mathbf{o}}(x,y)$, which almost have the same color distribution as the example image $\mathbf{o}(x,y)$. Thereafter, E_1 can be approximated by

$$E_1' = \int_\Phi \|L(\mathbf{s}(x,y)) - \mathbf{s}^{\mathbf{o}}(x,y)\|^2 dxdy. \tag{7}$$

To summarize, our final energy function is given by

$$E' = \arg\min_{L} E_1' + \lambda E_2. \tag{8}$$

In (8), the functions of the unknown variables are linear in both the first term and second term. Therefore, T and $\mathbf{b}$ can be efficiently calculated from (8) by solving a linear system with only 12 variables. The detailed steps to solve T and $\mathbf{b}$ are given in the supplemental materials. After solving T and $\mathbf{b}$, we then transform the colors and color gradients of the gradient mesh control points based on (2). The transformed color vectors are given by

$$\mathbf{c}^l = T\mathbf{c}^t + \mathbf{b}, \tag{9}$$

where $\mathbf{c}^l$ denotes the transformed color vector of a control point, $\mathbf{c}^t$ denotes the color vector of a control point in the target gradient mesh. The transformed color gradient vectors are given by

$$\begin{aligned} \mathbf{c}_u^l &= T\mathbf{c}_u^t, \\ \mathbf{c}_v^l &= T\mathbf{c}_v^t. \end{aligned} \tag{10}$$

where $\mathbf{c}_u^l$ and $\mathbf{c}_v^l$ denote the transformed color gradient vector of a control point, $\mathbf{c}_u^t$ and $\mathbf{c}_v^t$ denote the color gradient vector of a control point in the target gradient mesh.

Figure 3 shows an example in which the color statistics of the image in Figure 3(a) are transferred to the gradient mesh in Figure 3(b)(c). As a comparison, two PCA methods [XM06, AK07] which can be extended to gradient meshes are also used. As shown in Figure 3(d), our method faithfully captures the color style of the example image, and also avoids abnormal colors, such as the purple colors in Figures 3(e)(f)(g), which are not present in the reference image 3(a). Moreover, our method better preserves the details of the original gradient mesh compared to the previous method [XWL*13](see the highlights in the body and tail).

4.2. Multi-Swatch Color Transfer

When the single swatch method does not work well, we may choose separate color swatch pairs and transfer desired color effects between swatches [RAGS01]. This scheme can also provide more flexible user control of color appearance. Our linear framework can be easily extended to multi-swatch color transfer.

For the i-th pair of swatches, we can obtain the transform matrix T_i and $\mathbf{b}_i$ using the single-swatch color transfer method in Section 4.1. Then the recolored vectors will be a weighted sum of single-swatch recolored vectors, given by

$$\begin{aligned} \mathbf{c}^l &= \sum_{i=1}^{M} w_i(T_i\mathbf{c}^t + \mathbf{b}_i), \\ \mathbf{c}_u^l &= \sum_{i=1}^{M} w_i T_i \mathbf{c}_u^t, \\ \mathbf{c}_v^l &= \sum_{i=1}^{M} w_i T_i \mathbf{c}_v^t. \end{aligned} \tag{11}$$

where M is the number of swatch pairs. The problem left is how to calculate the weighting factor w_i's. Since weighting factors based on Mahalanobis distance have been shown to result in visually pleasing results [XWL*13], we choose to

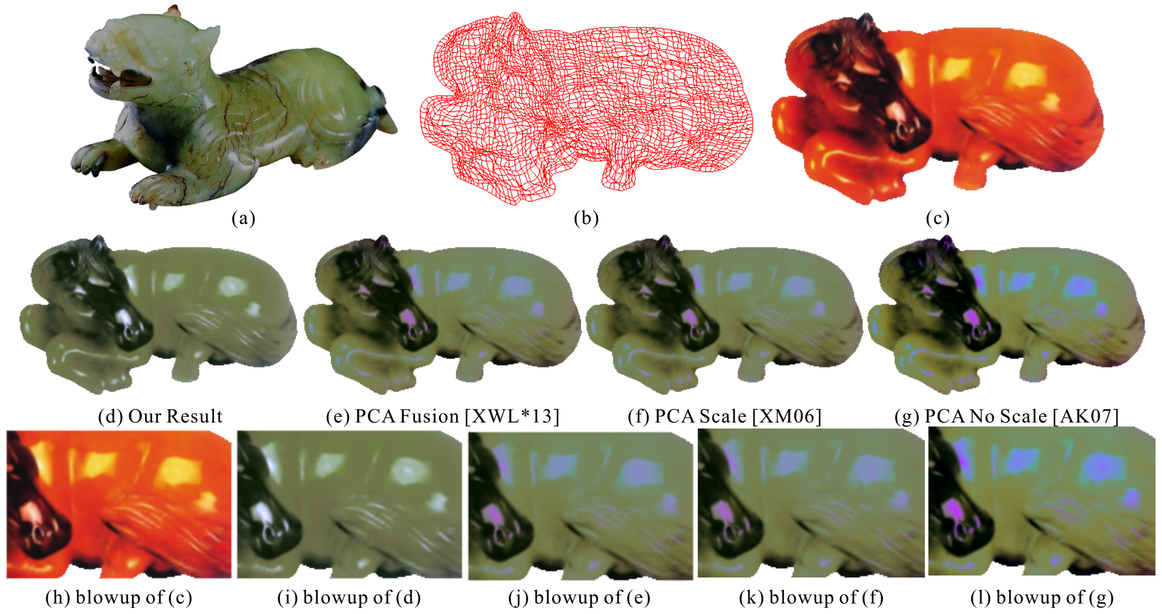


Figure 3: *Single-swatch color transfer using different methods: (a) Reference Image, (b) Mesh grids, (c) Rasterized gradient mesh, (d) Our result, (e) PCA Fusion [XWL*13], (f) PCA with scaling, (g) PCA without scaling. Note that our result does not have abnormal colors and faithfully preserves the details of the gradient meshes. λ is set to 1 in this example.*

