Computational Visual Media
https://doi.org/10.1007/s41095-021-0258-0

Research Article

Image-guided color mapping for categorical data visualization

Qian Zheng[1], Min Lu[2], Sicong Wu[2], Ruizhen Hu[2], Joel Lanir[3], and Hui Huang[2] (✉)

Abstract Appropriate color mapping for categorical data visualization can significantly facilitate the discovery of underlying data patterns and effectively bring out visual aesthetics. Some systems suggest predefined palettes for this task. However, a predefined color mapping is not always optimal, failing to consider users' needs for customization. Given an input categorical data visualization and a reference image, we present an effective method to automatically generate a coloring that resembles the reference while allowing classes to be easily distinguished. We extract a color palette with high perceptual distance between the colors by sampling dominant and discriminable colors from the image's color space. These colors are assigned to given classes by solving an integer quadratic program to optimize point distinctness of the given chart while preserving the color spatial relations in the source image. We show results on various coloring tasks, with a diverse set of new coloring appearances for the input data. We also compare our approach to state-of-the-art palettes in a controlled user study, which shows that our method achieves comparable performance in class discrimination, while being more similar to the source image. User feedback after using our system verifies its efficiency in automatically generating desirable colorings that meet the user's expectations when choosing a reference.

Keywords color palette; discriminability; image-guided; categorical data visualization

1 Suzhou University of Science and Technology, Suzhou 215009, China. E-mail: qianzheng85@gmail.com.

2 Shenzhen University, Shenzhen 518052, China. E-mail: M. Lu, lumin.vis@gmail.com; S. Wu, wuusicong@gmail.com; R. Hu, ruizhen.hu@gmail.com; H. Huang, hhzhiyan@gmail.com (✉).

3 The University of Haifa, Haifa 3498838, Israel. E-mail: ylanir@is.haifa.ac.il.

Manuscript received: 2021-07-1; accepted: 2021-10-01

1 Introduction

Color is one of the most important visual channels in data visualization. It is widely used to encode different data attributes and facilitate visual search. In particular, when presenting categorical data, a good color strategy can generate a visually attractive representation, clearly discriminate between different data categories, and help users effectively gain understanding and insight from the data [1, 2]. Nonetheless, it is often difficult for even professional designers to design appropriate color mappings for data classes, especially when they have specific requirements, e.g., tuning color for a certain design atmosphere or personality. Designers usually go through a process of trial and error to find suitable colors for tasks.

Generating a suitable color mapping for particular data often has two steps: designing or choosing a good color palette, and assigning the selected colors to different classes. Existing strategies mainly focus on one of the two steps. For example, ColorBrewer [3] and Colorgorical [4] provide predefined color palettes that ensure discriminability and aesthetics. Color assignment methods automatically assign the colors within a given palette to the data classes in the visualization chart according to semantics [5], visual appeal [6], or class structure visibility [7]. As color appearance often depends on the relationship to surrounding regions [8], combining the two steps can potentially improve class structure visibility [9].

Informal interviews with designers indicate that they often use images as references for color mapping. From a reference image, designers can extract its prominent colors to generate a visually pleasing color palette, as well as use it to assign selected colors to the classes. If the distribution of main colors, i.e., their positions, proportions, and harmonious spatial

arrangement, can be kept in the target, we assume that aesthetic perception of the reference image is thus transferred to the target.

Inspired by this, we propose to generate image-guided colorings of a given categorical visualization. A good coloring should allow easy perception of class structure and resemble the reference image in order to meet the user's expectations; see Fig. 1.

To achieve this goal, the selected colors need to be discriminable, easy to identify, and representative of the reference image. Furthermore, Wang et al. [7] suggested that the colors assigned to overlapping classes should have large perceptual differences. Differing from them, we additionally wish to retain the spatial distribution of colors of the source image, preserving both their adjacency relationships and the locations of dominant colors. Recently, Lu et al. [9] proposed to solve palette extraction and color assignment from the full RGB space in a unified optimization framework. However, an image's limited colors can cause the optimization to get stuck in a local optimum. Hence we solve palette extraction and color assignment in two stages; see Figs. 2(c) and 2(e).

To extract distinct colors, we look at palette selection as a farthest-point sampling problem [10] in the color space of the image, considering linear separation [11], the minimal color CIEDE2000 distance between any two colors, and their frequency of occurrence in the image. We iteratively update the color samples after initialization.

We solve the color assignment problem by finding a consistent one-to-one correspondence between the colors and class labels. This is formulated as an integer quadratic program containing unary and binary terms. The unary term evaluates each class–color assignment based on spatial position. The binary term assesses a pair of assignments based on the distance of two classes, the perceptual distance of the two colors, and their adjacency in the reference. An approximately optimal solution is found efficiently for this NP-hard problem.

In summary, the main contributions of this paper include:

- a color mapping method to extract the color palette of an image and coherently apply it to a categorical data visualization to generate distinct and aesthetic color appearances,
- a new way to extract a categorical color scheme representative of the source image, which can create a more distinct color palette than existing methods, and
- an optimization method to produce the color–class mapping, considering both spatial relations and color discriminablity, with comparable performance in class discrimination to prior works [7], yet being similar to the source image.

Fig. 1 A given categorical data visualization is colored to follow colors in different images. Each coloring is diverse with good visual separability of class structures, and is similar to its reference image. Class separation scores are 18,472, 15,382, and 17,149, respectively.

2 Related work

Existing related work on color design can be divided into three categories: color palette design, palette extraction from images, and color suggestion.

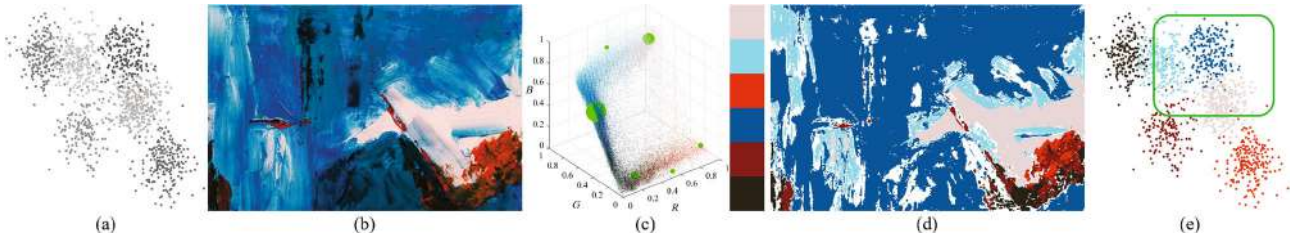


Fig. 2 Pipeline. Given an input categorical data visualization with M classes (a) and a reference image (b), we extract a distinct M-color palette from the image, indicated in RGB space by the green spheres (c). The radius of each sphere encodes the color's occurrences. After analyzing the color spatial distribution in the image (d) and the class distinctness of the given chart, we assign these colors to the classes (e). While making classes easy to discriminate, our mapping also follows colors in the source image. It keeps the major color adjacencies of the source, highlighted by the green box. It further retains the position of the light pink, a concentrated dominant color in the source.

2.1 Color palette design

Creating discriminable and aesthetically pleasing color palettes is a demanding task when visualizing categorical data. Guidelines for manually designing color palettes have been provided by several researchers. Colors in a palette design should meet the criterion of separability from each other [12] or overall compatibility [13]. Healey [11] proposed a method to choose effective colors allowing rapid and accurate identification of color targets. Recently, Bartram et al. [14] suggested that even a palette of a small set of colors could be manipulated to communicate emotions. These guidelines clarify the considerations for color design, but still, designing a palette from scratch is usually a cumbersome task for most users.

There are many online toolkits for color palette selection. Adobe Color (`https://color.adobe.com`) and COLOURLovers (`www.colourlovers.com`) are two websites commonly used to select color themes. ColorBrewer, developed by Harrower and Brewer [3], provides a large number of predefined, recognizable palettes for thematic maps. Recently, Gramazio et al. [4] proposed Colorgorical, a tool that allows users to create customized color palettes by balancing discriminability and aesthetic preference.

Methods for rating, expanding, or optimizing a given color palette have also been proposed. O'Donovan et al. [15] studied color compatibility theories using large datasets and offered a learned model to evaluate a coloring combination of five colors. Later, Kita and Miyata [16] extended this model to rate arbitrary-sized color combinations considering human aesthetic preferences, and extended color palettes while retaining color harmony. Fang et al. [17] optimized a given color palette by maximizing the perceptual distances while meeting a set of user-defined constraints.

However, choosing from pre-generated color palettes is not always intuitive, as color appearance depends on the relationships to surrounding regions and the spatial arrangement of classes varies according to the data.

2.2 Palette extraction from images

A possible route for generating color palettes is to capture the prominent colors from an input image. When representing the image, the discriminability of the extracted colors is not significant.

Lin and Hanrahan [18] created five-color themes from natural images based on a study considering the human perception. Their results closely match human-extracted themes. Poco et al. [19] developed a method to semi-automatically recover a color encoding from a bitmap visualization image by extracting the color and text information from its legend, enabling recoloring of the static image.

