Eurographics Symposium on Rendering 2014
Wojciech Jarosz and Pieter Peers
(Guest Editors)

Volume 33 (2014), Number 4

AutoStyle: Automatic Style Transfer from Image Collections to Users' Images

Yiming Liu[1] Michael Cohen[2] Matt Uyttendaele[2] Szymon Rusinkiewicz[1]

[1]Princeton University [2]Microsoft Research

Figure 1: *Our system stylizes a user's photo by transferring style from a collection of images returned by a web search for a particular keyword. Examples:* sepia *(tone of old photos, reduced local contrast),* desert *(enhanced orange color, reduced global and local contrast),* spring *(enhanced green color, enhanced saturation, enhanced local contrast),* New York *(enhanced blue/violet tone, enhanced local contrast).*

Abstract

Stylizing photos, to give them an antique or artistic look, has become popular in recent years. The available stylization filters, however, are usually created manually by artists, resulting in a narrow set of choices. Moreover, it can be difficult for the user to select a desired filter, since the filters' names often do not convey their functions. We investigate an approach to photo filtering in which the user provides one or more keywords, and the desired style is defined by the set of images returned by searching the web for those keywords. Our method clusters the returned images, allows the user to select a cluster, then stylizes the user's photos by transferring vignetting, color, and local contrast from that cluster. This approach vastly expands the range of available styles, and gives each filter a meaningful name by default. We demonstrate that our method is able to robustly transfer a wide range of styles from image collections to users' photos.

Categories and Subject Descriptors (according to ACM CCS): I.3.8 [Computer Graphics]: Applications

1. Introduction

Stylizing images by manipulating color and contrast is a common task in digital post-processing of photos. In recent years, software packages such as Adobe Photoshop Lightroom [Ado07] and Apple Aperture [App05] have provided the ability to adjust color and contrast, but users still have to manually drag sliders to tune hue, tint, saturation, contrast, sharpness, vignetting, etc. An alternative approach is provided by popular online photo sharing services such as Instagram [Fac10], which offer a set of pre-designed filters to automatically stylize photos. However, since their filters are all hand-crafted by artists, users have few choices. Also, most Instagram filters do not have intuitive names. (For in-

stance, Instagram's *toaster* filter is actually named after the dog of the CEO's friend.)

In this paper, we explore a data-driven methodology to make it much easier to stylize photos. Our insight is that a *collection* of photos with the same keyword often share the same style. For instance, photos with the keyword *grass* are mostly green and have a lot of sharp edges, while photos with the keyword *night* usually are dark, and have a low local contrast. Therefore, in addition to the user's input photo I_s, we ask the user for a keyword K. The collection of images returned by searching the web for the keyword K serves as the data source for our style transfer algorithm.

The advantages of such an approach are four-fold. First, the ability to use arbitrary keywords to define styles provides users with a greater variety of possible stylizations than any pre-built set of filters. Second, a *collection* of images is robust to outlier photos, i.e., photos that are returned by the search but not really related to the keyword. Next, a collection of images instead of a single image has more chance to capture the *common* properties of a style, preventing the algorithm from over-fitting to a single image. Finally, the style generated by our system uses K as its name, which is intuitive for the user.

As Adobe Lightroom and Instagram filters do, we identify the *vignetting*, *color*, and *contrast* as the properties of a *style*. We have found that a single keyword often relates to multiple different settings of properties. For instance, *sunset* photos may have orange, blue, or violet hue. Therefore, we divide the fetched photo collection into different clusters (Sec. 3.1). Very small clusters are eliminated to make our system more robust to outliers. The final choice among the remaining clusters is left to the user.

Our style transfer algorithm begins with vignetting (Sec. 3.2). Instead of trying to find the physically correct vignetting layer, we focus on extracting a vignetting layer that captures the radially symmetric layout of lightness in the images. In the second stage, we transfer color from a cluster of images to the user's photo (Sec. 3.3). To do so we match the *3D* color histogram from a single image to a *collection* of images. We address this by modeling it as a transportation problem, and computing the color transform for each histogram bin by matching the lightness channel and a 2D elliptical distribution over the chrominance channels. Finally, we transfer local contrast by matching the fine-scale levels of Laplacian pyramids (Sec. 3.4). Some sample results of this process are shown in Fig. 1.

