Color Restoration with Visual Feature Code

Shusuke Kibune[1] and Shigeru Kuriyama[1]

[1]Toyohashi University of Technology, Toyohashi, Aichi, Japan

***Abstract*— We propose a method for restoring original colors of photographed images by detecting visually coded color features of original images. A color transfer technology that refers a small amount of color features is introduced for correcting colors, and visual code technique with colored frames is proposed for embedding the color features in the same image. We demonstrate the superior performance of our color restoration technique to existing ones and the practicability of our visual code, through the experiments for many kinds of images with a digital camera on a mobile phone.**

***Index Terms*— Color Restoration, Color Transfer, Image Processing, Visual Data Coding.**

I. Introduction

Color restoration technique is important in design and medical care because many applications in these fields require the accuracy in color representation. It usually introduces a color chart for adjusting white or color balances of images. The color chart is referenced as measures of size and exposure for diminishing the changes of scale and color depending on the conditions in photographing. This correction, however, essentially has no capability in recovering the original color of the image because it predicts the overall change of colors from those observed in the color chart. Moreover, taking a color chart together within a scene is impermissible for ordinary photography.

We here introduce the color transfer method for accurately restoring original colors. Instead of using a color chart, our method directly refers the information about original color features that are compactly coded and painted along the frame of an image. Our color restoration is therefore regarded as the integrated technique of a color transfer and feature coding.

Color transfer can change the atmosphere of an image (source) by transferring its color impression into the other image (target) with a small amount of features. On the other hand, color-coding technique had the limitation on the size of embeddable data. For these reasons, we make the best effort to minimize the amount of color features and compactly represent them as visual code, with which the color transfer technique can restore images in sufficient quality.

S. Kibune is with Toyohashi University of Technology, Aichi, Toyohashi, Japan (e-mail: kibune@val.cs.tut.ac.jp).

S. Kuriyama is with Toyohashi University of Technology, Aichi, Toyohashi, Japan (e-mail: kuriyama@cs.tut.ac.jp).

II. Related Works

A. Color Transfer

Color transfer technology is a main part of our integrated methodology; the quality of restoration totally depends on this process.

The one of the simplest methods of color transfer was proposed by Reinhard et. al. [1] that transfer colors in lαβ color space [2]. The lαβ space is designed on the basis of human visual perception and can be uniquely converted from RGB color space. Jing et. al. [3] introduced eigen color space obtained by converting pixel values with eigen values and vectors through principal component analysis (or PCA) in RGB color space. The axes of these color spaces are mutually uncorrelated and can therefore create high-quality images.

Color transfer techniques also can be regarded as a colorization to grayscale images. Levin et. al. [4] proposed a colorization method from the colored strokes drawn on selected regions. This propagates the color used in the strokes by analyzing the feature of the corresponding grayscale image with numerical optimization.

Pouli et. al. [5] proposed a color transfer using transformation of color histogram. This constructs the histogram for each component in CIELab color space and matches the histogram of a source image to target's one. This can flexibly change the impression or atmosphere of an image by controlling the degree of histogram matching. This process is efficient because most computations are executed on operations with respect to the histograms, and the transferred colors always reside in the matched histogram, by which original color distributions of the transferred image is safely preserved.

B. Data Coding

Arbitrary data can be embedded into images by introducing data hiding techniques that are roughly categorized into two types: visible code and invisible one. Both techniques modulate colors or other feature of source image for embedding coded data, sacrificing the quality of modulated images. Our method aims to restore the original colors of images as accurate as possible, and we therefore avoid such degradation caused in hiding data.

The other data-coding techniques introduce some visual representation of embedded data. The most popular methodology uses 2D-gridded matrix-formed black-and-white patters. Some 2D bar codes such as QR-code are now widely utilized as visual code by which mobile phones can read URL information from the captured image. The 2D bar code can robustly read

ISBN: 978-602-8944-11-3

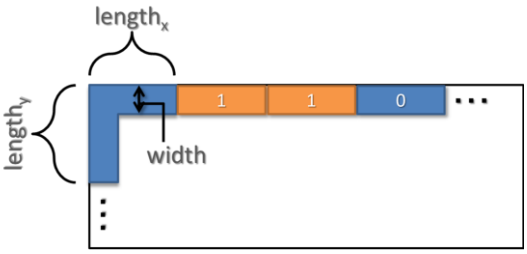

Fig. 1. Coding method.