Recently, researchers in image processing looked at color palette extraction as a means to allow non-experts to recolor an input image by editing the colors in the extracted palette. In 2015, Chang et al. [20] proposed a variant of the k-means algorithm to generate a suitable palette of a user-chosen size. Tan et al. [21] proposed a geometric method, identifying the color palette as the vertices of a simplified convex hull of image pixels in RGB-space. They also proposed a simple scheme for automatic palette size selection in following work [22]. The picked colors do not necessarily exist in the image. A different approach proposed by Aksoy et al. [23] automatically estimates a statistical color model from an image based on an energy formulation. It automatically decides the sizes of prominent colors and exclusively selects colors that exist in the image. Zhang et al. [24] presented an approach to edit colors of an image so as to preserve inherent color characteristics of the source by extracting and adjusting a compact color palette.

Some approaches deal with image collections. The group color theme extraction method of Nguyen et al. [25] derives the group color theme from input images' individual color palettes, for recoloring multiple images consistently. To capture color styles of a fine art collection, Phan et al. [26] introduced a method to order extracted color palettes, enabling palette prediction and interpolation. These approaches focus on image collections with a consistent color style or color theme.

2.3 Color suggestion

Many methods take the characteristics and content of the data into consideration when performing coloring.

Some methods combine colors with semantics, so that the color mapping matches typical colors for items defining the categorical data. Lin et al. [5] presented an algorithm to automatically select semantically-resonant colors from the 20 common colors suggested by Tableau to represent data. These

resonant colors for each category were computed by finding relevant images using Google Image Search. Later Setlur and Stone [27] explored linguistic information to generate semantic coloring. However, not all classes shown in a visual chart have explicit semantics.

Some works aim to generate aesthetically pleasing color suggestions for uncolored pattern templates. Kim et al. [6] proposed a perceptually driven method for automatically coloring a 2D pattern template with a given color palette, guided by commonly accepted rules in color theories. They considered color harmony and luminance contrast of adjacent segments. Lin et al. [8] presented a probabilistic factor graph model for automatically generating color suggestions for an input pattern. The model was trained on example patterns made by experts. Due to the significant differences between patterns and visualization charts, their trained model does not work well on visualizations. Furthermore, the color variation between available example visualization charts is much smaller, making it hard to train the model.

Other works aim to generate color suggestions that make class structures more noticeable. Lee et al. [28] introduced the concept of class visibility to measure the utility of a color palette to present a categorical structure. Based on this concept, they proposed a method to optimize the luminance and saturation of assigned colors, to better display all class structures. Recently, Wang et al. [7] proposed a color assignment method with a given color palette for multi-category scatterplots, by modeling class separation considering spatial relationships, density, and point clusters. In following work, Lu et al. [9] introduced a data-aware color palette framework for both creating and assigning color palettes with maximal discriminability. While these systems are similar to our approach in considering color relationships as well as the spatial configuration of data, our method also takes the spatial distribution of colors in an image into consideration, making the colored visualization similar to the reference, to meet users' expectations.

Recently, Bohra and Gandhi [29] proposed a system for automatic colorization of graphic arts. Itautomatically selects a relevant reference image for any input pattern template from a built-in dataset, and then propagates the reference's colors to the input. The color propagation considers the composition of colors and the adjacency of colors, as we do. However, they do not encode color differences within the color propagation. Instead, our approach incorporates both the color spatial relations of the reference and class visibility of data for an optimal solution.

3 Palette extraction

A palette C containing $|C|$ colors is denoted by $C = \{c_1, \cdots, c_{|C|}\}$. If C is a distinct color palette, the minimal color distance of any two colors in the palette, denoted by

$$d_C = \min_{1 \leqslant a,b \leqslant |C|, a \neq b} E(c_a, c_b) \tag{1}$$

should be large. Here, $E(c_a, c_b)$ represents the difference of two colors c_a and c_b in the CIEDE2000 color distance metric [30], which is commonly used for measuring the difference of colors within a color palette [4, 9].

To form a distinct palette C of a given size M, the selected colors should be prominent, exist in the image, and approximate the image's color space. This is a farthest-point sampling problem in the color space and is hard to solve directly. To quickly generate an approximate optimal solution, we first reduce the

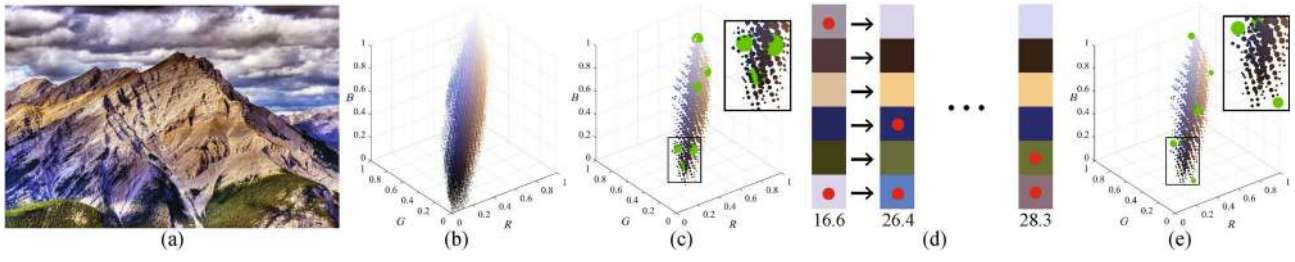


Fig. 3 Example of palette extraction. (a) Reference image, and (b) its colors in RGB space. (c) The color space is divided into voxels, and too light or too dark colors are moved. Green dots indicate the initially selected colors. (d) Left to right: initialization, after one iteration, and the final palette. Each iteration updates every color to increase the distinctness score. The two most similar colors at each step are highlighted by red dots; their distinctness score is shown below. (e) The final colors are shown as green dots in RGB space.

sampling space, create a good initialization, and then iteratively update the color samples; see Fig. 3.

Though an image's distribution (Fig. 2(b)) only covers a subset of the whole color space, it still has thousands of different colors. A minimum CIELAB interval, indicated by the noticeable difference function, is required to discriminate the colors of two graphical marks [31]. Thus, we perform color quantization by dividing the CIELAB color space into voxels in steps of 5 units along each axis. For each voxel, we record its frequency n_a (the number of pixels whose color falls inside it), and compute its mean color c_a. We then exclude colors that are too light ($L > 85$) or too dark ($L < 20$) [4]. To remove outliers of low frequency, we successively exclude the color with the lowest frequency until the frequency sum of all removed colors exceeds 3% of the input image. Figure 3(c) shows colors by spheres of different sizes according to frequency.

We use the initialization of Zhang et al. [24] to select M dominant colors. We successively choose the color with the highest frequency, and update the frequencies of remaining colors by different factors to penalize colors similar to it:

$$n_a^{\tau} = n_a^{\tau-1}(1 - \exp(-(E(c_a, c_b)/\sigma)^2)$$

where τ represents the iteration index, $n_a^0 = n_a$, and σ is set to 80 by default. We repeat this until M colors have been selected. However, some chosen colors may still be too similar to distinguish; see for example the light and dark gray marked by red dots in the leftmost palette in Fig. 3(d).

We gradually update the palette C to increase d_C by farthest-point sampling [10]. In each iteration, we successively move each color in the current palette to a new position that maximizes the minimum distance and its linear separation to the rest of the colors in the palette, in ascending order of frequency. To replace a color, we first select the $k = 3$ farthest colors from the rest of the palette as candidates using CIEDE2000 color distance. Then we evaluate each candidate c_a by considering its linear separation s_a, color frequency n_a, and minimal distance:

$$d_a = \min_{c_b \in C \setminus c_a} E(c_a, c_b)$$

The score of c_a is a weighted combination of the three terms:

$$S(c_a) = w_s s_a + w_n n_a + d_a \quad (2)$$

We approximate s_a in the u–v plane of the CIELUV color space as the minimal distance to the convex hull of the remaining colors of the palette [11]. Weights w_s and w_n balance the three terms. After analyzing these terms' ranges, we set w_s equal to 0.15 and w_n equal to the inverse of 0.03% of the image's pixel number by trial and error. The minimal distance d_C increases in each iteration. This scheme usually converges after several iterations; we stop after 20. Figure 3 shows an example.

The reference image is then approximated by the color palette C. A pixel is either deleted or represented by the most similar color in C if their CIEDE2000 color distance is smaller than a threshold (15 by default); see Fig. 2(d).

4 Color assignment

4.1 Goals

The given visualization contains N points $Q = \{q_1, \cdots, q_N\}$ with M different labels $L = \{l_1, \cdots, l_M\}$. The points with label l_i are denoted by Q_i. The aim of color assignment is to find a bijective mapping between the colors in palette C and the data labels, represented by a set of M pairs (i, a), where $l_i \in L$ and $c_a \in C$.

We formulate color assignment as an integer quadratic program, taking into consideration both unary and binary terms: for a candidate assignment (i, a), we measure how well label l_i matches color c_a; we also evaluate how compatible each pair of assignments (i, a) and (j, b) is. From a list of candidates—in our case, all possible assignments—we compute an optimal subset to form a one-to-one correspondence mapping that maximizes the quadratic score function; see Figs. 4(a) and 4(b).