We summarize our contributions as follows:

- We design a system that performs photo stylization based on arbitrary keywords.
- We propose algorithms for transferring vignetting, color, and contrast from an image *collection* to the user's photos.

2. Related Work

Interactive Photo Enhancement. PixelTone [LDW*13] enables users to enhance photos via touch and voice. Shapira et al. [SSCO09] propose an interactive system for users to navigate the color transfer space. Cohen-Or et al. [COSG*06] develop a system to improve the color harmony in a photo. Though all of these systems are helpful for photo enhancement, they do not approach the problem as one of style *transfer*. Instead, users must imagine a target style, and use the capabilities of the systems to achieve that style. In contrast, in our system, the user just has to provide a keyword.

Style Analysis for Image Collections. Deselaers and Ferrari [DF11] have observed that in ImageNet [DDS*09], images with higher semantic similarity are more visually similar, which is a validation of our insight.

The work most closely related to ours is by Lindner et al. [LSBS12]. They also propose to analyze the style of photos by comparing various features of the photos annotated with a specific keyword. They analyze a million general photos using a Mann-Whitney-Wilcoxon test to assess features particularly associated with that keyword, and transfer the recovered "style". Our system has four key differences from this work. First, we do not rely on a fixed corpus of photos, but rather leverage the full power of internet image search engines. Second, Lindner et al. assume a single style for each keyword, which we show to be invalid for many keywords. For these keywords, Lindner et al. either fails, or is only able to produce an oversmoothed "average" of all styles related to that keyword. Neither meets users' expectations. In contrast, our method identifies distinct styles for the same keyword through simple and effective clustering. Third, in addition to color and global contrast, our method transfers vignetting and local contrast, which are key components of a visual style. Fourth, we contribute a collection-based multi-dimensional histogram matching algorithm that unifies the color transfer and local contrast transfer under the same framework.

Color Transfer. There are two types of color transfer algorithms: color theme based and histogram based. The color theme based methods extract a small number of the most representative colors from the image as an abstraction of the color distribution in the image. Reinhard et al. [RAGS01] and Bonneel et al. [BSPP13] transfer color between images by matching the mean and variance of colors in the CIELAB color space. Murray et al. [MSMP11] group colors with a convex clustering algorithm, and compute the color transfer by solving a minimum-cost max-flow problem. Wang et al. [WYW*10] model the relationship between texture and color based on the images fetched from the Internet to avoid unnatural color transfer. These abstraction of images often result in just a very small number of colors, ignoring much of the information.

The histogram based methods describe color distribution with histograms, which are more precise when the number of bins is large enough. Pouli and Reinhard [PR10] progressively match histograms in a multi-scale manner. Pitie et al. [PKD07] progresively match two 3D color histograms via matching 1D marginal distributions. When the source and target histograms are highly incompatible, histogram-based

Figure 2: *An overview of our system. (a) user's photo; (b) three representative images in one cluster of the web images with the keyword* sepia*; (c) after vignetting transfer; (d) after color transfer; (e) after local contrast transfer.*

methods may overfit, which causes artifacts. To resolve this issue, Freedman and Kisilev [FK10] relax the conservation constraints for the count of pixels in each bin of histograms, and compute the transfer for each histogram bin with the mean and variance of pixel values in the bin. This is a compromise between the color theme based methods and the histogram based methods. Our method is similar to that of Freedman and Kisilev [FK10], but with one key difference: we take advantage of the range of counts in each histogram bin of the image collection to further avoid overfitting.

Shapira et al. [SAHO13] is also a collection-based histogram matching algorithm, but it is limited to 1D histograms, which are unable to model the correlation between color channels.

Laffont et al. [LBP*12] propose an intrinsic image decomposition algorithm that relies on a collection of photos *captured at the same place*, with an application on relighting. Though relighting is able to mimic color transfer styles that is closely related to changes in lighting, it is difficult to be generalized to a wider spectrum of styles.

