

Interactive Local Color Transfer Between Images

Alla Maslennikova, Vladimir Vezhnevets

Graphics and Media Lab., Department of Computational Mathematics and Cybernetics

Lomonosov Moscow State University,

Moscow, Russia

alla@graphics.cs.msu.ru, vvp@graphics.cs.msu.ru

Abstract

Altering image's color is one of the most common tasks in image processing. However, most of existing methods are aimed to perform global color transfer. This usually means that the whole image is affected. But in many cases colors of only a part of an image needs changing, so it is important that the rest of the image remains unmodified.

In this article we offer a fast and simple interactive algorithm based on local color statistics that allows altering color of only a part of an image, preserving image's details and natural look.

Keywords: *image processing, recoloring, color transfer, local color information, image segmentation.*

1. INTRODUCTION

The goal of a common task of image recoloring is to apply color scheme of one image to another. Usually this task supposes global color transfer. But there are cases, mostly in design, when there's no need to alter color of the whole image, because only one object or area of the image needs to be recolored.

Recoloring of a part of an image is a common task in artistic design. Usually it is solved fully manually. This means that designer has to select an object of interest with great precision and then apply some color transformation to it. To achieve natural-looking color change, object matting task needs to be solved, which is still a tedious and time-consuming task. In other cases designer has to repaint the object in a totally manual mode, using an instrument like brush.

In this paper we too consider the task of partial image recoloring for the case, when user loosely specifies an object or region of interest at the target image. The goal of our algorithm is to save user from the trouble of complicated manual work of object matting. Selecting the object is supposed in terms of object's color range. It is enough for the user to select a rectangular region, including main colors of the object of interest, like it is shown at Fig. 2, to achieve natural-looking recoloring result.

The data of the selected region is used to calculate its color statistics. We use them to estimate pixels of the target image belong, or rather are close enough, to the selected color range. Then color transformation is applied to the image, according to this estimation. The details of the algorithm will be described in section 3.

2. RELATED WORK

Most of published papers consider the task of global image recoloring. For example, in [1] color statistics of the whole source and target images are used to transfer color. This means, that even

if we just want to correct some segment of the image, the whole image will be affected.

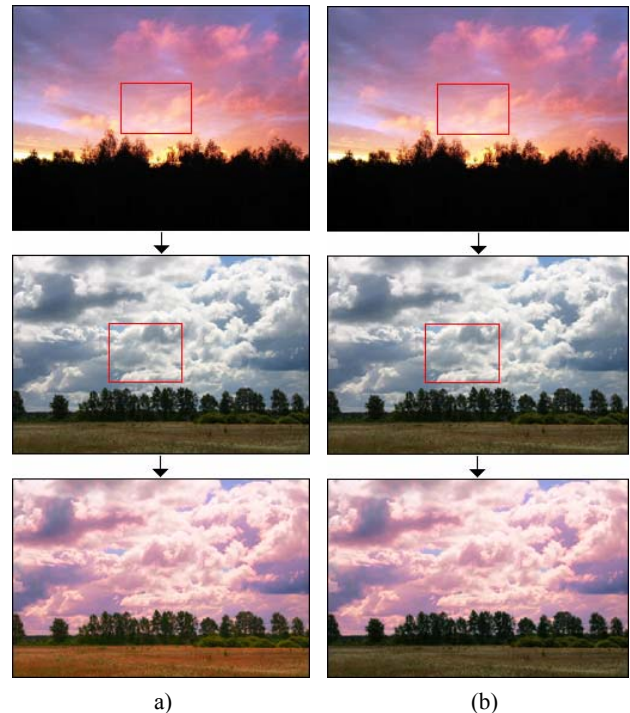


Figure 1: (a) The effect of total image recoloring using the method of Reinhard et al [1]. (b) Partial image recoloring with an image fragment using proposed method.

At Fig. 1 you can see, that the attempt to correct the color of the sky has led to spoiling the color of the field and the trees.

Some authors try to solve this problem using complex image spatial or color characteristics, but these methods have other limitations. Method based on Basic Color Categories [2],[7] is limited in variations of color changing, because any color can be replaced only by a color from the same color category. For example, we can't turn blue into red. Another method, described in [3], uses complex image color and spatial characteristics to determine image palette associations. Image color segmentation using Expectation Maximization method is offered in [5] to solve the problem of local color transfer, but segmentation and region color decision is performed fully automatically, and again for the whole image, which is not always desirable. Cellular automata is used successfully in [6] to select an object of interest, but only a single color can be used as color source and recoloring is very uniform, even when some variability of color shade is desired according to initial look and feel of the object.