4.2 Pairwise term

Classes with close-by points should be assigned with colors having large differences and adjacent in the reference if possible. Based on this intuition, we define the compatibility score of a pair of assignments (i, a) and (j, b) using three terms:

$$S((i, a), (j, b)) = D(Q_i, Q_j)A(c_a, c_b)E(c_a, c_b) \quad (3)$$

while $D(l_i, l_j)$ represents the closeness of two point sets, $E(c_a, c_b)$ is the color difference in the CIEDE2000 metric, and $A(c_a, c_b)$ measures the adjacency of two colors in the reference image.

Let $q_{i'}$ denote a point in Q_i. We define $D(Q_i, Q_j)$ as the summed closeness between $q_{i'}$ and Q_j:

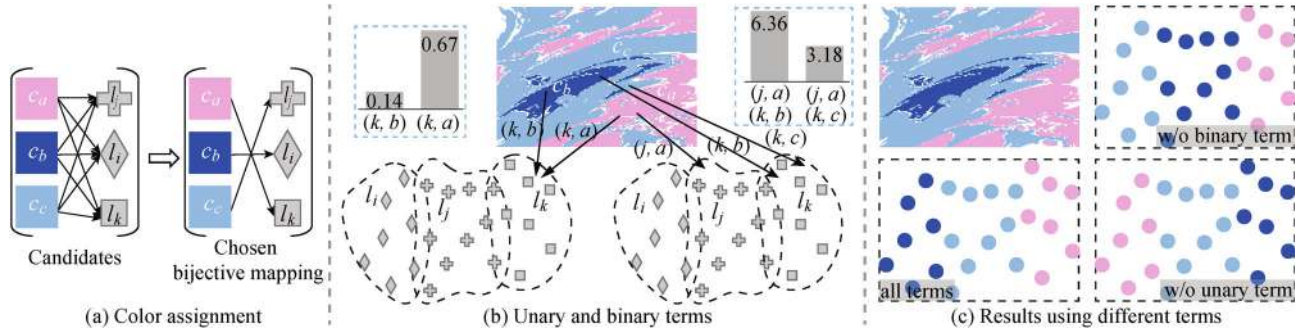


Fig. 4 Color assignment. (a) From a list of candidates, we aim to compute an optimal bijective mapping. Each symbol represents a class. (b) For clarity, the point set is repeated below; the virtual boundary of each class is shown by dashed lines. We measure the quality of a class–color assignment (the unary term) by comparing the locations of the class and the color; the bar chart on the top left corner shows the quality scores of two assignments. We also measure the quality of a pair of assignments (the binary term), considering the closeness of the two classes, the difference of their colors, and the adjacency of the two colors in the image; the bar chart on the top right shows the quality scores of two pairs. (c) The result using both terms is most similar to the reference.

$$D(Q_i, Q_j) = \sum_{i' \in Q_i} \frac{1}{|\mathcal{N}(i')|} \sum_{j' \in \Omega_{i'}} g(\|q_{i'} - q_{j'}\|) \quad (4)$$

This is inspired by *point distinctness* which measures the distinctness of a data point from its neighbours [7]. Here, $|\mathcal{N}(i')|$ represents the number of points in $q_{i'}$'s nearest neighbors $\mathcal{N}(i')$. Within them, the points with label l_j (the intersection of Q_j and $\mathcal{N}(i')$) is denoted $\Omega_{i'}$. We define the nearest neighbors $\mathcal{N}(i')$ using the α-shape graph as suggested by Lu et al. [9], which selects the closest neighbors of a given point within a certain radius. $g(\|q_{i'} - q_{j'}\|)$ is a distance-based function that varies inversely with Euclidean distance between point $q_{i'}$ and point $q_{j'}$; we use $g(d) = 1/d$. $D(Q_i, Q_j)E(c_a, c_b)$ measures the visual separation of point sets Q_i with color c_a and Q_j with color c_b.

The spatial adjacency of color c_a and c_b is computed based on the closeness score of their associated pixel sets. We assign large values to close-by colors in the image and small ones to faraway colors:

$$\tilde{A}(c_a, c_b) = |P_a| \frac{D(P_a, P_b)}{\sum_{1<c<|C|, c \neq a} D(P_a, P_c)} \quad (5)$$

where P_a and $|P_a|$ are the pixel group represented by color c_a and its number of pixels, respectively. We measure how much a color surrounds other colors, and divide the value by its number of pixels. As $\tilde{A}(c_a, c_b)$ is not symmetric, we define $A(c_a, c_b)$ by linearly mapping $\max(\tilde{A}, \tilde{A}^{\mathrm{T}})$ to the range of $[1, 2]$. We choose a narrow range for color adjacency as it is less important than point distinctness. When they conflict, the latter should dominate.

Figure 5 shows an example of assigning colors to a six-class pie chart when only considering the binary term. The closeness score of any two pie sectors $D(Q_i, Q_j)$ is either set to 1 if they are adjacent or 0 otherwise. When the color difference score is removed from Eq. (3), some adjacent colors with low discriminability are given to close-by sectors, which diminishes the chart's distinctness: see Fig. 5(b).

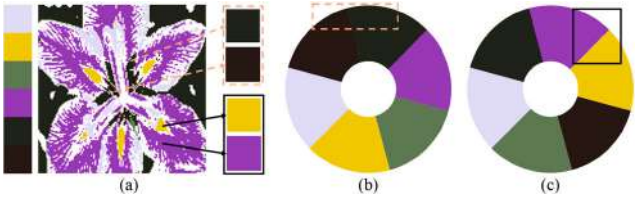


Fig. 5 Assigning color using only the binary term. (a) Some adjacent colors in the image, e.g., the two in the dotted pink box, are distinguishable, but with low discriminability. (b) Ignoring color differences in the binary term may cause assigning these colors to close-by classes, diminishing the chart's distinctness. (c) Considering the color difference can avoid such cases.

4.3 Unary term

The unary term measures the quality of a class–color assignment. We give a high score to an assignment (i, a) if the locations of points Q_i in the visualization are similar to the locations of color c_a in the reference image. We align the image to the visualization by their boundaries using scaling and translation, and then measure the distance between the point set Q_i and the pixel set P_a:

$$S((i, a)) = f(\tilde{d}_{\mathrm{H}}(Q_i, P_a)) \quad (6)$$

where $\tilde{d}_{\mathrm{H}}(Q_i, P_a)$ is the one-sided Hausdorff distance from Q_i to P_a, and $f(d)$ is inversely related to distance. The one-sided Hausdorff distance is preferable, as in our case we desire partial matching

of the point set to the pixel set. We use $f(d) = \exp(-8d^2)$ so that $f(d)$ declines rapidly from 1 to 0.1 as d grows from 0 to 0.5. Note that image size is normalized to $[0, 1]$. To speed up the computation of $\tilde{d}_{\mathrm{H}}$, we apply the Euclidean distance transform to binary images, each of which represents the pixels of a color. Figure 4(b) shows an example on the left.

4.4 Solution

Given a candidate set $\mathbb{F}$ of $|\mathbb{F}|$ assignments, a chosen subset is represented by an indicator vector $\boldsymbol{x} \in \{0, 1\}^n$ where chosen elements have a flag value of 1. The optimal subset is the solution of the objective function as in Eq. (7):

$$\boldsymbol{x}^* = \operatorname{argmax} \boldsymbol{x}^{\mathrm{T}} \boldsymbol{M} \boldsymbol{x} \quad \text{such that} \quad \boldsymbol{B}\boldsymbol{x} = \boldsymbol{1} \tag{7}$$

where $\boldsymbol{B}\boldsymbol{x} = \boldsymbol{1}$ represents the constraint that $\boldsymbol{x}$ is a bijective mapping. $\boldsymbol{M}$ is an $|\mathbb{F}| \times n$ matrix. The k-th and m-th assignments are denoted by (i, a) and (j, b), and we have

$$\boldsymbol{M}_{k,m} = \begin{cases} S((i,a),(j,b)), & k \neq m \\ w_u S(i,a), & k = m \end{cases} \tag{8}$$

where w_u is a weight to balance the binary and unary terms. We use $w_u = 0.2 \max_{k,m,k\neq m} \boldsymbol{M}_{k,m}$ in our setting. Though this is an NP-hard problem, an approximate solution can be found efficiently. We use the integer projected fixed point (IPFP) method [32] to solve it. Algorithm 1 shows the key steps for color assignment. Figure 4(c) shows how the result considering both unary and binary terms is most similar to the reference.

Note that if we ignore the color spatial information in the reference image, i.e., exclude the unary term and the color adjacency score ($A(c_a, c_b)$) in the binary term, the objective function is the same as that of Wang et al. [7]. Instead of using a genetic algorithm, we solve the problem by graph matching, which can quickly return an approximately optimal solution. By respecting both the spatial positions of colors and their spatial relationships, we can transfer colors' spatial information to the visualization.