Contrast Transfer. Contrast transfer is mostly done in a multi-scale way. Bae et al. [BPD06] decompose the image into a base layer and a detail layer with bilateral filtering, and transfer contrast via histogram matching on each layer. Sunkavalli et al. [SJMP10] decompose the image into a over-sampled Haar pyramid, and match the statistics via histogram matching with gain control [LSA05]. Our method follows the same line, while we choose to decompose the image into a four-level over-sampled Laplacian pyramid because of its simplicity and high efficiency.

Example-Based Effect Transfer. Example-based effect transfer algorithms begin with original / enhanced image pairs, and transfer their effects to the users' photos. Image analogies [HJO*01] transfers the effects of filters with texture synthesis. Kang et al. [KKL10] and Caicedo et al. [CKK11] transfer parameters for generic image enhancement from a set of exemplars to users' photos. Wang et al. [WYX11] and Bychkovsky et al. [BPCD11] transfer the color and tone style based on a set of before-and-after enhancement pairs. They all require pixel-aligned pairs of original (without the style) and enhanced (with the style) images, while our method only requires images with the desired style. Shih et al. [SPDF13] transfer the visual appearance at a different time of day to users' photo based on time-lapse videos, which essentially contain pixel-aligned image pairs.

3. Style Transfer

Fig. 2 illustrates how our system stylizes an input photo I_s (Fig. 2a). We begin by asking the user for a keyword K and passing K to Bing Image Search to retrieve a collection of 500 images. We group these photos into clusters containing images with similar contrast distribution and spatial layout of colors. The user selects one of these clusters to indicate the desired stylization. Fig. 2b shows three representative images in one cluster.

We transfer the style in the CIELab color space. The a and b channels encode the color. The L channel is decomposed into a four-level oversampled Laplacian pyramid L_1, L_2, L_3, L_4 (i.e., none of the levels are downsampled). The coarse-scale level L_4 encodes global contrast, while the finer-scale levels contains most of the high-frequency information in the image, thus encoding local contrast. For convenience, we let $L^{(L)}$ denote $L_1 + L_2 + L_3$.

To transfer style from an image cluster to the user's photo I_s, we extract a vignetting layer from the L_1 channels of the images in the cluster, and multiply I_s with it to obtain the vignetting transfer result I_v (Fig. 2c). We then match the statistics of the cluster's 3D color histogram to I_v, to obtain the color transfer result I_c (Fig. 2d). Finally, we transfer the local contrast by matching the statistics of L_1, L_2, and L_3 to obtain the final result I_l (Fig. 2e).

3.1. Image Collection

The image collection is built from the top search results on Bing Image Search. After excluding invalid images and du-

Figure 3: *(a, b) Two groups of images with the keyword* sunset. *(c, d) Two groups of images with the keyword* beach.

plicates, we take the top 500 images. The downloaded images are then resized to make their longest sides have the same size as the user's photo.

Although images with the same keyword tend to be visually similar, there are often multiple distributions of color and contrast. For instance, in Fig. 3a and 3b, we see two groups of sunset images with high similarity within each group, but quite different tones between the groups. In Fig. 3c and 3d we see two clusters of beach images, in which the first group does not have any green color, while the second has substantial green. Other image search results may display an even wider variation in styles: we observed that searching for "cloud" returns images of both clouds in the sky and a popular video game character. In any case, there is not a unique mapping from keyword to visual style, and we need additional information to determine *which* of the styles returned by the keyword the user wishes to use.

Therefore, we group the downloaded image collection via k-means clustering. We represent each image with a combination of a chrominance thumbnail and gist features [OT01]. The chrominance thumbnail is computed by averaging the CIELab a and b channels down to 8×8 spatial resolution, which captures the color information. The gist feature is built by aggregating 6 oriented edge responses at 4 scales at 4×4 spatial resolution, which captures the contrast information. The spatial information present in both features is helpful for the discovery of effective vignetting structure later. The two features are concatenated with the same weight. To be more robust to outliers, the k-means clustering is done with an L_1 distance metric. Small clusters (with fewer than five images) are removed, as they are likely to be outliers. In practice, we find that using $k = 20$ (before small-cluster removal) strikes a good balance between intra-class variation and the size of the clusters. The clusters are sorted by their average L_1 distance from each instance to the cluster centroid. As a result, clusters with small intra-cluster variation are ranked higher.