adopt this way to calculate w_i's, given by

$$w_i = d_i / \sum_{j=1}^{M} d_j, \tag{12}$$

where d_i denotes the reciprocal of the Mahalonobis distance from a given color c^t to a target color swatch $I^t(i)$. d_i is given by

$$d_i = \frac{1}{\sqrt{(c^t - \eta_s(i))^T M_s(i)^{-1}(c^t - \eta_s(i))}}, \tag{13}$$

where $\eta_s(i)$ and $M_s(i)$ denote the mean vector and covariance matrix of swatch $I^t(i)$, respectively.

Figure 4 shows an example of using our multi-swatch color transfer scheme. As shown in Figure 4(e), the color style of the recolored peppers is quite close to the reference swatches. Compared to our result, the results generated by [XWL*13, XM06, AK07] are obviously yellowish as shown in Figure 4(f)(g)(h), especially for the green pepper and the stems. Note that we transfer the color of a light green apple to the dark green pepper. Moreover, our method better maintains the contrast of the gradient mesh because of the gradient constraint in our model.

5. Experimental Results

We generate gradient mesh vector graphics based on the algorithms from Sun et al. [SLWS07] and Lai et al. [LHM09]. For a given raster image, we first apply Lai et al.'s algorithm [LHM09] to create a gradient mesh automatically. If the quality of the rasterized gradient mesh is not very good, we optimize the gradient mesh using Sun et al.'s algorithm [SLWS07]. Note that [LHM09] extends the gradient mesh to tolerate image holes. To utilize [SLWS07], we decompose one hole-tolerated gradient mesh to two normal meshes at the hole. When the input image is complex, the image is segmented into several parts by manual or automatic methods [FH04, RKB04], and each part is approximated by one gradient mesh. Table 3 shows the number and size of the gradient meshes and the corresponding rasterized images used in the experiments.

5.1. The Impact of λ

In our model (8), we use the parameter λ to balance the influence of the color distribution term and the gradient term. In most of our experiments, we set $\lambda = 1$, which generates pleasing results in most cases. In this subsection, we use an example (Figure 5) to illustrate the impact of λ. As shown in Figure 5(f), when we only use the color distribution term E_1' in (8), i.e. $\lambda = 0$, the recolored result can faithfully capture the color features of the reference swatches. However, some details of the original gradient mesh are destroyed, for instance the artifacts in the color transition regions. When we use the gradient term as a constraint and set $\lambda = 1$, the artifacts are removed as shown in Figure 5(g). Further increasing λ can better preserve the

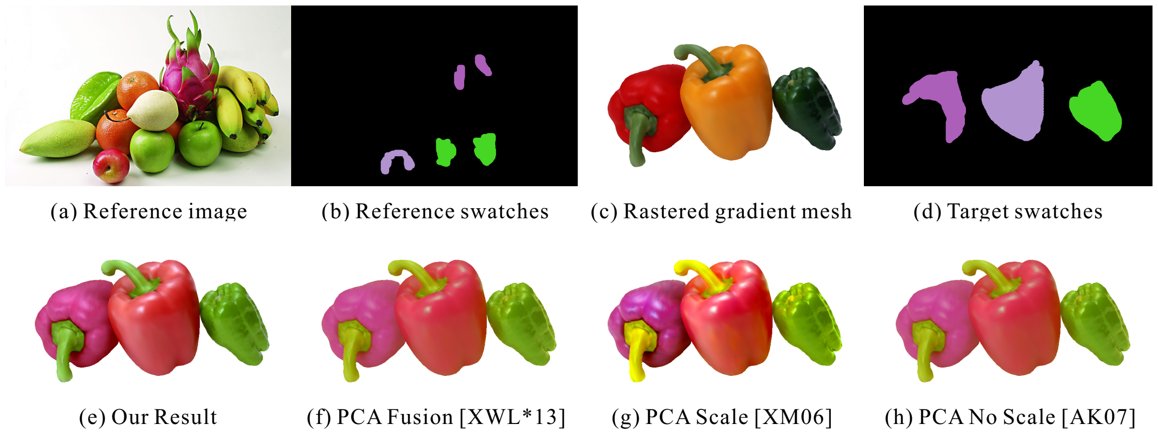


Figure 4: *Multi-swatch color transfer using different methods. (a) Reference image, (b) Reference swatches, (c) Rasterized gradient mesh, (d) Target swatches, (e) Our result, (f) Result of PCA Fusion, (g) Result of PCA with scaling, (h) Result of PCA without scaling. Our result is more pleasing than the results of other methods. Note that a swatch pair is marked with the same colors.* λ *is set to 1 in this example.*