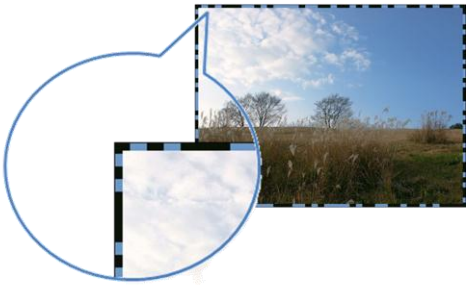

Fig. 2. Example of data coding.

embedded code and ensures sufficient amount of data (or payload); they, however, occupy a large area for their visual representation. Moreover, we may separately capture two images for target image and visual code, which damages usability for ordinary usages. For overcome these defects, we propose a new type of data-coding technique requiring only small space for its visual representation.

Miyaoku et. al. proposed a ring-shaped colored code, called C-Band [6], as a replacement of QR-code [7] or Color code [8]. This visual code converts embedded data into color variations painted along the looped and curved bar. The resulting color image is visually conspicuous and damages the appearance of a surrounded image because the painted colors span the full range of chromaticity.

III. Restoration Mechanism

A. Color Restoration

We first introduce the method [3] for converting the color components of RGB space into the uncorrelated components by using PCA.

The color of the *i*-th pixel, denoted by $\boldsymbol{c}_i = [R_i, G_i, B_i]^T$ is converted into

$$\boldsymbol{e}_i = \boldsymbol{D}^{-1/2}\boldsymbol{U}^T\boldsymbol{d}_i = \boldsymbol{D}^{-1/2}\boldsymbol{U}^T(\boldsymbol{c}_i - \bar{c}), \qquad (1)$$

where $\bar{c}$ denotes the average of all n pixels, $\boldsymbol{U} = [\boldsymbol{u}_1\ \boldsymbol{u}_2\ \boldsymbol{u}_3]$ and $\boldsymbol{D} = \mathrm{diag}(\lambda_1, \lambda_2, \lambda_3)$ represent the eigen vectors and eigen values, respectively, obtained by using PCA computed with variance-covariance matrix $\boldsymbol{M} = \sum_{i=0}^{n} \boldsymbol{d}_i\boldsymbol{d}_i^T/n$, $\boldsymbol{d}_i = \boldsymbol{c}_i - \bar{c}$, and superscript T denotes transpose. On the other hand, the inverse mapping from uncorrelated components $\boldsymbol{e}_i$ into the RGB color space is given as

$$\hat{\boldsymbol{c}}_{\boldsymbol{i}} = \hat{\bar{c}} + \widehat{\boldsymbol{U}}\widehat{\boldsymbol{D}}^{1/2}\boldsymbol{e}_i. \qquad (2)$$

The color transfer predicts original colors $\hat{\boldsymbol{c}}_{\boldsymbol{i}}$ by Eq. (2) with $\boldsymbol{e}_i$ computed by Eq. (1) for the photographed image and with $\widehat{\boldsymbol{D}}, \widehat{\boldsymbol{U}}, \hat{\bar{c}}$ that are pre-computed by Eq. (1) for the original image and embedded in a visual code.

B. Color Feature Compression

In coding color features, the values of $\widehat{\boldsymbol{D}}, \widehat{\boldsymbol{U}}, \hat{\bar{c}}$ should be compactly represented in order to minimize the size of embedded data. Here we focus on the orthogonality of three eigen vectors $[\boldsymbol{u}_1\ \boldsymbol{u}_2\ \boldsymbol{u}_3]$ that can be compactly represented as the rotations of a coordinate system. We can represent the eigen vectors as one rotational quaternion whose four components are quantized into 1000 levels that can be represented with 10 bits and 1 signed bit. Consequently, recording the eigen vectors requires 4 x 11 = 44 bits in total. Moreover, the theory of Gerschgorin claims that the number of eigen values does not exceed 2^{15}, which means that their integer representation requires 15 bits at the largest case, which requires 3 x 15 = 45 bits for recording all eigen values. Finally, the average of color $\hat{\bar{c}}$ is given by 8 x 3 = 24 bits, and we can reduce the total of the data required for color transfer to 44 + 45 + 24 = 113 bits ≈ 15 bytes.