We have also experimented with mean-shift clustering. We find that it generally leads to results with similar quality while requiring significantly more computation.

In our system, each cluster yields a unique style transfer result. For instance, the *sunset* filter might produce results tinted orange, blue, or violet, and the user is able to choose among them. In comparison, the method of Lindner et al. [LSBS12] only produces one result.

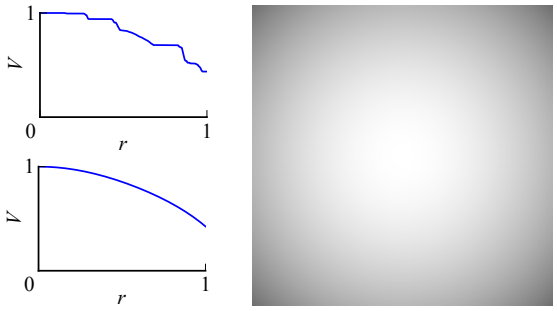

Figure 4: *Vignetting layer extraction. Left: $V(r)$ before and after parameterization. Right: The vignetting layer created with the parameterized $V(r)$. The vignetting layer is extracted from the cluster shown in Fig. 3(a).*

3.2. Vignetting Transfer

Vignetting is the attenuation of light at the image periphery, which often has the effect of drawing attention to the objects in the image center. In addition to vignetting caused by the imaging system, which we observe for keywords such as *vintage*, we notice that images with keywords such as *sunset*, *horror*, and *flowers* often exhibit a similar effect: they have a visually outstanding object in the image center, with darker pixels in the periphery. To capture both real and "effective" vignetting, therefore, we do not focus on recovering a physically correct vignetting layer, but rather aim simply to characterize the radial distribution of luminance.

We make several assumptions about the vignetting layer. First, we assume that the layer is radially symmetric; i.e., it can be represented by a function $V(r)$, where r is the distance from a pixel to the center of the image and $r = 0$ and $r = 1$ stand for the center and the corners of the image, respectively. Second, we assume that vignetting darkens the image gradually from the center to the border of the image; i.e., $V(r)$ is a non-increasing function.

To find the vignetting function, we first find a function $V_i(r)$ for every image in the cluster. For each image i, we first normalize its L_1 by setting the 10th and 90th percentiles to 0 and 1, respectively. For each possible radius r, we collect all pixels with radius r, and robustly estimate the highest pixel value $M_i(r)$ by taking the 90th percentile of these pixels. To ensure that we have a non-increasing function, we compute the vignetting function $V_i(r) = \max_{x \geq r} M_i(x)$. The vignetting function $V(r)$ of the cluster is finally found by taking the median value at each r: $V(r) = \text{median}_i V_i(r)$. Similarly to [GC05], we parameterize $V(r)$ with a 6th-order even polynomial, holding its 0th-order coefficient at 1:

$$V(r) = 1 + k_2 r_2^2 + k_4 r_4^4 + k_6 r_6^6. \quad (1)$$

Fig. 4 demonstrates an example of extracting a vignetting layer from the image cluster shown in Fig. 3(a).

To apply the vignetting layer to the user's photo, we simply multiply each pixel by $V(r)$. Vignetting transfer changes color and contrast of the image, particularly for filters like *night* and *sunset*. Therefore, we apply vignetting transfer *before* color transfer and local contrast transfer. In addition, the spatially varying luminance caused by the vignetting transfer guides color transfer and local contrast transfer to mimic the spatial layout of global and local contrast.

3.3. Color Transfer

3.3.1. Transportation Problem

The color transfer phase essentially matches color distributions. In our system, the color distribution of the vignetting transfer result is represented by a histogram h. The histogram is normalized so that the sum of all bins is 1. In the image collection fetched from the Internet, each image cluster C_i has multiple images $I_1,\ldots,I_n$, with normalized histograms $h_1,\ldots,h_n$.

In contrast to traditional single-image-based color transfer algorithms, we face a new challenge: how to match h to the *collection* of 3D histograms? One naive method might be a "loose" matching that only requires that each bin of h lie somewhere within the range of the corresponding bins of the h_i. Alternatively, a "precise" matching might force each bin of h to be exactly equal to, say, the median of the corresponding bins of the h_i. We choose an intermediate strategy, by stating that we wish to have each bin of h lie between the ath percentile and $(100-a)$th percentile of the corresponding bins in the cluster. Choosing $a=0$ would thus give us the "loose" matching strategy described above, while $a=50$ would correspond to "precise" matching. We discuss the tradeoffs in the choice of a later.