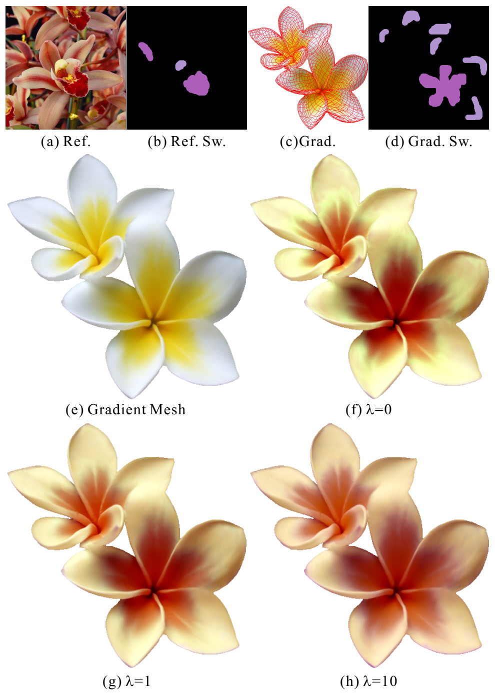


Figure 5: *The influence of* λ *on the results. (a) Reference image, (b) Reference swatches, (c) Gradient mesh, (d) Gradient mesh swatches, (e) Rasterized gradient mesh, (f) Result of* $\lambda = 0$*, (g) Result of* $\lambda = 1$*, (h) Result of* $\lambda = 10$*. As shown in (g), we obtain a good balance between the influence of the color distribution term and gradient term by setting* $\lambda = 1$.

details, but reduces the ability to match the color distribution as shown in Figure 5(h).

5.2. Visual Comparisons

Figures 1, 3, 4, 6, 7, 8 show different examples recolored by our method. Among the figures, the results in Figures 1, 3, 6 and 8 are generated by our single-swatch color transfer method, and the others are generated by our multi-swatch color transfer method. For the single-swatch color transfer, no user input is required. For the multiple-swatch method, the only user intervention in our system is to specify the color swatches. For example, in Figure 4, we use three pairs of color swatches. Each pair of swatches is labeled with the same color in the figure. In Figures 4 and 7, each of the three peppers is represented by a single gradient mesh, and just one color swatch is specified for each pepper. For the purpose of comparison, we extend the two versions of the PCA method [XM06, AK07] to handle gradient mesh color transfer in our experiments. The results of the PCA fusion method [XWL*13] are also given.

As shown in these figures, although all the four methods can change the color styles of the gradient mesh towards the reference images, the results of our method are visually more pleasing than those of PCA and PCA-Fusion. In general, our method can faithfully capture the color statistics of the reference images, avoid abnormal colors that are not present in the reference image, and preserve the detail information of the gradient meshes. For instance, in the cloud example in Figure 6, the results of PCA-Fusion and PCA with scaling have some green colors in the sky (see Figure 6(e)(f)); the results of PCA without scaling have some blue colors in the sky (see Figure 6(g)). Neither the green colors nor the blue colors are present in the reference image (see Figure 6(a)). For the results of multi-swatch method, we can observe similar situations in Figure 7.

As mentioned in Section 1, the PCA-Fusion method

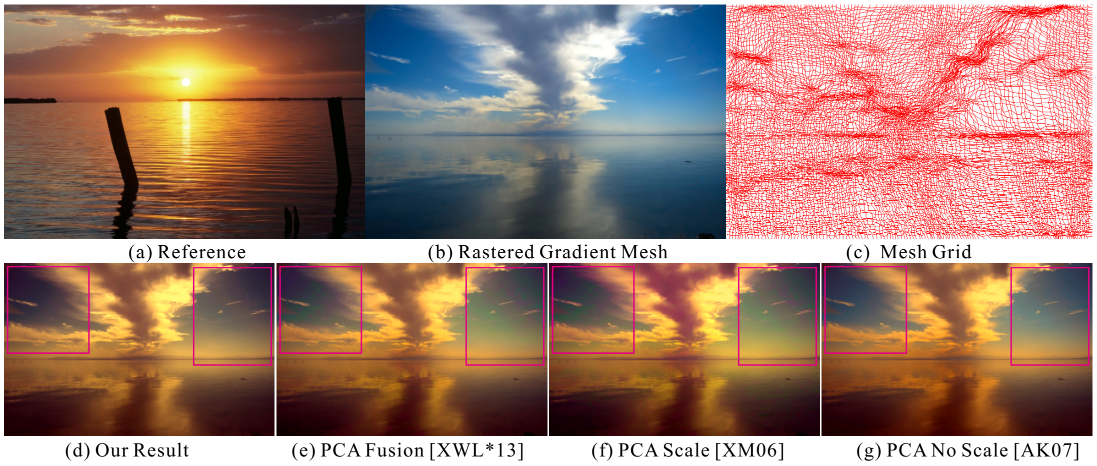


Figure 6: *Results of the cloud example. Note the unexpected green colors and blue colors in the highlighted regions of figures (e)(f)(g). The color of our method is visually closer to the reference image. λ is set to 1 in this example.*