The proposed method allows the user to recolor a part of the image in a simple and intuitive way, preserving other color intact and achieving natural look of the result for wide variety of input images.

3. LOCAL COLOR TRANSFER

Our method is developed to correct an object on a *target* image, selected by user. It is important, that user doesn't have to select the object by shape, for it is difficult to do precisely. The object is selected in terms of its color range and calculating its color statistics, see the details in subsection 3.1.

The source color can be defined as a single color or a region of another image, that we will call the *source* image. In this case source color statistics are calculated.

After the information of the target color range is gathered, we prepare the target image's Color Influence Map (CIM). It is a mask that specifies what parts of the target image will be affected according to the selected color range. The details are described in subsection 3.2.

The next step is recoloring itself, that is applied to the target image according to the prepared CIM. To recolor the target image we use a modified version of the Color Transfer algorithm, described in [1]. The basic algorithm uses transformation method, based on source and target color statistics in $\alpha\beta$ color space [1], [4]. The limitations of this method are 1) only an image can be used as a source for recoloring; 2) only the whole image can be recolored. These limitations are removed in our algorithm.

3.1 Calculating object's color statistics

As mentioned above, at first user has to select an object to correct at the target image. This can be done by loosely selecting a rectangular region of the object, that needs correction. There's no need to select the region precisely close to the edges of the area, that user wants to correct, because we will use its color range without spatial characteristics. The only limitation is that the whole region must be inside this object.



Figure 2: The region is selected to correct the sky.

After the user has selected the region at the target image, we calculate color statistics (mean and variation) for this region, for each channel of the working color space separately:

$$\mu_c^R = \frac{1}{N_R} \sum_{i=i_1}^{i_2} \sum_{j=j_1}^{j_2} c(i, j), \quad (1)$$

$$\sigma_c^R = \sqrt{\frac{1}{N_R - 1} \sum_{i=i_1}^{i_2} \sum_{j=j_1}^{j_2} (c(i, j) - E^R c)^2}, \quad (2)$$

where $N_R = (i_2 - i_1 + 1) \cdot (j_2 - j_1 + 1)$ is the number of pixels in the selected region R , $1 \leq i_1 < i_2 \leq H$ and $1 \leq j_1 < j_2 \leq W$ define the rectangular region R and $c(i, j)$ is a processed color channel of pixel (i, j) .

If colors of another image must be used as color source, then this step is applied to the source image as well. Note that all these and further calculations can be made in any color space, but it is shown in [4] that the most suitable color space for color transformations is Rudermann et al. $\alpha\beta$. It is also used in basic recoloring algorithm from [1], where the transformation from RGB to $\alpha\beta$ and vice versa is described. The reason for this choice is the smallest correlation between the axes in $\alpha\beta$ color space.

3.2 Building the Color Influence Map

After color statistics of the target image are gathered, they are used to determine the mask for the target image recoloring (the Color Influence Map, CIM).

CIM contains weights for color transformation for each pixel of the target image. Each pixel's weight is determined from its proximity to the color range, selected by user and stored in color statistics information. That is Mahalanobis distance between the pixel and the center of the color distribution, defined by the stored color statistics. Since $\alpha\beta$ is a color space with decorrelated axes (as stated in [1]) Mahalanobis distance turns to Euclidian:

$$\rho(x, \mu) = \|x - \mu^R\|_E, \quad (3)$$

where x is a color vector in working color space

and $\mu^R = (\mu_b^R, \mu_w^R, \mu_\beta^R)$, obtained from (1).

Weights f_{ij} in CIM are build using this formula:

$$f_{ij} = F(\rho(\mu^R, C(i, j))), \quad (4)$$

where $C(i, j)$ is a color vector, $\rho(\mu^R, C(i, j))$ is obtained from (3), $F(x)$ is defined at $[0, \infty)$ and $\lim_{x \rightarrow \infty} F(x) = 0$; $F(0) = 1$.

$F(x)$ is not strictly defined in (4), because there can be used various functions. Our experiments have shown that for photographs of good enough quality better results can be achieved using:

$$F(x) = e^{-3 \cdot x^2} \quad (5)$$

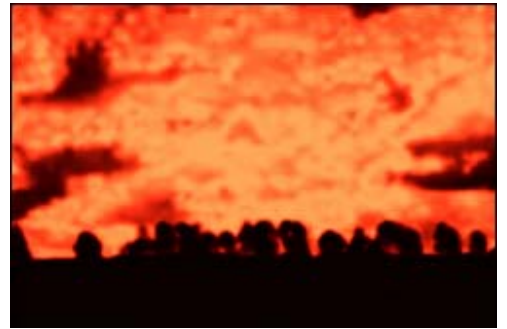


Figure 3: CIM for the selected region from Fig. 2 using (5).