To implement this matching strategy, we define the desired lower and upper bounds on each histogram bin as

$$l(\mathbf{x}) = P_{a\%}\big(h_1(\mathbf{x}),\ldots,h_n(\mathbf{x})\big) \tag{2}$$

$$u(\mathbf{x}) = P_{(100-a)\%}\big(h_1(\mathbf{x}),\ldots,h_n(\mathbf{x})\big) \tag{3}$$

and formulate a transportation problem. We imagine each histogram bin as a warehouse, and consider how to move goods among the warehouses. Initially, warehouse $\mathbf{x}$ has $h(\mathbf{x})$ goods, and after transportation we would like it to have at least $l(\mathbf{x})$ and at most $u(\mathbf{x})$. We therefore solve for the amount of goods $f(\mathbf{x},\mathbf{y})$ to be moved between each pair of warehouses $\mathbf{x}$ and $\mathbf{y}$.

This problem is a relaxation of the Earth Mover's distance [RTG00], and can be solved by setting up a constrained optimization. First, the total amount of goods transported *from* each warehouse $\mathbf{x}$ should be equal to the amount it had originally, $h(\mathbf{x})$:

$$\sum_{\mathbf{y}} f(\mathbf{x},\mathbf{y}) = h(\mathbf{x}). \tag{4}$$

Second, at the end, the total amount transported *to* each warehouse $\mathbf{y}$ should lie in the desired interval $[l(\mathbf{y}),u(\mathbf{y})]$:

$$l(\mathbf{y}) \le \sum_{\mathbf{x}} f(\mathbf{x},\mathbf{y}) \le u(\mathbf{y}). \tag{5}$$

Third, the amount of transported goods must be non-negative:

$$f(\mathbf{x},\mathbf{y}) \ge 0. \tag{6}$$

Finally, our goal is to minimize cost:

$$\min_f \sum_{\mathbf{x},\mathbf{y}} f(\mathbf{x},\mathbf{y})\, c(\mathbf{x},\mathbf{y}), \tag{7}$$

where in our system, $\mathbf{x}$ is a 3D color vector (L_x,a_x,b_x) that represents the mean pixel value in the histogram bin, and the cost function $c(\mathbf{x},\mathbf{y})$ is defined as squared L_2 distance between colors:

$$c(\mathbf{x},\mathbf{y}) = k_L|L_x-L_y|^2 + |a_x-a_y|^2 + |b_x-b_y|^2. \tag{8}$$

To prevent flipping of lightness channel, we assign it a higher weight $k_L=9$. This is a linear programming problem (since it has linear objective and linear constraints), and can be solved quickly using the Simplex method.

3.3.2. Color Transform of Each Histogram Bin

The transportation plan f enables us to compute the color transform $T_b(\mathbf{x})$ for each source histogram bin $\mathbf{x}$. We transform the chrominance channels a,b and the lightness channel L separately.

Chrominance Channels. The distribution of a and b chrominance values in the histogram bin $\mathbf{x}$ of the vignetting transfer result can be represented by $(\mu_v(\mathbf{x}),\Sigma_v(\mathbf{x}))$, where μ_v is the vector of mean chrominance values and Σ_v is the covariance matrix. We estimate the distribution of the transformed chrominance values $(\mu_d(\mathbf{x}),\Sigma_d(\mathbf{x}))$, supposing the distribution of chrominance values in the images in the cluster is represented by $(\mu_c(\mathbf{x}),\Sigma_c(\mathbf{x}))$:

$$\mu_d(\mathbf{x}) = \sum_{\mathbf{y}} p(\mathbf{x},\mathbf{y})\,\mu_c(\mathbf{y}) \tag{9}$$

$$\Sigma_d(\mathbf{x}) = \sum_{\mathbf{y}} p(\mathbf{x},\mathbf{y})\big(\Sigma_c(\mathbf{y}) + \mu_c(\mathbf{y})\mu_c(\mathbf{y})^T\big) - \mu_d(\mathbf{x})\mu_d(\mathbf{x})^T. \tag{10}$$