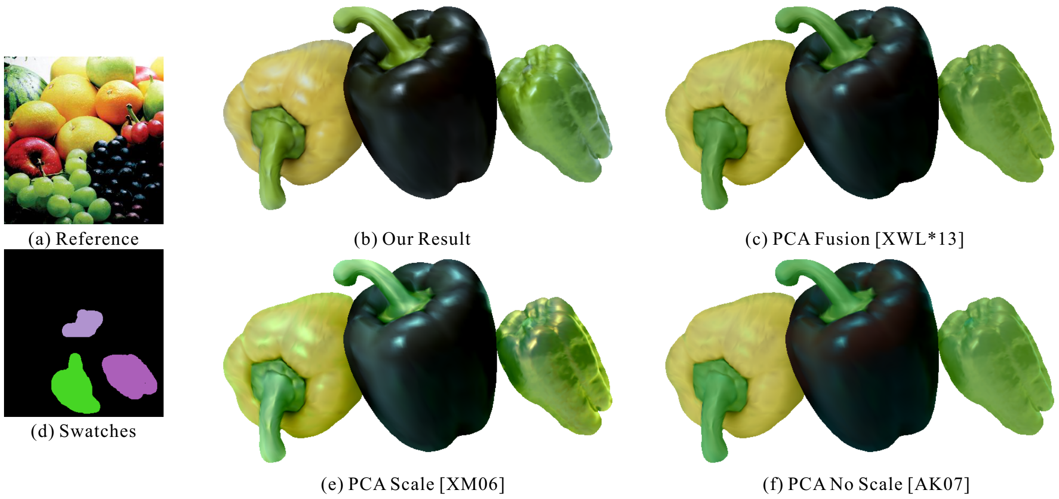


Figure 7: *Results of the peppers example: (a) Reference image; (b) Reference swatches; (c) Our Result; (d) Result of PCA fusion [XWL*13];(e) Result of PCA with scaling; (f) Result of PCA without scaling. Notice the colors of the green pepper, the stem and the highlight areas, the colors of our method are more similar to those of the reference image. The gradient mesh and its swatches are the same in Figure 4. Note that a swatch pair is marked with the same colors. λ is set to 1 in this example.*

may cause some artifacts during the fusion step. Even if a gradient-preserving step [XWL*13] is applied to relieve the artifacts, they are still visible sometimes. As shown in Figure 8(f), the result of PCA-Fusion has some extra textures which are not included in the original gradient mesh (see Figure 8(a)), while our result faithfully preserves the details (see Figure 8(e)).

5.3. Gradient Mesh v.s. Rasterized Gradient Mesh

In [XWL*13], the authors already pointed out the need to directly perform color transfer on gradient meshes. Performing color transfer on the rasterized gradient mesh and regenerating the gradient mesh not only consumes extra computing time, but also changes the structure of the gradient mesh. The changes in the structure are mainly caused by two reasons. First, the regenerating processes in [SLWS07,LHM09] are approximating operations, therefore, details of recolored image will be blurred. Second, performing the color transfer on the gradient mesh and its raster image may produce different results for multi-swatch color transfer. In multi-swatch color transfer, we sum the weighted result of single swatches. Since we process grid points or pixels separately, neighboring grid points or pixels may have inconsistent weighting. For gradient meshes, they have their own structures which naturally avoid color bleeding. That is, even inconsistent weighting in neighboring grid points will generate smooth transitions. For the rasterized images of gradient meshes, this structure information is lost. Inconsistent weighting in neighboring pixels on the raster image may introduce obvious artifacts inside the inner regions of the gradient mesh patches. The artifacts can destroy the smooth transition of the gradient mesh. Figure 9 shows an example. In this figure, the black, yellow, and light green colors are transferred to the orange, red, and dark green peppers, respectively. As we can observe in Figure 9(e), there is an unexpected black band near the pepper stem. In contrast, the recolored gradient mesh (Figure 9(d)) does not have such artifacts, and its color transition remains as smooth as the original gradient mesh (Figure 9(c)). The reason for the artifacts is that the pixels in inner regions may be mapped to colors quite different from neighboring pixels. This problem is more likely to appear when the swatches have quite different colors. For gradient meshes, only the colors and gradients of grid points are transformed, which guarantees inner regions have smooth transition.

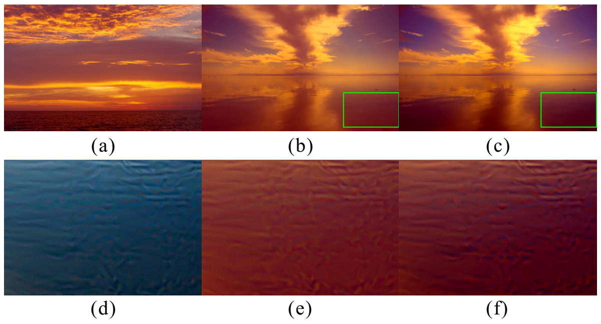


Figure 8: *Results of the cloud example: (a) Reference image; (b) Our Result; (c) Result of PCA-Fusion [XWL*13]; (d) Blow-up of gradient mesh (enhanced); (e) Blow-up of Our Method (enhanced) (f) Blow-up of PCA-Fusion (enhanced). The PCA-Fusion method incurs small artifacts (extra textures), while our method has no such artifacts. Please view the digital images to observe the artifacts. The gradient mesh used in the same as in Figures 6(b)(c). λ is set to 1 in this example.*