Algorithm 1 Color assignment

Input: A reference image, a color palette C, and a visualization containing a point set with labels L.
Output: A bijective mapping between C and L, represented by a set of pairs (i, a) with $L_i \in L$ and $c_a \in C$.
Initialization: Generate the set $\mathbb{F}$ of all possible assignments.
for all assignment $(i, c) \in \mathbb{F}$ **do**
  Compute its quality $S((i, a))$ by Eq. (6);
  for all assignment $(j, b) \in \mathbb{F} \backslash (i, a)$ **do**
    Compute the compatibility score of two assignments $S((i, a), (j, b))$ by Eq. (3).
  end for
end for
Solve for the optimal indicator vector $\boldsymbol{x}^*$ using Eqs. (7) and (8).
Return the chosen subset of $\boldsymbol{F}$ whose flags from $\boldsymbol{x}^* = 1$.

5 Results and applications

Here we first describe implementation details. After that, we show coloring results for different types of charts. Lastly, we present two extensions in which user-specified constraints are added.

5.1 Implementation details

Although the core algorithm requires a point set with class information as input, we can easily apply the approach to other categorical visualizations, such as bar charts and pie charts, by sampling points inside the region of each class. We use Poisson disk sampling to generate samples from a blue noise distribution [33]. We sample about 300 points in each example; see the leftmost of Fig. 6. The randomness in generated samples may affect the class closeness score $D(Q_i, Q_j)$, so we sample 10 times and use the mean closeness score.

We implemented the algorithm using MATLAB. All experiments were run on a computer with an Intel Xeon Gold 5118 processor with 64 GB memory, and took less than 2 s. We designed an interactive interface using JavaScript to invoke the core algorithm; see Fig. 7. The interface works with SVG files as inputs and outputs. Users can select or upload an input visualization, and then upload or choose an image to set the visualization's color scheme. For more information, please see the video in the Electronic Supplementary Material (ESM).

5.2 Results

The class separation score for a coloring is $\sum_{(i,a)} \sum_{(j,b)} D(Q_i, Q_j) E(c_a, c_b)$. This measurement is also dependent on the points' distributions Q, so it is only meaningful to compare these scores for different color mappings of the same data. The spatial consistency between colors in reference images and those in the coloring is measured by two components. The position score of a color assignment

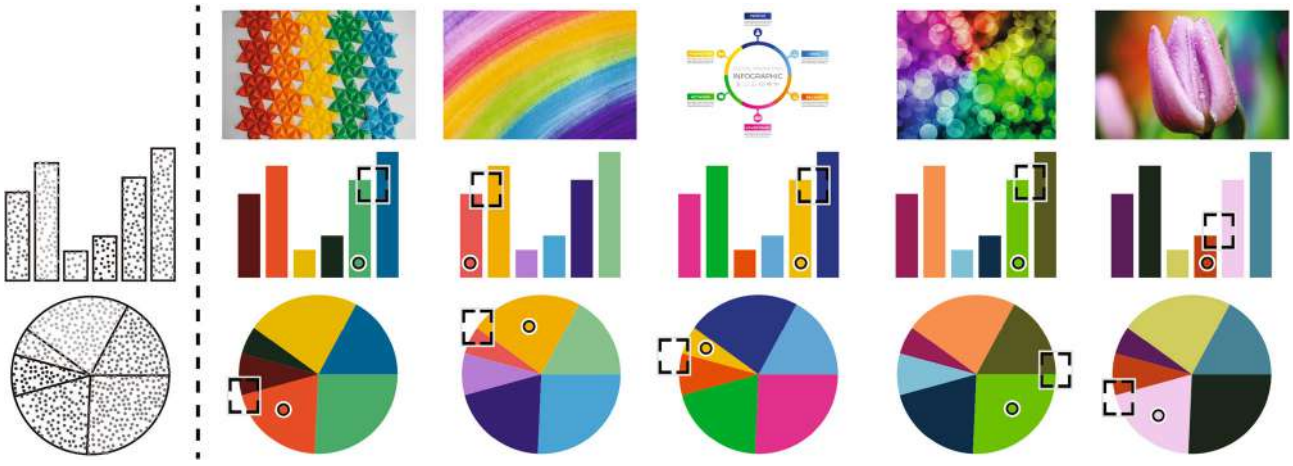


Fig. 6 Color mappings of a bar chart and a pie chart guided by five images. We generate points inside different bars and sectors by Poisson disc sampling. Regions with the highest spatial position similarity and with the highest adjacency similarities are highlighted by circles and rectangles, respectively.

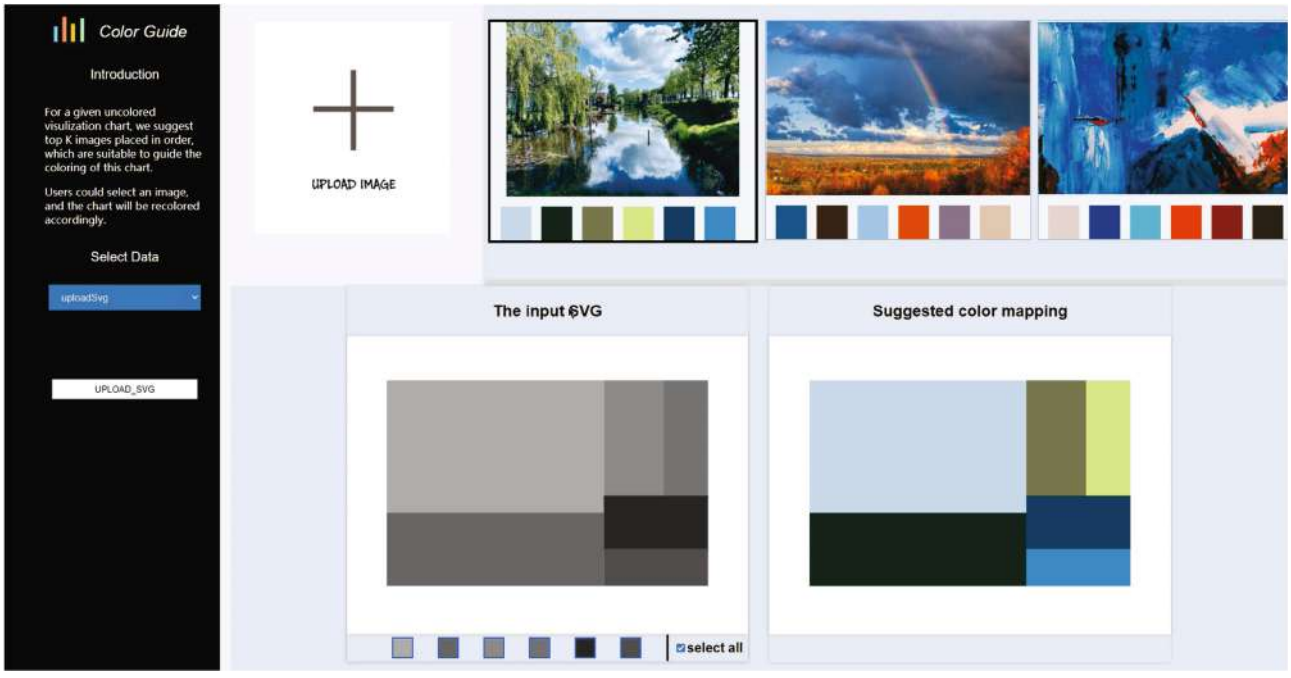


Fig. 7 Our interactive interface for color visualizations.

is defined by the unary term: $\sum_{(i,a)} S((i,a))$. The adjacency score is defined by the binary term: $\sum_{(i,a)} \sum_{(j,b)} D(Q_i, Q_j) A(c_a, c_b)$.

Figure 6 shows five different images and corresponding colorings of a bar chart and a pie chart with 6 classes. Most of the linear color arrangements in the first and second images are transferred to the corresponding colored bar charts. Most of the circular color arrangements in the third and fourth images also present in the colored pie charts.

All the input scatterplots except in Fig. 8 were generated from simulated categorical data. We randomly set each cluster's center, and then sampled points for each cluster using a Gaussian distribution, following Wang et al. [7].

Figure 9 shows examples of two scatterplots of 6 and 8 classes colored according to five different images, together with their extracted palettes. Note that an image's 6-color palette is not a subset of its 8-color palette, for increased distinctness. We can see that classes are easy to spot even though all points are represented by quite small dots. Although the

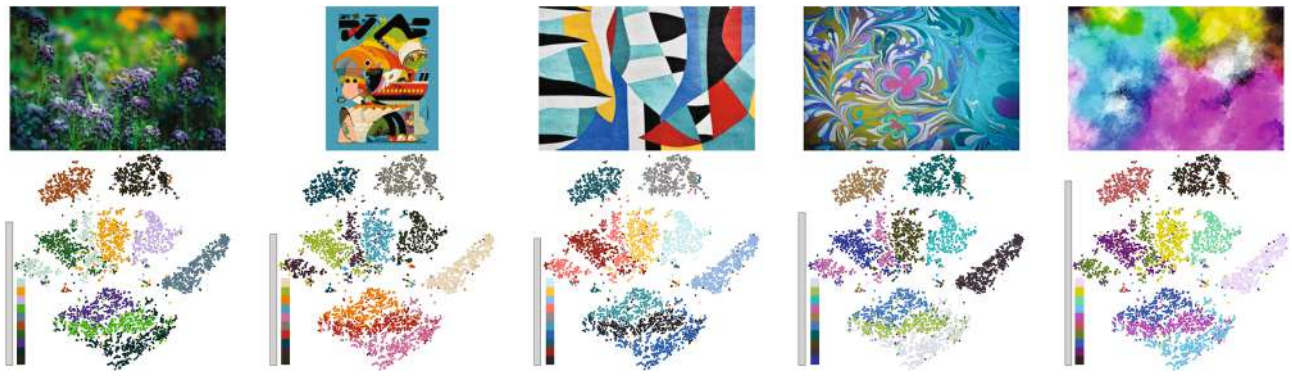


Fig. 8 Five color mappings of a 10-category scatterplot of the MNIST dataset. Class separation scores are shown by gray bars, together with the extracted palettes.