We seek a transform that minimizes the stretch of chrominance values $T_b(\mathbf{c}) = \mathbf{K}\mathbf{c}+\Delta$, where $\mathbf{c}$ is the chrominance vector (a,b). We first apply SVD to Σ_v and Σ_d:

$$\Sigma_v = U_v D_v U_v^T, \quad \Sigma_d = U_d D_d U_d^T, \tag{11}$$

where $D_v = \mathrm{diag}(d_{v,1},d_{v,2})$ and $D_d = \mathrm{diag}(d_{d,1},d_{d,2})$. As described by Freedman and Kisilev [FK10], if π is the permutation that minimizes $\sqrt{d_{d,1}/d_{v,\pi(1)}} + \sqrt{d_{d,2}/d_{v,\pi(2)}}$, then

$$\mathbf{K} = U_d S_\pi P_\pi U_v^T, \tag{12}$$

where $S_\pi = \mathrm{diag}(\sqrt{d_{d,1}/d_{v,\pi(1)}}, \sqrt{d_{d,2}/d_{v,\pi(2)}})$ and P_π is the permutation matrix of π. After $\mathbf{K}$ is solved, we can compute Δ by $\Delta = \mu_d - \mathbf{K}\mu_v$.

Lightness Channel The lightness channel is transformed in

Figure 5: *The impact of the parameter a. (a) input; (b) three images in the cluster; (c–e) results with* $a = 40$, 20, *and* 1.

the same way as chrominance channels. The only difference is the lightness value is 1D instead of 2D, which makes it even more straightforward.

3.3.3. Color Transform of Each Pixel

We define the color transform $T_p(\mathbf{p})$ of each pixel $\mathbf{p}$ as a linear combination of the color transforms of all histogram bins: $T_p(\mathbf{p}) = \sum_{\mathbf{x}} \beta(\mathbf{p},\mathbf{x})\, T_b(\mathbf{x})$, where

$$\beta(\mathbf{p},\mathbf{x}) = \frac{s(\mathbf{p},\mathbf{x})}{\sum_{\mathbf{x}} s(\mathbf{p},\mathbf{x})}. \tag{13}$$

The similarity function $s(\mathbf{p},\mathbf{x})$ is important to the quality of the color transfer result. We use a Gaussian as the similarity function:

$$s(\mathbf{p},\mathbf{x}) = e^{-\frac{\|\mathbf{p}-\mathbf{x}\|^2}{2\sigma^2}}, \tag{14}$$

where $\sigma = 0.04$.

3.3.4. Parameter Settings

We normalize the three channels of the CIELab color space with the same scaling factor. After normalization, the ranges are $[0, 0.49]$, $[0, 0.84]$, and $[0, 1]$. Our color histograms have resolution $7 \times 12 \times 15$, and each histogram bin is a cube with edge length 0.07.

The value of the parameter a should be selected carefully. If a is close to 50, then it is equivalent to picking the median at each histogram bin. As a result, the variance between different images in the cluster is lost, which yields a color transfer result with a washed-out appearance. If a is close to 0, then each histogram bin is likely to be too flexible, which typically leads to the result that even before transferring color, the vignetting transfer result already satisfies the range at each histogram bin. Fig. 5 demonstrates an example. Notice that in (c) the variation between color on the buildings has been washed out, while in (e) the image has not been modified sufficiently to match the desired style. In our system, we choose $a = 20$.

3.4. Local Contrast Transfer

We represent the local contrast of an image with its high-frequency information, i.e., the finer-scale levels L_1, L_2, and L_3. Similarly to color transfer, we model these three levels with a 3D histogram over values in the Laplacian pyramid. Since local contrast is only meaningful for edges, we ignore all flat regions by excluding all pixels with $L_1 < 0.003$. The statistics of the 3D histogram are then transferred using the same algorithm as for color transfer, to obtain $L_l^{(L)} = L_{l1} + L_{l2} + L_{l3}$. Finally, we combine the local contrast transfer result with the coarse-scale level of the color transfer, to obtain the final result: $I_l = L_{v4} + L_l^{(L)}$.