5.4. User Study

We conducted a user study to evaluate our method. Our user study consisted of a questionnaire given to 20 participants including 10 males and 10 females. Each questionnaire comprised twenty images, with five different images modified using four methods: our linear optimization method (LO for short) and three existing methods. The images were shown in a random order to each participant. Among them, 5 persons are artists working in design or other related areas. These people were asked to score each recolored result image according to 3 criteria:

1. Color similarity between the result image and the example one. Since colors in the example image should be transferred, here we describe the color similarity as the similarity of color contrast or lighting ratio.
2. Color transition of the result image. Some methods may produce unnatural effects or introduce extra colors, and these kinds of artifacts would be expected to receive a low mark.
3. Detail delineation of the result image. If a result image

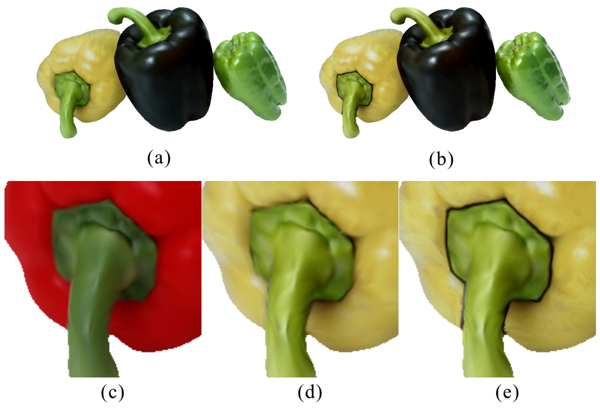
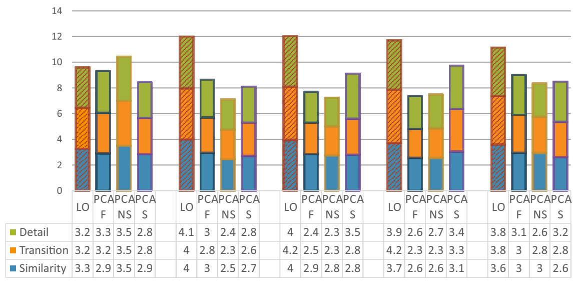


Figure 9: *Comparison between results on gradient mesh vs on raster images. (a) Result on gradient mesh, (b) Result on raster image, (c) blow up of original gradient mesh, (d) blow up of (a), (e) blow up of (b). The swatches used are the same as those in Figure 7.*



	LO	PCAF	PCANS	PCAS	LO	PCAF	PCANS	PCAS	LO	PCAF	PCANS	PCAS	LO	PCAF	PCANS	PCAS	LO	PCAF	PCANS	PCAS
🟩 Detail	3.2	3.3	3.5	2.8	4.1	3	2.4	2.8	4	2.4	2.3	3.5	3.9	2.6	2.7	3.4	3.8	3.1	2.6	3.2
🟧 Transition	3.2	3.2	3.5	2.8	4	2.8	2.3	2.6	4.2	2.5	2.3	2.8	4.2	2.3	2.3	3.3	3.8	3	2.8	2.8
🟦 Similarity	3.3	2.9	3.5	2.9	4	3	2.5	2.7	4	2.9	2.8	2.8	3.7	2.6	2.6	3.1	3.6	3	3	2.6

Figure 10: *Average scores of 4 methods. LO stands for our linear optimization method, PCAF for PCA Fusion [XWL*13], PCAS for PCA Scale [XM06], and PCANS for PCA No Scale [AK07]. From left to right, the image groups correspond to Figures 6, 3, 1, 4, and 7.*

loses texture, or adds texture unreasonably, then its score would be lower.

Figure 10 shows the results of user study. There are 5 groups of images. The table under the bars list the average scores given by 20 participants. Note that each criterion has a five-point scale to evaluate the performance of different methods. The higher the score is, the better the result looks. The histogram illustrates the accumulated average scores of the 3 criteria. This graph informally indicates that people prefer the results of our method over the others.

For each criterion, we further analyze user study data by means of repeated measures analysis of variance (ANOVA). As listed in Table 1, the method type and the image group are the independent variables, and the user score is the dependent variable. We can see that the effect of the method type is significant, F=16.65, Sig<0.001 for color similarity; F=28.5, Sig<0.001 for color transition; F=24.72, Sig<0.001 for detail delineation. The effect of the image group, on the other hand, is not significant, since F=0.62, Sig>0.05 for color similarity; F=2.46, Sig>0.05 for color transition; F=0.72, Sig>0.05 for detail delineation. Furthermore, the LSD multiple comparison among different methods in Table 2 shows that our method is significantly better than PCA Fusion [XWL*13], PCAS for PCA Scale [XM06], and PCANS for PCA No Scale [AK07], in terms of the three criteria respectively. These three comparitive methods have similar performance in terms of color similarity and color transition, while PCA Scale is significantly better than the other two with respect to detail delineation.