C. Visual Code

This section explains about the technique of visually coding the abovementioned color features. This embeds the data by drawing colored patterns along the frame surrounding the image. We first subdivide an image into 32 x 32 blocks, and computed averaged colors inside them. We next select the pair of the averaged colors that have the longest distance from each other, and these two colors are painted along the rectangular frame of the image. We code the data of 42 bits along each thick edge of the frame as shown in Fig. 1, where two end bits are referred for calibrating colors, and 40 bits actually represent color features. As a result, 160 bits are embedded along the frame in total.

The width and length of each edge segment per bit are given as

$$width = \min(I_w, I_h)/60, \qquad (3)$$

$$length_w = I_w/N_{code}, \qquad (4)$$

$$length_h = I_h/N_{code}, \qquad (5)$$

where I_w, I_h represent the width and height of the image, $N_{code} = 42$, and $length_w$ and $length_h$ are the length of the segments along horizontal and vertical directions, respectively. Figure 2 shows the example of this visual code.

D. Data Detection

We next explain how to detect embedded data from the image captured with a digital camera. We first try to extract rectangular area by detecting its corners, and compute projective transformation of the image from the shape of the rectangle (see Fig. 3).

We then compute the averages $\mu_{k=r,g,b}$ and variations $\sigma^2_{k=r,g,b}$ of the pixel colors while traversing the edges of the frame as follows:

$$\mu_k = \sum_{i=0}^{I_w,I_h} \sum_{j=0}^{width} \frac{c_k(i,j)}{width \times I_w, I_h}, \quad (6)$$

$$\sigma_k^2 = \sum_{i=0}^{I_w,I_h} \sum_{j=0}^{width} \frac{(c_k(i,j) - \mu_k)^2}{width \times I_w, I_h}, \quad (7)$$

$$k = r, g, b.$$

We then select the component of RGB channel that has the biggest color variation; argmax $\sigma_{k=r,g,b}$ and make histogram of the component for all visited pixels along the frame. The threshold for binarization is then determined by using discrimination analysis [10] with this histogram. The embedded binary code is extracted by mapping 0 and 1 value to the corresponding binary colors, where the 0-mapped color is embedded at the corner points of the frames for calibrating the color.

IV. Experimental Results

A. Quality in color restorations

We compared the quality of our color restoration against those of existing color correction with color chart which is called QPcard [9]. We estimated the quality of restoration by using the value of PSNR (Power Signal Noise Ratio) where the original images are used as true signals. To make qualitative comparison strictly in the same condition, we experimentally corrected the images that include color chart, as shown in Fig. 4. The images in the first (right-most) row are produced by photographing original samples in the middle-right row. The restored images with our method are demonstrated in the middle-left row, and the resulting images of QPcard are shown in the final (left-most) row. Notice that our restoration technique requires no color chart. Table 1 presents the quantitative evaluation of the corrections for 6 images. The PSNR is computed with the averaged colors of every blocks consisting of 3 by 3 pixels, in order to reduce the errors caused by geometric distortions in projective transformations. Photographed images were generated by taking a photo of 2M pixels with the digital camera mounted on mobile phone of Xperia DoCoMo. Our restoration technique has larger PSNR for all images than those obtained by QPcard. This means that our method can recover the original images in higher accuracy.

$$PSNR = 10\, log_{10}\left(\frac{255^2}{MSE}\right), \quad (8)$$

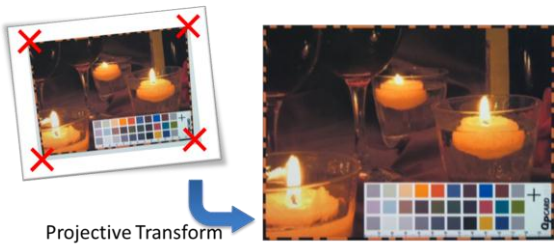

Fig. 3. Example of projective transform.