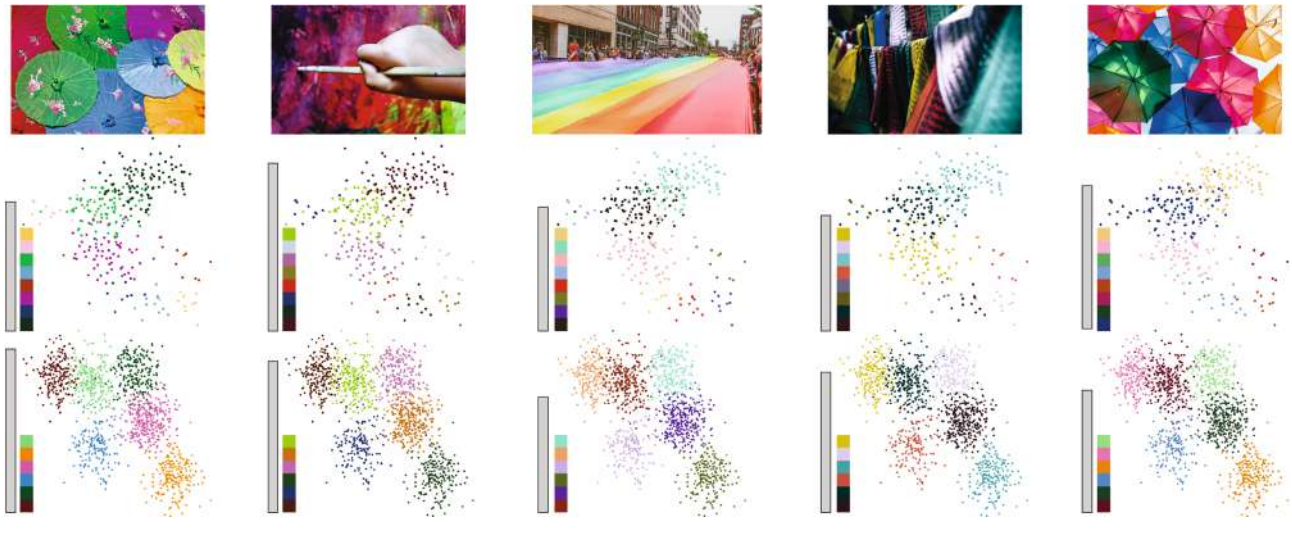


Fig. 9 Color mappings of two scatterplots with eight and six classes, guided by five images. Class separation scores are shown by gray bars, together with the extracted palettes.

two scatterplots have different class spatial structures, both still look similar to the corresponding reference images. For example, in the first column of Fig. 9, the yellow color is placed at the bottom right corner in both scatterplots, the same as its position in the reference; the adjacency of light and dark green colors in the image is also present in both scatterplots. Note the best among the five reference images in terms of class separation changes when the input alters.

We generated a scatterplot by projecting the MNIST dataset [34], which contains thousands of handwritten images of 10 digits, to the 2D plane using t-SNE [35]. Figure 8 shows the color mappings of this 10-class scatterplot using colorful images as guidance.

Our method can also suggest color mappings for webpage layouts and infographics. In webpage design, designers often use an image as the header and use its color scheme for the webpage. Figure 10 shows six suggested color mappings of a webpage layout according to different header images. The input is a web layout pattern specifying which connected components can be colored and which components must map to the same color.

In infographic design, images related to the infographic's topic are often used as references for the color setting. Figure 11 shows four suggested color mappings of an infographic according to different background images, generated using the interface in Fig. 7. Our method identifies all colors used in the input infographic and asks the users to mark the colors to replace. Then new color mappings are automatically generated according to the reference images. However, our method is limited to infographics without gradient colors. Although the first source image only has two dominant colors

Fig. 10 Color mappings of a web layout according to six images. In each case, part of the corresponding image is the page header.

Fig. 11 Color mappings of an infographic according to four images. In each case, the corresponding image in the center is also the background. Note that the light gray color in the second image is not selected because of its similarity to the infographic's background. The infographic is from Freepik.

(red-brown and green), our method still extracts a distinct 4-color palette, and the two different versions of red-brown are adjacent in the infographic, just as in the source. The presented results are able to nicely combine the given image with the webpage or infographic in a coherent and harmonious way.

5.3 Extensions

It is easy to extend the method to optimize the color mapping to any user-specified background color. Without modifying other part of the method, we simply let the minimal color distance of a palette C take into account the background color c_0:

$$d_C = \min_{0 \leqslant a,b \leqslant |C|, a \neq b} E(c_a, c_b)$$

The colorization result is thus tailored according to c_0. Three colorization results with black, white, and gray backgrounds are presented in Fig. 12. Note that the source image is not very colorful, but we are still able to extract colors distinct from each other and from the background in each case.

It is also easy to add user-defined constraints to the color assignment step. We only need to change the candidate set according to the given constraints. Figure 13 shows three color mappings of a treemap, in which only the 2 most dominant colors of each reference image are allowed to be assigned to the largest rectangle.

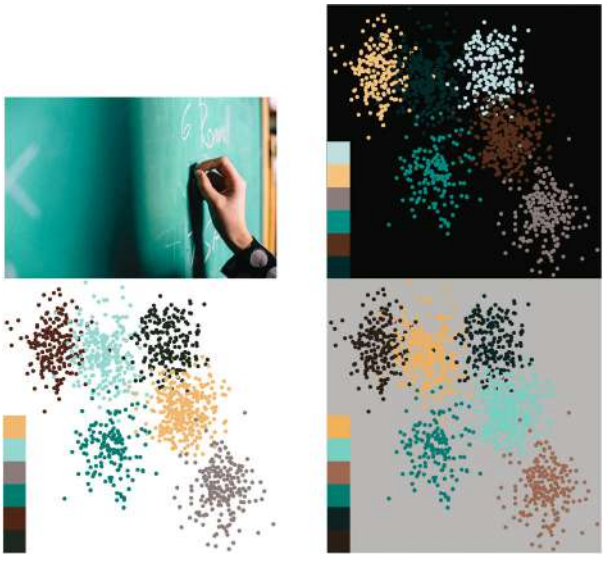


Fig. 12 Integrating a background color with the color mapping. The source image is shown in the top left. The extracted color palette changes for different backgrounds. Colors of each palette are in order of lightness values.

Fig. 13 Three color mappings of a treemap. We add a constraint that only the two most dominant colors are candidates for the largest rectangle.

6 Evaluation

We first evaluate our palette extraction step and compare it with existing methods. Then we evaluate our class assignment for visualization tasks by a controlled user study. A limitation is shown later.

6.1 Palette extraction evaluation

To evaluate the extracted palettes, we visually compare results of our approach with those from previous palette extraction methods [15, 18, 20, 24, 36]: see Fig. 14. The three test images have commonly been used in previous works for comparison, and palettes of other methods are taken from Zhang et al. [24]. Our model tends to extract palettes with large distances between colors, which are often easy-to-spot and vivid. Our results are most similar to those of Zhang et al. [24] as we use their initialization to select seed colors. However, our method produces palettes with lower similarities between different color components.

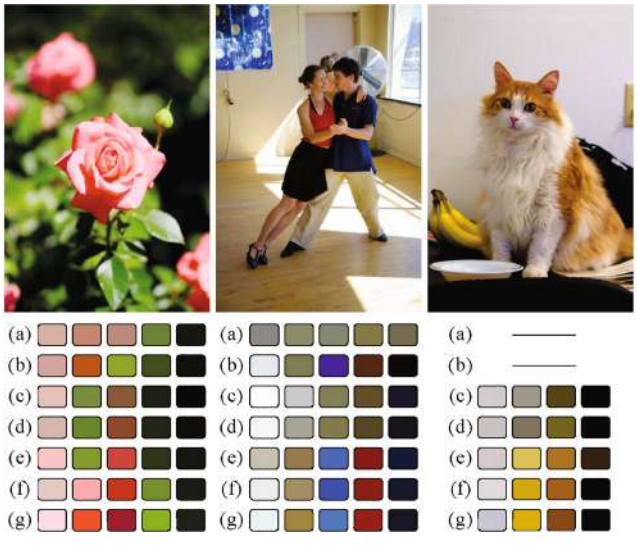


Fig. 14 Examples of images and their associated color palettes generated by O'Donovan et al. [15] (a), Lin and Hanrahan [18] (b), Shapira et al. [36] (c), k-means (d), Chang et al. [20] (e), Zhang et al. [24], and our method (g). Our results provide a more discriminable palette. For example, in the middle photo, our method chooses a brighter blue, which is more separable from the dark blue. Testing images and palettes for other methods are taken from Zhang et al. [24].