Table 1: *User Study Repeated Measures Analysis*

Similarity source	Type III sum of sqaures	df	Mean sqaure	F	Sig
Method	57.31	3	19.10	16.65	0.00
Image group	1.27	4	0.32	0.62	0.65
Transition source	**Type III sum of sqaures**	**df**	**Mean sqaure**	**F**	**Sig**
Method	94.89	3	31.63	28.50	0.00
Image group	3.82	4	0.95	2.46	0.053
Detail source	**Type III sum of sqaures**	**df**	**Mean sqaure**	**F**	**Sig**
Method	69.26	3	23.09	24.72	0.00
Image group	1.40	4	0.35	0.72	0.58

Table 2: *User Study LSD Multiple Comparisons*

	Similarity Sig	Transition Sig	Detail Sig
LO vs. PCAF	0.00	0.00	0.00
LO vs. PCANS	0.00	0.00	0.00
LO vs. PCAS	0.00	0.00	0.00
PCAF vs. PCANS	1.00	0.38	0.07
PCAF vs. PCAS	0.81	033	0.02
PCANS vs. PCAS	0.76	0.08	0.00

Name	Num.	Patches	Resolution
jade (Fig. 1)	3	28×13 28×8 28×16	600×450
horse (Fig. 3)	1	44×78	300×174
peppers (Fig. 4)	3	30×33 26×33 38×63	800×500
plumeria (Fig. 5)	2	37×32 37×30	446×456
cloud (Fig. 6)	1	126×127	800×511

Table 3: *The size of gradient meshes and the corresponding raster images used in the experiments.*

5.5. Time Performance

We also evaluate the complexity and running time of our method. Our method is implemented using Matlab-C++ mixed programming. The N-dimensional probability distribution transfer and solving (8) are implemented in Matlab. They are compiled to DLL files, which are then called in our C++ interface. Since Matlab is not good at loop operation and extra time is spent on the data type conversion between Matlab and C++, our method is currently slower than the PCA-Fusion method [XWL*13]. The running time of our method mainly depends on the time spent on the N-dimensional probability distribution transfer method [PKD07], solving (8), and performing (9) and (10). The first two parts are affected by the resolution of the rasterized gradient mesh, which are used for calculating the N-dimensional distribution matching and the gradient constraint. The computational complexity of the N-dimensional distribution transfer is $O(n)$ [PKD07], where n is the number of pixels in the rasterized gradient mesh. As we can see from the supplemental material, the complexity to calculate the terms of the linear system is also $O(n)$. The complexity to solve the 12×12 linear system is $O(1)$. In performing (9) and (10), the complexity is $O(k)$, here k is the number of control points of the gradient mesh. The running time of our method is shown by parts in Table 4. In general, it takes our method at most 90 seconds for the examples in the experiments. The time difference between the raster images and the gradient meshes are not large. This is because the main time is spent on the N-dimensional probability distribution transfer and the solving (8).

If running speed is a concern, we can improve the speed of our method in two ways. One is to implement the Matlab part with C++. The other is to use a rasterized gradient mesh of smaller resolution for calculating the N-dimensional distribution matching and the gradient constraint. This can greatly improve the running speed with little impact on the recolored image quality. Figure 11 shows an example. If we use a rasterized image of half width and half height, the run time of Part 1 and Part 2 (defined in Table 4) are reduced

	Gradient Mesh			Raster Image		
Part	1	2	3	1	2	3
jade (Fig. 1)	4.39	38.15	0.01	4.42	38.16	0.20
horse (Fig. 3)	1.95	7.87	0.02	2.20	8.20	0.05
peppers (Fig. 4)	2.37	12.97	0.16	2.38	13.14	3.68
plumeria (Fig. 5)	1.17	5.09	0.06	1.24	5.02	1.79
cloud (Fig. 6)	20.28	71.75	0.06	20.17	71.35	0.59

Table 4: *Comparisons of running time (seconds) on gradient mesh and the corresponding raster image. "Part 1" refers to the N-dimensional probability distribution transfer. "Part 2" refers to solving (8). "Part 3" refers to performing (9) and (10).*

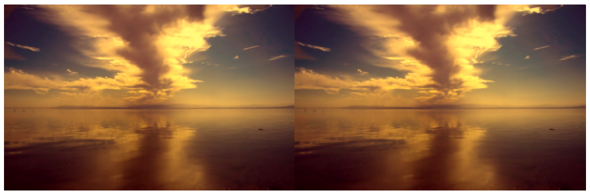

(a) (b)

Figure 11: *Comparison of results using a smaller raster image. (a) The recolored gradient mesh using a rasterized image whose resolution is* 800×511.*(b) The recolored gradient mesh using a rasterized image whose resolution is* 400×256. *By using a raster image of half width and half height, the run time of Part 1 and Part 2 (defined in Table 4) are reduced from 20.28 and 71.75 to 10.14 and 17.79, respectively. The PSNR of these two recolored figures is 50.45. The example image and the gradient mesh are the same as in Figure 6.*

from 20.28 and 71.75 to 10.14 and 17.79, respectively, while the recolored results have no visual differences. The PSNR of these two results is 50.45. Another possible way is to use the GPU to accelerate the histogram based operations in the N-dimensional distribution transfer as in [SH07].

6. Limitations and Discussion

One limitation of our method is that finding a suitable reference image is sometimes difficult, which is the same as other color transfer methods. Therefore, we would like to investigate more applications in our framework, which are more intuitive, such as color theme enhancement [YP08, WYW*10], and scribble-based recoloring [LLW04]. Another limitation is that there is no standard way to choose an optimal λ in (8). It may require user tuning based on the visual effects. Nevertheless, pleasing results can be obtained by setting $\lambda = 1$ in most cases, as shown in the figures of this paper. In our current work, we focus on the linear transform, which makes use of the linear property of gradient meshes. In future work, we will also consider non-linear transforms for color editing.