TABLE I
COMPARATIVE RESULTS OF COLOR CORRECTION WITH PSNR.

Image	photographed image [dB]	Result of our method [dB]	Result of QPcard [dB]
Baby	25.979	33.890	26.006
Candle	26.943	31.490	26.864
Dishes	18.902	29.716	20.342
Oranges	25.856	28.539	26.491
Cosmos	24.335	30.689	25.949
Sky	22.270	31.260	24.584

TABLE II
ACCURACY IN DATA DETECTION OF OUR VISUAL CODE.

Name	Error bit rate [%]	Failure rate of corner detection [%]	Perfect detection rate [%]
Baby	0.63	0	70
Candle	3.94	10	90
Dishes	7.38	30	50
Oranges	1.63	0	60
Cosmos	0.81	0	70
Sky	0.38	0	90

$$MSE = \sum_{i=0}^{\lceil \frac{I_w}{3} \rceil - 1} \sum_{j=0}^{\lceil \frac{I_h}{3} \rceil - 1} \sum_{k=r,g,b} \frac{\left(B(c_k,i,j) - B(c'_k,i,j)\right)^2}{\lceil I_w/3 \rceil \times \lceil I_h/3 \rceil \times 3}, \quad (9)$$

$$B(c,i,j) = \sum_{di=0}^{2} \sum_{dj=0}^{2} \frac{c(i \times 3 + di, j \times 3 + dj)}{9}. \quad (10)$$

B. Accuracy in data detection

We examined the error bits in detecting data using our visual code for evaluating its accuracy by taking 10 images on the same condition used in evaluating the color restoration. Table II shows the averages of error bit rates for 10 images, where the rate are computed by dividing the number of incorrectly detected bits by the total number of embeddable bits. It also shows the rates of failed detections of image corners and perfect detections of the data. The larger error rates in Candle and Dishes are caused by the mis-detection of corners that loses all bits of the data. Therefore, the error bits are very small and permissible in actual application using some error correction techniques, if the corners can be robustly detected.

V. Discussions and Conclusions

We have proposed a color restoration method based on color transfer and visual code for embedding original color features. The experimental results show that our visual code can robustly detected the embedded features with commercial mobile phones and the quality of color restoration surpasses the technique using color chart.

Our visual code cannot always be detected perfectly; it can, however, correct about 10% of the missing bits by additionally embedding Leed-Solomon code in the unused 40 bits. Although the visual codes of the examples in Fig. 4 are successfully detected, improper dominant colors of the sample often discolor the two colors of a visual code. This effect decreases the accuracy in code detection because of the shortening of the color distances. Our visual code is very simple and easy to implement; however, more sophisticated representation may increase payload or robustness in detection, which is the part of our future works.

References

[1] E. Reinhard, M. Ashikhmin, B. Gooch and P. Shirley, "Color transfer between images," IEEE Computer Graphics and Applications, 21, 5, pp. 34-41, 2001.

[2] D. L. Ruderman, T. W. Cronin and C. C. Chiao, "Statistics of Cone Response to Natural images: Implications for Visual Coding," J. Optical Soc. of America, 15, 8, pp. 2036-2045, 1998.

[3] L. Jing and K. Urahama, "Image Recoloring by Eigencolor Mapping," IEICE Technical Report, pp. 375-380, 2005.

[4] A. Levin, D. Lischinski and Y. Weiss, "Colorization using optimization," SIGGRAPH'04, pp. 689-694, 2004.

[5] T. Pouli and E. Reinhard, "Progressive Histogram Reshaping for Creative Color Transfer and Tone Reproduction," Non-photorealistic Animation and Rendering, pp. 81-90, 2010.

[6] K. Miyaoku and S. Fels, "C-Band: A Flexible Color Ring Tag System," User Interface Software and Technology, 2005.