We next consider palette extraction for varying sizes of palette. The minimal distance of all color pairs in the extracted palette (d_C) decreases quite slowly when the number of colors increases; see the solid gray line for the left image and the dashed black one for the right in Fig. 16. Although the values of d_C depend on the given images, the changing trends are similar. Thus, we assume we can generate over 12 discriminable colors if they exist in the image.

6.2 Quantitative comparison

We quantitatively compare our method with two other palette extraction methods that automatically decide an optimal number of prominent colors in an image. One is a color extraction Web API provided by TinEye based on k-means clustering [37], chosen for its popularity. The other was proposed by Aksoy et al. [23] and exclusively selects colors that exist in the image. We chose this state-of-the-art method as it provides 100 testing images, as well as the extracted palettes. According to the two methods, 17 out of the 100 images include fewer than six distinct colors. We removed these and compared six-color palettes extracted from the remaining images. When the palettes extracted by other methods have over six colors, we selected six colors by greedily iteratively removing the color that most increases the minimal color distance of the remaining colors.

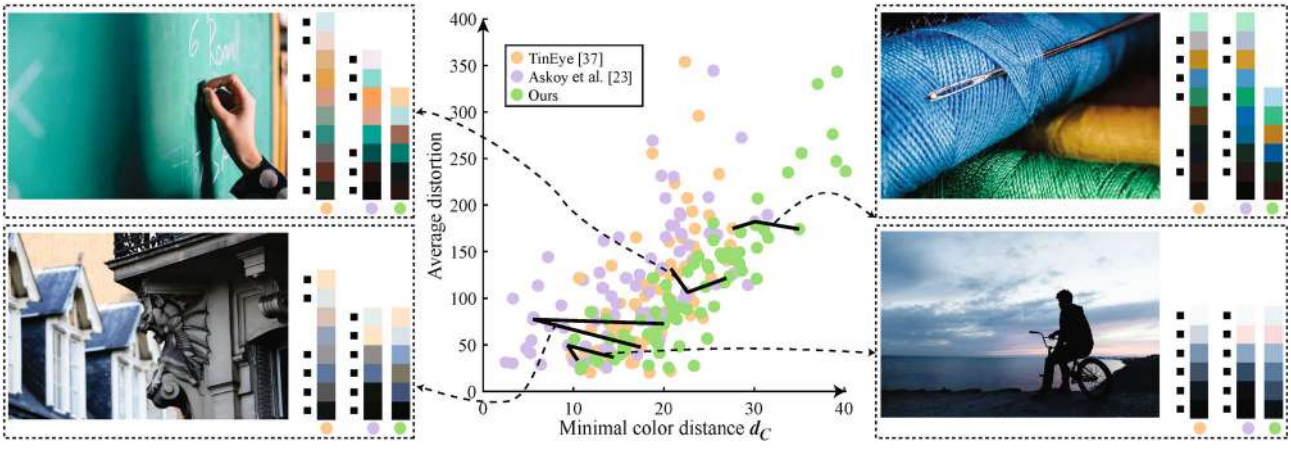


Fig. 15 Comparison with two common color palette extraction methods, TinEye [37] and Aksoy et al. [23]. Sometimes the number of colors in a palette from other methods is larger than six. When this happens, we select 6 colors by greedily removing the color that can best increase the minimal color distance of the remaining colors. We evaluate each palette using two measures: its minimal color distance d_C, and the average distortion between the quantized image and the original. Four example images and their corresponding palettes are shown. The minimal color distances of our results (green) are larger than for the others. Testing images are taken from Aksoy et al. [23].

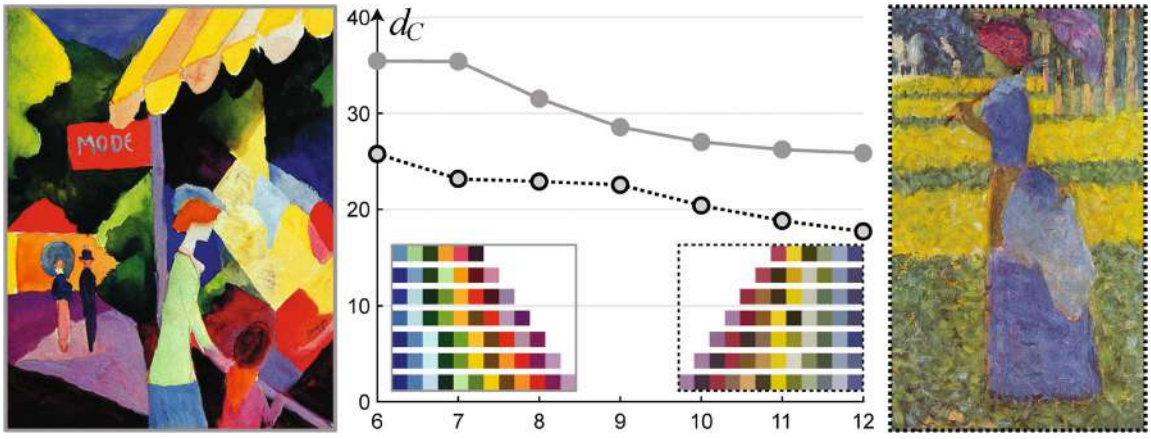


Fig. 16 Each line represents the minimal distances d_C of a palette set, extracted from an image conditioned on different number of colors. The colors inside each palette are in order of hue values. The two 12-color palettes in the bottom still have discriminable colors.

We evaluated each palette by two measures: its minimal color distance and the average distortion (MSE) between the quantized image and the original. Figure 15 shows these values for four examples. It can be seen that for most images, our palettes have similar average distortion but larger minimal color distances than others. Though we only compare six-color palettes, our method is more advantageous when the number of colors in the extracted palette increases, since others do not explicitly penalize similar colors. Thus, as the number of required colors increases, they are more likely to choose similar colors than our method.

We conducted a paired sample t-test to compare the minimal color distance scores. Results found a significant difference in the distance score comparing our method ($M = 23.06, \mathrm{SD} = 7.46$) to TinEye ($M = 18.84, \mathrm{SD} = 4.32$, $t(72) = 7.24, p < 0.0001$), and Aksoy's method ($M = 16.33, \mathrm{SD} = 7.25$, $t(72) = 9.50$, $p < 0.0001$). This indicates that our method can create a more distinct color palette than these others.

6.3 User study

6.3.1 Background

In general, it is difficult to define any ground truth for a color mapping task. The choice often depends on users' preference for the reference image, the aesthetic preferences of the user, and more. Thus, in our user evaluation, we do not examine subjective preferences, but rather focus on two main issues:

- *Discrimination.* In terms of discrimination, are the color mappings generated by our method comparable to state-of-the-art methods? To answer this, we performed several discrimination tasks in which we examined participants' perception of the number of classes in a given scatterplot. We hypothesized that the our method will be comparable in these discrimination tasks.
- *Similarity.* Is the color mapping generated by our method based on an image indeed more similar to that image than color mappings generated by non-spatial methods? To answer this, we performed several similarity tasks, in which we asked participants to judge which of two scatterplots is more similar to a source image in terms of color configuration. A neutral choice option was also provided. We hypothesized that our method will be perceived as more similar to the source image in the similarity tasks.

As far as we know, no existing method explicitly uses the spatial relationship between colors in an image for color mapping of a data visualization chart. Therefore we chose the recent color assignment method for optimizing perception of class separability [7] (referred to as *SepOpt*) for comparison. To ensure a fair comparison, the results of *SepOpt* were also solved by graph matching after removing the color spatial information from the formulation, instead of the original genetic algorithm. As our core algorithm deals with point sets, we performed the user studies on scatterplots.

6.3.2 Participants

We recruited 20 participants, out of which 14 were male and 6 were female, 18 were students and 2 were designers. Their ages ranged from 22 to 36 years old. None of them reported color vision deficiency. We paid each participant $6 for the two tasks. The

entire study took about 20–30 min. Each participant first provided demographic information, and then performed the discrimination tasks followed by the similarity tasks.

6.3.3 Discrimination task

In each trial, each participant was first shown a gray dot in the middle of the screen to calibrate the eye gaze, followed by a scatterplot. They were instructed to press a key when they knew the number of classes. When the key was pressed, the scatterplot disappeared. Then the participant chose the number of classes from five options (5–9). We recorded the time taken for each trial and counted the difference between the actual class number and the participant response as the error.

We used five multi-class scatterplots (1 of 6 classes, two of 7 classes, and 2 of 8 classes), generating them using the *SepOpt* method. First, a synthetic scatterplot of 307 points with 6 classes was created. Each class was placed according to a Gaussian distribution with strong overlaps between classes. Then we created other scatterplots by randomly selecting some points to form new classes. As the same points were used in all scatterplots, we rotated each by a randomized angle for variety.