7. Conclusion

We have proposed an optimal linear operator for gradient meshes color transfer. The color transfer process is approximated by a linear operator solved from a minimization problem. The minimization problem explicitly considers both the color statistics and the constraints to match the color features and meanwhile avoid artifacts. The colors and color gradients of control points in gradient meshes are then transformed by the linear operator. The grid structure of the gradient mesh is preserved. The experimental results show that our method can generate pleasing recolored gradient mesh.

Acknowledgement

We thank Prof. Xinyao Li at Sun Yat-Sen University for helping conducting the ANOVA analysis. The work was mainly supported by RGC General Research Fund from Hong Kong (Project No.: CityU 115612), and partially supported by the Fundamental Research Funds for the Central Universities (Project No.: 531107040842), Specialized Research Fund for the Doctoral Program of Higher Education, China (Project No.: 20110032120041), NSFC (Project No. 61272293), Shenzhen Nanshan Innovative Institution Establishment Fund (Project No. KC2013ZDZJ0007A), and Shenzhen Basic Research Project (Project No. JCYJ20120619152326448).

References

[AK07] ABADPOUR A., KASAEI S.: An efficient pca-based color transfer method. *Journal of Visual Communication and Image Representation 18*, 1 (2007), 15–34. 2, 3, 5, 6, 7, 9, 10

[AP10] AN X., PELLACINI F.: User-controllable color transfer. *Computer Graphics Forum 29*, 2 (2010), 263–271. 3, 5

[CHS08] CHARPIAT G., HOFMANN M., SCHÖLKOPF B.: Automatic image colorization via multimodal predictions. In *Computer Vision ĺC ECCV* (2008), vol. 5304, pp. 126–139. 3

[CSN03] CHANG Y., SAITO S., NAKAJIMA M.: A framework for transfer colors based on the basic color categories. In *Proc. of Computer Graphics International* (2003), pp. 176–181. 3

[CSN07] CHANG Y., SAITO S., NAKAJIMA M.: Example-based color transformation of image and video using basic color categories. *IEEE Transactions on Image Processing 16*, 2 (2007), 329–336. 3

[DBZP10] DONG W., BAO G., ZHANG X., PAUL J.-C.: Fast local color transfer via dominant colors mapping. In *ACM SIGGRAPH ASIA Sketches* (2010), pp. 46:1–46:2. 3

[Fer64] FERGUSON J.: Multivariable curve interpolation. *Journal of ACM 11*, 2 (1964), 221–228. 3

[FH04] FELZENSZWALB P. F., HUTTENLOCHER D. P.: Efficient graph-based image segmentation. *International Journal of Computer Vision 59* (2004), 2004. 6

[FPC*14] FARIDUL H. S., POULI T., CHAMARET C., STAUDER J., TREMEAU A., REINHARD E.: A survey of color mapping and its applications. In *Eurographics State of the Art Report* (2014), pp. 43–67. 2

[GD05a] GRUNDLAND M., DODGSON N. A.: Color histogram specification by histogram warping. In *Processings of the SPIE: Color Imaging X: Processing, Hardcopy, and Applications* (2005), vol. 5667, pp. 610–621. 2

[GD05b] GRUNDLAND M., DODGSON N. A.: Color search and replace. In *Proc. of the First Eurographics Conference on Computational Aesthetics in Graphics, Visualization and Imaging* (2005), pp. 101–109. 3

[GH03] GREENFIELD G.-R., HOUSE D.-H.: Image recoloring induced by palette color associations. *Journal of WSCG 11*, 1 (2003), 189–196. 2

[HK05] HORIUCHI T., KOTERA H.: Colorization for monochrome image with texture. In *Proc. of the 13th Color Imaging Conference* (2005), pp. 245–250. 3

[HTC*05] HUANG Y.-C., TUNG Y.-S., CHEN J.-C., WANG S.-W., WU J.-L.: An adaptive edge detection based colorization algorithm and its applications. In *Proc. of the 13th annual ACM international conference on Multimedia* (2005), pp. 351–354. 3

[ICOL05] IRONY R., COHEN-OR D., LISCHINSKI D.: Colorization by example. In *Proceedings of the Sixteenth Eurographics Conference on Rendering Techniques* (2005), pp. 201–210. 3

[JLWT04] JI Y., LIU H.-B., WANG X.-K., TANG Y.-Y.: Color transfer to greyscale images using texture spectrum. In *Proc. of International Conference on Machine Learning and Cybernetics* (Aug 2004), vol. 7, pp. 4057–4061. 3

[Kot05] KOTERA H.: A scene-referred color transfer for pleasant imaging on display. In *Proc. of IEEE International Conference on Image Processing* (Sept 2005), vol. 2, pp. II–5–8. 2

[LHM09] LAI Y.-K., HU S.-M., MARTIN R. R.: Automatic and topology-preserving gradient mesh generation for image vectorization. *ACM Transactions on Graphics 28*, 3 (2009), 1–8. 1, 2, 3, 6, 8

[LHZ08] LI J., HAO P., ZHANG C.: Transferring colours to grayscale images by locally linear embedding. In *Proceedings of the British Machine Vision Conference* (2008), pp. 83.1–83.10. 3

[LLW04] LEVIN A., LISCHINSKI D., WEISS Y.: Colorization using optimization. *ACM Transactions on Graphics 23*, 3 (2004), 689–694. 3, 11