[7] http://www.denso-wave.com/qrcode/index-e.html

[8] http://www.colorzip.co.jp/en/

[9] http://www.qpcard.se/

[10] N. Otsu, "A threshold selection method from gray-level histograms," IEEE Trans. On System, Man, and Cybernetics, 9, 1, pp. 62-66, 1979.

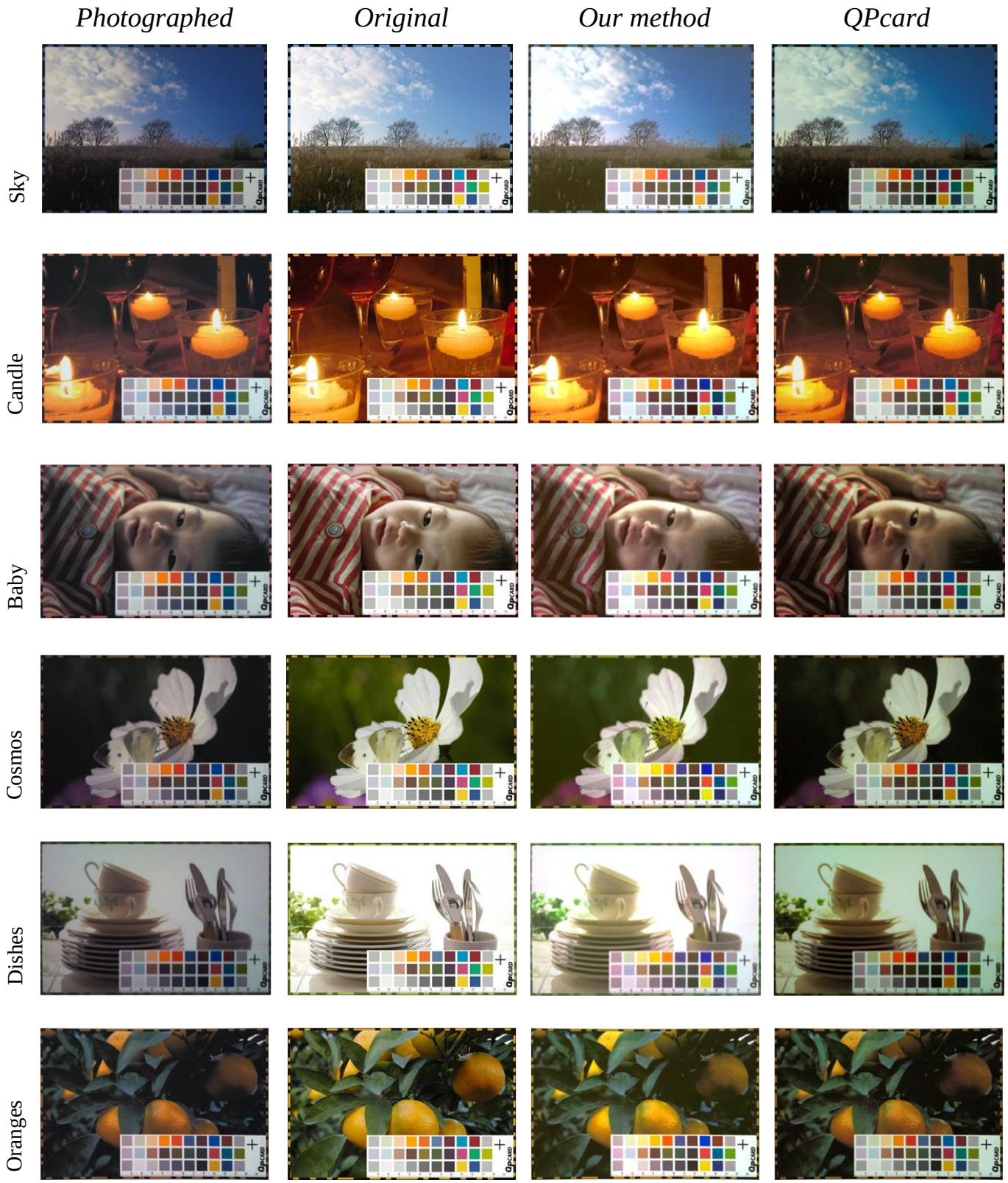

Fig. 4. Results of color restoration.