We selected the top 20 from images used for Fig. 15 according to the minimal color distances d_C of their palettes. To avoid unwanted learning effects, we allowed each participant to see each color palette only once, using a between-subject layout. After shuffling the chosen images, we subdivided them into five equal-sized subsets, each assigned to a scatterplot. Thus there were two participant groups; every participant was assigned to either group G1 or G2 and completed 20 trials. For the first 10 images G1 saw *SepOpt* results and G2 saw our results. For the next 10 images, the groups were swapped. The corresponding colorized scatterplots were shown at half of their original resolution of 1048×1048 pixels with a small point size of 4 pixels.

The results are summarized in Fig. 17. The reported error is the sum of all errors made by a participant on a given method. Figure 17 shows the performance of participants on the two methods are similar in terms of both error and time. An independent samples t-test was conducted to compare error based on our method ($M = 2.95, \mathrm{SD} = 2.84$) and *SepOpt* ($M = 2.80, \mathrm{SD} = 2.31$). We observed

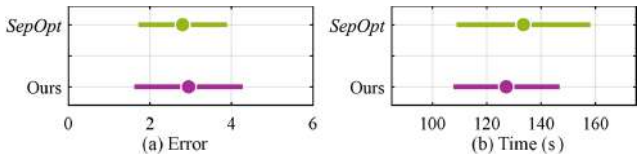


Fig. 17 Results of discrimination tasks, showing mean values and deviation as 95% CIs of user error and timing.

no significant differences between them ($t(38) = 0.18, p = 0.855$), suggesting that our approach is no worse than *SepOpt*. A second independent sample t-test was conducted to compare time with our method ($M = 127.19, \mathrm{SD} = 41.57$) and *SepOpt* ($M = 133.50, \mathrm{SD} = 52.36$). We found no significant differences in terms of time ($t(38) = -0.422, p = 0.675$). Figure 18(c) shows that the class separation of our results is slightly lower, but it does not affect the actual performance in counting tasks. A possible reason is that the small change in class separation score is not easy to spot when the value is high.

6.3.4 Similarity tasks

This task checked whether the color mappings generated by our method are perceived as more similar to the corresponding source images. We used six palettes extracted from six images, three of which are shown in Fig. 1. We used four different scatterplots with 6 or 8 classes, two of which were from Fig. 9 and the others were from *SepOpt*. Hence, the total number of trials was 4 (scatterplots) $\times$ 6 (palettes) = 24. In each trial, two scatterplots generated by different methods using the same image were shown in pairs. To avoid bias, we randomised the locations for the two methods. An example is shown in Fig. 19(top).

We asked participants to compare them with the reference image on a 5-point Likert scale: *The left is much more similar*, *The left is slightly more similar*, *No difference*, *The right is slightly more similar*, and *The right is much more similar*. The

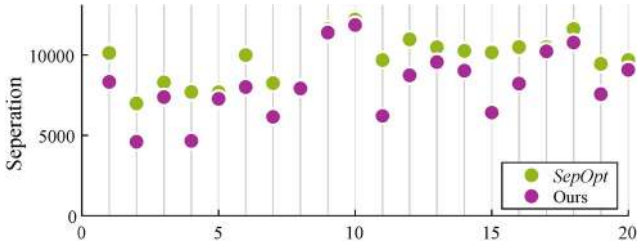


Fig. 18 Class separation of the 40 scatterplots in counting tasks.

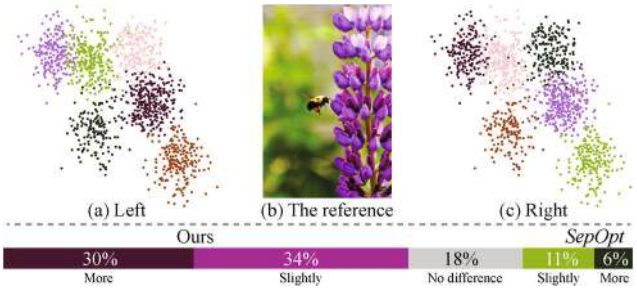


Fig. 19 Top: example of similarity tasks. Participants were instructed to compare the similarity of two colorings related to a reference image. In this example, our result is shown on the left. Bottom: result summarised as a bar. Most participants thought our results resemble the source images better.

colored scatterplots were shown at a resolution of 1048×1048 pixels, with a larger point size of 6 pixels, as similarity tasks should be insensitive to the point size.

The participant's choices are summarized in Fig. 19(bottom). Overall, in 64% of the trials, participants thought our results were more similar to the source image, while only in 17% of the trials did participants think *SepOpt* was more similar. We conclude that our method manages to bring the spatial relationship of the source image to the target visualization. The scores of the spatial consistency between colors in reference images and in the scatterplots, shown in Fig. 20, also indicate our color mappings can better preserve the color spatial relations.

6.4 Limitations

Our method is built on an input reference image, and can thus fail to produce a good color mapping when the source image is unsuitable: more specifically, when it lacks sufficient distinct colors. Figure 21(left) shows a case in which some classes are difficult to discriminate. Another color mapping which makes

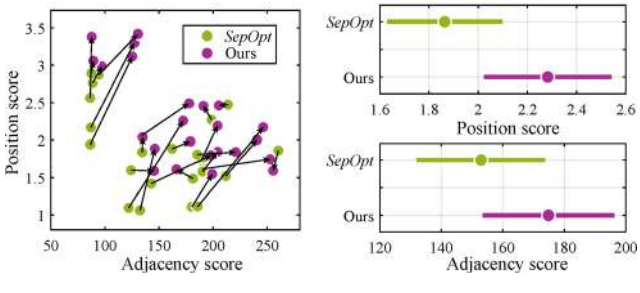


Fig. 20 Left: color spatial consistency between reference images and scatterplots used for similarity tasks in terms of adjacency and position. Right: mean values and deviation as 95% CIs of position and adjacency measurements.

Fig. 21 Left: an undesirable color mapping that fails to distinguish classes from each other. Right: a good one. The respective class separation scores are 11,555 and 19,477. The quality of our generated color mappings depends on the reference images.

classes distinguishable and has a much larger class separation score is shown on the right. We could easily detect such failures by examining the minimal color distance d_C of extracted palettes and the class separation scores. When d_C is smaller than a threshold such as 10, or the class separation score is significantly lower than the scores of other color mappings, we could warn users and suggest changing the reference.

Furthermore, some image features, such as saliency, foreground, and background, are ignored during color palette extraction. Our algorithm also has several parameters that need to be set: weights for linear separation, color frequency, and minimal distance in the palette extraction, and weights to balance the binary and unary terms. Changing these weights may slightly change the results. However, in all presented experiments and results, we used constant values.

Finally, we evaluated the similarity between colored visualizations and their corresponding references only in terms of color spatial consistency.

7 Conclusions

We have presented an automatic image-guided approach to produce color mappings for categorical visualizations that are visually similar to source images while allowing visual discrimination of classes. The key idea is to extract a discriminable and representative color palette from the reference image using farthest-point sampling, and assign its colors to classes in the visualization considering both class discrimination and spatial distribution of colors in the image. We show various results, evaluate the approach numerically, and perform a controlled user study to examine the perceptual effect of the method.

Some image features, such as the saliency, foreground, and background, are ignored during color palette extraction. There is a lot left for further exploration. Now we exploit images to give graphical

marks suitable colors. In the future, we could explore image information to set other channels of graphical marks, such as gradients, non-linear color triads [38], outline colors, and even mark shapes. In addition, we would also like to extend our method to support the coloring of other visualizations such as maps.

Acknowledgements

We thank the reviewers for their valuable comments and constructive suggestions. This work was supported in parts by National Natural Science Foundation of China (U2001206, 61872250), GD Talent Program (2019JC05X328), GD Natural Science Foundation (2020A0505100064, 2021B1515020085), DEGP Key Project (2018KZDXM058), and Shenzhen Science and Technology Key Program (RCJC20200714114435012, JCYJ20210324120213036).

Declaration of competing interest

The authors have no competing interests to declare that are relevant to the content of this article.

Electronic Supplementary Material

Supplementary material is available in the online version of this article at `https://doi.org/10.1007/s41095-021-0258-0`.

References

[1] Behrisch, M.; Blumenschein, M.; Kim, N. W.; Shao, L.; El-Assady, M.; Fuchs, J.; Seebacher, D.; Diehl, A.; Brandes, U.; Pfister, H.; Schreck, T.; Weiskopf, D.; Keim, D. A. Quality metrics for information visualization. *Computer Graphics Forum* Vol. 37, No. 3, 625–662, 2018.

[2] Silva, S.; Sousa Santos, B.; Madeira, J. Using color in visualization: A survey. *Computers & Graphics* Vol. 35, No. 2, 320–333, 2011.

[3] Harrower, M.; Brewer, C. A. ColorBrewer.org: An online tool for selecting colour schemes for maps. *The Cartographic Journal* Vol. 40, No. 1, 27–37, 2003.

[4] Gramazio, C. C.; Laidlaw, D. H.; Schloss, K. B. Colorgorical: Creating discriminable and preferable color palettes for information visualization. *IEEE Transactions on Visualization and Computer Graphics* Vol. 23, No. 1, 521–530, 2017.