[LWCO*07] LUAN Q., WEN F., COHEN-OR D., LIANG L., XU Y.-Q., SHUM H.-Y.: Natural image colorization. In *Proceedings of the 18th Eurographics conference on Rendering Techniques* (2007), pp. 309–320. 3

[LWX07] LUAN Q., WEN F., XU Y.-Q.: Color transfer brush. In *Proc. of the 15th Pacific Conference on Computer Graphics and Applications* (2007), pp. 465–468. 3

[MS03] MOROVIC J., SUN P.-L.: Accurate 3d image colour histogram transformation. *Pattern Recognition Letters 24*, 11 (2003), 1725 – 1735. 2

[MTN09] MORIMOTO Y., TAGUCHI Y., NAEMURA T.: Automatic colorization of grayscale images using multiple images on the web. In *SIGGRAPH: poster* (2009), pp. 59:1–59:1. 3

[NN05] NEUMANN L., NEUMANN A.: Color style transfer techniques using hue, lightness and saturation histogram matching. In *Computational Aesthetics in Graphics, Visualization and Imaging* (2005), pp. 111–122. 2

[PKD07] PITIÉ F., KOKARAM A. C., DAHYOT R.: Automated colour grading using colour distribution transfer. *Computer Vision and Image Understanding 107*, 1-2 (2007), 123–137. 2, 3, 4, 5, 10

[PR11] POULI T., REINHARD E.: Progressive color transfer for images of arbitrary dynamic range. *Computers and Graphics 35*, 1 (2011), 67 – 80. 2

[RAGS01] REINHARD E., ASHIKHMIN M., GOOCH B., SHIRLEY P.: Color transfer between images. *IEEE Computer Graphics and Applications 21*, 5 (2001), 34–41. 1, 2, 3, 5

[RKB04] ROTHER C., KOLMOGOROV V., BLAKE A.: "grabcut": interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics 23*, 3 (2004), 309–314. 6

[RP11] REINHARD E., POULI T.: Colour spaces for colour transfer. In *Proceedings of the Third International Conference on Computational Color Imaging* (2011), pp. 1–15. 2

[SDYL12] SU Z., DENG D., YANG X., LUO X.: Color transfer based on multiscale gradient-aware decomposition and color distribution mapping. In *Proceedings of the 20th ACM international conference on Multimedia* (2012), pp. 753–756. 3

[SH07] SCHEUERMANN T., HENSLEY J.: Efficient histogram generation using scattering on gpus. In *Proc. of the Symposium on Interactive 3D Graphics and Games* (2007), pp. 33–37. 11

[SLWS07] SUN J., LIANG L., WEN F., SHUM H.-Y.: Image vectorization using optimized gradient meshes. *ACM Transactions on Graphics 26*, 3 (2007), 11. 1, 2, 3, 6, 8

[TJT05] TAI Y.-W., JIA J.-Y., TANG C.-K.: Local color transfer via probabilistic segmentation by expectation-maximization. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition* (2005), vol. 1, pp. 747–754. 3

[WAM02] WELSH T., ASHIKHMIN M., MUELLER K.: Transferring color to greyscale images. *ACM Transactions on Graphics 21*, 3 (2002), 277–280. 3

[WDM*11] WU F., DONG W., MEI X., ZHANG X., JIA X., PAUL J.-C.: Distribution-aware image color transfer. In *SIGGRAPH Asia Sketches* (2011), pp. 8:1–8:2. 3

[WH04] WANG C. M., HUANG Y. H.: A novel color transfer algorithm for image sequences. *Journal of Information Science and Engineering 20*, 6 (2004), 1039 – 1056. 2

[WHCO08] WEN C.-L., HSIEH C.-H., CHEN B.-Y., OUHYOUNG M.: Example-based multiple local color transfer by strokes. *Computer Graphics Forum 27*, 7 (2008), 1765–1772. 3

[WSM99] WEEKS A. R., SARTOR L. J., MYLER H. R.: Histogram specification of 24-bit color images in the color difference (c-y) color space. *Journal of Electronic Imaging 8*, 3 (1999), 290–300. 2

[WYW*10] WANG B., YU Y., WONG T.-T., CHEN C., XU Y.-Q.: Data-driven image color theme enhancement. *ACM Transactions on Graphics 29*, 6 (2010), 146:1–146:10. 11

[XM06] XIAO X.-Z., MA L.-Z.: Color transfer in correlated color space. In *Proc. of the ACM international conference on Virtual reality continuum and its applications* (2006), pp. 305–309. 2, 3, 5, 6, 7, 9, 10

[XM09] XIAO X.-Z., MA L.-Z.: Gradient-preserving color transfer. *Computer Graphics Forum 28*, 7 (2009), 1879–1886. 3, 4

[XWL*13] XIAO Y., WAN L., LEUNG C.-S., LAI Y.-K., WONG T.-T.: Example-based color transfer for gradient meshes. *IEEE Transactions on Multimedia 15*, 3 (2013), 549–560. 2, 4, 5, 6, 7, 8, 9, 10

[YP08] YANG C.-K., PENG L.-K.: Automatic mood-transferring between color images. *IEEE Computer Graphics and Applications 28*, 2 (2008), 52–61. 11

[YS06] YATZIV L., SAPIRO G.: Fast image and video colorization using chrominance blending. *IEEE Transactions on Image Processing 15*, 5 (2006), 1120–1129. 3