3.3 Color transfer

At this step the target image is recolored according to the prepared CIM. The basic color transformation uses the stored color statistics. The below original formula from [1] is applied to all the pixels of the target image, separately for each color channel. Here and further the source data are marked with index s and the target data are marked with index t .

$$c_t^{new}(i, j) = \mu_s + \frac{\sigma_s}{\sigma_t} (c_t(i, j) - \mu_t) \quad (6)$$

We modified this transformation to include the CIM information and to apply a single color as a source, if necessary. Thus, if the source color is taken from an image, then the stored source color statistics are used:

$$c_t^{new}(i, j) = c_t(i, j) + f_{ij} \cdot \left(\mu_c^{R_s} + \frac{\sigma_c^{R_s}}{\sigma_c^{R_t}} (c_t(i, j) - \mu_c^{R_t}) - c_t(i, j) \right) \quad (7)$$

$$i = \overline{1, H_t}, j = \overline{1, W_t},$$

where: $c_t^{new}(i, j)$, $c_t(i, j)$ are new and old values of a color channel of pixel (i, j) of the target image respectively; $\mu_c^{R_s}, \mu_c^{R_t}$ are taken from (1); $\sigma_c^{R_s}, \sigma_c^{R_t}$ are taken from (2), f_{ij} is taken from (4), H_t is target image height, W_t is target image width.

If the single color Col is used as the source for recoloring, then (7) turns to:

$$c_t^{new}(i, j) = c_t(i, j) + f_{ij} \cdot (Col - \mu_c^{R_t}), \quad (8)$$

$$i = \overline{1, H_t}, j = \overline{1, W_t}.$$

Fig. 1 (b) shows the result for the image and selection from Figs. 1 (a) and 2. We can see that only the sky was corrected, while the field and the trees were left unchanged. Moreover, the details of the recolored area are saved, so that it has quite natural look.

Fig. 4 shows the example of using a single color as source color. Like in the previous example, only the sky was corrected while the field and trees weren't damaged.

And both Figs. 1(b) and 4 show, that the whole area of the selected color range was affected by color transformation. Note that in the lab color space luminance and chrominance information is separate, so it allows to make image recoloring optional in this way easily.

4. RESULTS

The proposed algorithm was tested on a set of images of different composition. Our experiments included the cases of 1) using an image fragment or a single color as color source; 2) correction of both luminance and chrominance or chrominance only and 3) using images of similar or different composition for the case of using an image fragment as color source.

The examples of image recoloring using an image fragment in a pair of images with similar composition, landscape and portrait, are shown at Figs. 1 and 6 respectively. The example of using a single color as color source is shown at Fig. 4. All these examples show correction of both chrominance and luminance.



Figure 4. Partial image recoloring with a single color used as color source.

Fig. 5 shows the example of images with different compositions. Both chrominance and luminance are used for recoloring. And Fig. 6 shows partial image recoloring using chrominance only. It is important to note that separated details of the same color range were corrected, as both figures show.

One of the important features of our algorithm is saving natural look of the image, because of the weights structure of image's CIM. Another important feature is preserving high speed of image processing, for the additional operations compared to original approach [1] have low computational complexity.

5. CONCLUSION

In this article we proposed a simple and fast algorithm for partial image recoloring that allows user to correct an object of interest at an image saving one from the trouble of selecting it precisely. The proposed method allows the user to recolor a part of the image in a simple and intuitive way, preserving other color intact and achieving natural look of the result for wide variety of input images.