[5] Lin, S.; Fortuna, J.; Kulkarni, C.; Stone, M.; Heer, J. Selecting semantically-resonant colors for data visualization. *Computer Graphics Forum* Vol. 32, No. 3pt4, 401–410, 2013.

[6] Kim, H. R.; Yoo, M. J.; Kang, H.; Lee, I. K. Perceptually-based color assignment. *Computer Graphics Forum* Vol. 33, No. 7, 309–318, 2014.

[7] Wang, Y.; Chen, X.; Ge, T.; Bao, C.; Sedlmair, M.; Fu, C. W.; Deussen, O.; Chen, B. Optimizing color assignment for perception of class separability in multiclass scatterplots. *IEEE Transactions on Visualization and Computer Graphics* Vol. 25, No. 1, 820–829, 2019.

[8] Lin, S.; Ritchie, D.; Fisher, M.; Hanrahan, P. Probabilistic color-by-numbers. *ACM Transactions on Graphics* Vol. 32, No. 4, Article No. 37, 2013.

[9] Lu, K. C.; Feng, M.; Chen, X.; Sedlmair, M.; Deussen, O.; Lischinski, D.; Cheng, Z.; Wang, Y. Palettailor: Discriminable colorization for categorical data. *IEEE Transactions on Visualization and Computer Graphics* Vol. 27, No. 2, 475–484, 2021.

[10] Schlömer, T.; Heck, D.; Deussen, O. Farthest-point optimized point sets with maximized minimum distance. In: Proceedings of the ACM SIGGRAPH Symposium on High Performance Graphics, 135–142, 2011.

[11] Healey, C. G. Choosing effective colours for data visualization. In: Proceedings of the 7th Annual IEEE Visualization, 263–270, 1996.

[12] Trumbo, B. E. A theory for coloring bivariate statistical maps. *The American Statistician* Vol. 35, No. 4, 220–226, 1981.

[13] Zeileis, A.; Hornik, K.; Murrell, P. Escaping RGBland: Selecting colors for statistical graphics. *Computational Statistics & Data Analysis* Vol. 53, No. 9, 3259–3270, 2009.

[14] Bartram, L.; Patra, A.; Stone, M. Affective color in visualization. In: Proceedings of the CHI Conference on Human Factors in Computing Systems, 1364–1374, 2017.

[15] O'Donovan, P.; Agarwala, A.; Hertzmann, A. Color compatibility from large datasets. *ACM Transactions on Graphics* Vol. 30, No. 4, Article No. 63, 2011.

[16] Kita, N.; Miyata, K. Aesthetic rating and color suggestion for color palettes. *Computer Graphics Forum* Vol. 35, No. 7, 127–136, 2016.

[17] Fang, H.; Walton, S.; Delahaye, E.; Harris, J.; Storchak, D. A.; Chen, M. Categorical colormap optimization with visualization case studies. *IEEE Transactions on Visualization and Computer Graphics* Vol. 23, No. 1, 871–880, 2017.

[18] Lin, S.; Hanrahan, P. Modeling how people extract color themes from images. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, 3101–3110, 2013.

[19] Poco, J.; Mayhua, A.; Heer, J. Extracting and retargeting color mappings from bitmap images of visualizations. *IEEE Transactions on Visualization and Computer Graphics* Vol. 24, No. 1, 637–646, 2018.

[20] Chang, H. W.; Fried, O.; Liu, Y. M.; DiVerdi, S.; Finkelstein, A. Palette-based photo recoloring. *ACM Transactions on Graphics* Vol. 34, No. 4, Article No. 139, 2015.

[21] Tan, J. C.; Lien, J. M.; Gingold, Y. Decomposing images into layers via RGB-space geometry. *ACM Transactions on Graphics* Vol. 36, No. 4, Article No. 7, 2017.

[22] Tan, J. C.; Echevarria, J.; Gingold, Y. Efficient palette-based decomposition and recoloring of images via RGBXY-space geometry. *ACM Transactions on Graphics* Vol. 37, No. 6, Article No. 262, 2018.

[23] Aksoy, Y.; Aydin, T. O.; Smolić, A.; Pollefeys, M. Unmixing-based soft color segmentation for image manipulation. *ACM Transactions on Graphics* Vol. 36, No. 4, Article No. 19, 2017.

[24] Zhang, Q.; Xiao, C. X.; Sun, H. Q.; Tang, F. Palette-based image recoloring using color decomposition optimization. *IEEE Transactions on Image Processing* Vol. 26, No. 4, 1952–1964, 2017.

[25] Nguyen, R. M. H.; Price, B.; Cohen, S.; Brown, M. S. Group-theme recoloring for multi-image color consistency. *Computer Graphics Forum* Vol. 36, No. 7, 83–92, 2017.

[26] Phan, H. Q.; Fu, H. B.; Chan, A. B. Color orchestra: Ordering color palettes for interpolation and prediction. *IEEE Transactions on Visualization and Computer Graphics* Vol. 24, No. 6, 1942–1955, 2018.

[27] Setlur, V.; Stone, M. C. A linguistic approach to categorical color assignment for data visualization. *IEEE Transactions on Visualization and Computer Graphics* Vol. 22, No. 1, 698–707, 2016.

[28] Lee, S.; Sips, M.; Seidel, H. P. Perceptually driven visibility optimization for categorical data visualization. *IEEE Transactions on Visualization and Computer Graphics* Vol. 19, No. 10, 1746–1757, 2013.

[29] Bohra, M.; Gandhi, V. ColorArt: Suggesting colorizations for graphic arts using optimal color-graph matching. In: Proceedings of the Graphics Interface, 95–102, 2020.

[30] Sharma, G.; Wu, W. C.; Dalal, E. N. The CIEDE2000 color-difference formula: Implementation notes, supplementary test data, and mathematical observations. *Color Research & Application* Vol. 30, No. 1, 21–30, 2005.

[31] Stone, M.; Szafir, D. A.; Setlur, V. An engineering model for color difference as a function of size. In: Proceedings of the Color and Imaging Conference, 253–258, 2014.

[32] Leordeanu, M.; Hebert, M.; Sukthankar, R. An integer projected fixed point method for graph matching and MAP inference. In: Proceedings of the 22nd International Conference on Neural Information Processing Systems, 1114–1122, 2009.

[33] Bridson, R. Fast Poisson disk sampling in arbitrary dimensions. In: Proceedings of the ACM SIGGRAPH Sketches, 22–es, 2007.

[34] Lecun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE* Vol. 86, No. 11, 2278–2324, 1998.

[35] Van der Maaten, L.; Hinton, G. Visualizing data using t-SNE. *Journal of Machine Learning Research* Vol. 9, No. 86, 2579–2605, 2008.

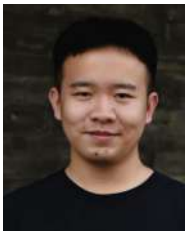
[36] Shapira, L.; Shamir, A.; Cohen-Or, D. Image appearance exploration by model-based navigation. *Computer Graphics Forum* Vol. 28, No. 2, 629–638, 2009.

[37] TinEye. Color extraction. 2021. Available at https://labs.tineye.com/color/.

[38] Shugrina, M.; Kar, A.; Fidler, S.; Singh, K. Nonlinear color triads for approximation, learning and direct manipulation of color distributions. *ACM Transactions on Graphics* Vol. 39, No. 4, Article No. 97, 2020.

Qian Zheng received her doctoral degree in computer science from Shenzhen Institutes of Advanced Technology (SIAT), Chinese Academy of Sciences, in 2015. She is now a lecturer at Suzhou University of Science and Technology. Her interests include computer graphics and information visualization.

Min Lu is an assistant professor at Shenzhen University. She received her B.Sc. degree in computer engineering from Beijing Normal University, China, in 2011, and received her Ph.D. degree in computer science from Peking University in 2017. Her major research interests include visualization methodology and visual analytics.



Sicong Wu received his bachelor degree in computer science from Southwest University of Science and Technology in 2020. He is currently working towards a master degree in Shenzhen University. His research interests include visualization and visual analytics.

Ruizhen Hu is an associate professor at Shenzhen University. She received her Ph.D. degree from Zhejiang University. Previously, she spent two years visiting Simon Fraser University, Canada. Her research interests are in computer graphics, with a recent focus on applying machine learning to advance the understanding and generative modeling of visual data including 3D shapes and indoor scenes. She is an editorial board member of *The Visual Computer* and *IEEE CG & A*.

Joel Lanir is a senior lecturer and faculty member in the Information Systems Department at the University of Haifa, Israel, where he heads the Human–Computer Interaction Lab. He received his Ph.D. degree in computer science from the University of British Columbia in 2009. His research interests lie in the fields of human–computer interaction, ubiquitous computing, and information visualization.

Hui Huang is a distinguished professor at Shenzhen University, where she directs the Visual Computing Research Center. She received her Ph.D. degree in applied math from The University of British Columbia in 2008. Her research interests span computer graphics, computer vision, and visualization. She is currently a senior member of IEEE/ACM/CSIG, a distinguished member of CCF, and is on the editorial boards of *ACM TOG* and *IEEE TVCG*.