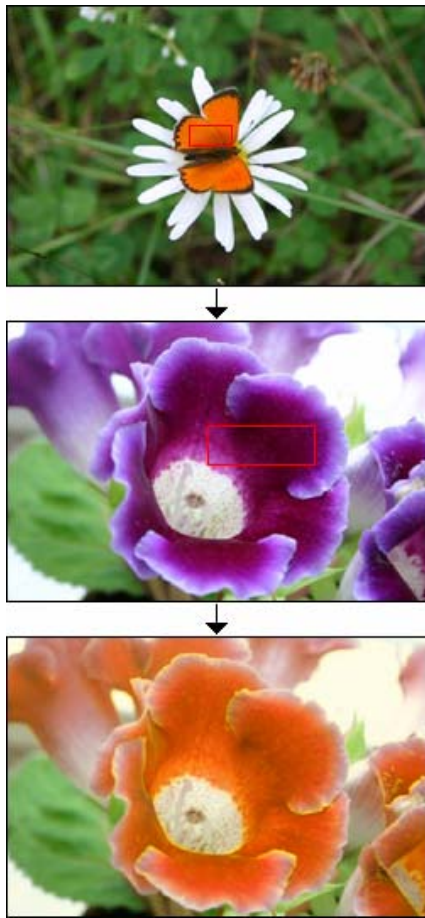


Figure 5. Partial image recoloring using images of different composition; separate areas recolored.

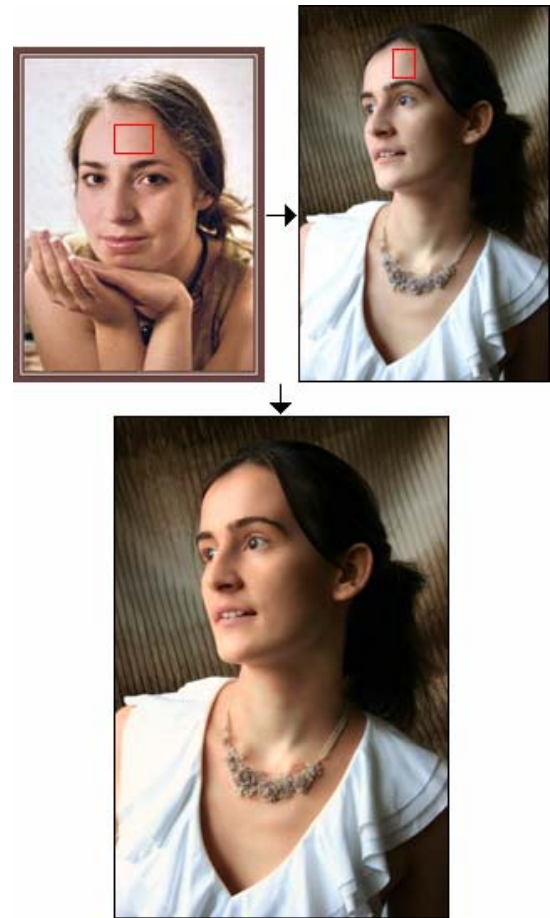


Figure 6. Skin color correction at a portrait. The model looks tanned at the result image.

6. REFERENCES

- [1] Eric Reinhard, Michael Ashikhmin, Bruce Gooch, Peter Shirley, *"Color Transfer Between Images"*, Computer Graphics and Applications, 2001, Vol.21, Issue 5, pp. 34-41.
- [2] Youngha Chang, Suguru Saito, Masayuki Nakajima, *"Color transformation based on Basic Color Categories of a Painting"*, Proceedings of Computer Graphics International, 2003, pp. 176-181.
- [3] Donald H. House, Gary R. Greenfield, *"Image Recoloring Induced by Palette Color Associations"*, Journal of WSCG, 2003, Vol.11, No.1, pp. 189-196.
- [4] Daniel L. Rudermann, Thomas W. Cronin, Chuan-Chin Chiao, *"Statistics of cone responses to natural images: implications for visual coding"*, Journal of the Optical Society of America, 1998, Vol.15, Issue 8, pp. 2036-2045.
- [5] Jiaya Jia, Chi-Keung Tang, Yu-Wing Tai, *"Local Color Transfer via Probabilistic Segmentation by Expectation-Maximization"*, IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005, Vol.1, pp. 747-754.
- [6] V. Konushin, V. Vezhnevets, *"Interactive Image Colorization and Recoloring Based on Coupled Map Lattices"*, Graphicon'2006 Conference Proceedings, 2006, pp.231-234.

- [7] B. Berlin, P. Kay, *"Basic Color Terms: Their Universality and Evolution"*, University of California Press, 1969.

About the authors

Alla Maslennikova is a Ph.D. student at Moscow State University, Department of Computational Mathematics and Cybernetics, Graphics and Media Lab.

Her contact email is alla@graphics.cs.msu.ru.

Vladimir Vezhnevets is a Ph.D. at Moscow State University, Department of Computational Mathematics and Cybernetics, Graphics and Media Lab.

His contact email is vvp@graphics.cs.msu.ru.