Noise Reduction. Both color transfer and local contrast transfer potentially cause noise amplification. To prevent this, we convert L_{v1} into an alpha blending mask α by clamping 0.003 to 0, 0.006 to 1, and linearly varying in between. We blend $L_l^{(L)}$ and $L_v^{(L)}$ with a blurred version of this mask $\alpha * G_\alpha$, where G_α is a Gaussian function with $\sigma = 2.4$.

4. Experimental Results

4.1. Comparison to Single-Image-Based Style Transfer

First, we validate our insight of using a *collection* of images to define a style. We take color transfer as an example. For each cluster of images, we compare our result (which uses the entire cluster) with the color transfer result based on just the top-ranked image in that cluster.

For this experiment, the original image is shown in Fig. 7(a), and due to space limitations we only demonstrate comparison results on two test cases here. We refer the readers to our supplementary materials for more comparisons.

For a style defined by a single image, each histogram bin has a fixed count value instead of a range. If the color distribution of the target image is highly incompatible with the user's photo, as in Fig. 6(b), we find that the result looks unnatural.

This issue can be partially resolved by introducing an *artificial* range $[h_1(\mathbf{x}) - \sigma, h_1(\mathbf{x}) + \sigma]$ for each histogram bin $\mathbf{x}$, where

$$\sigma = \text{median}_{\mathbf{x}} \Big(\text{stdev} \left(h_1(\mathbf{x}), \ldots, h_n(\mathbf{x}) \right) \Big). \tag{15}$$

This essentially assigns the same range to all histogram bins, resolving the problem with un-natural results. However, it causes a new problem: unrelated histogram bins may be incorrectly emphasized. As shown in Fig. 6(e–h), for this image cluster we would expect the blue and green tones to be emphasized while de-emphasizing other tones. The single-image-based transfer result, however, incorrectly emphasizes the red tone.

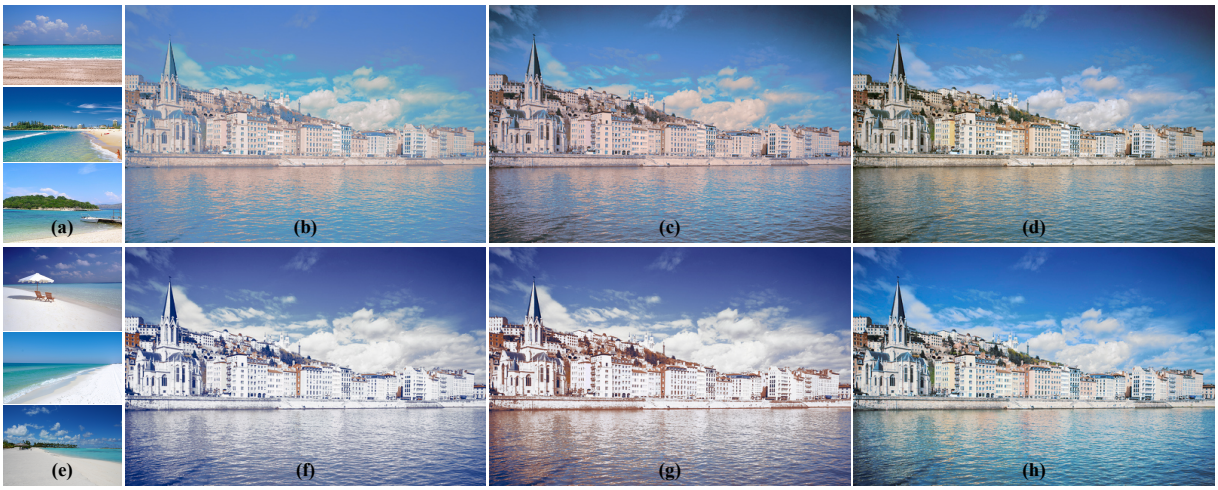

Figure 6: *(a, e) Three images in each of two clusters for the keyword* beach*. (b, f) Single-image-based color transfer for the top image in each cluster, without artificial ranges. (c, g) Single-image-based color transfer, with artificial ranges. (d, h) Our collection-based color transfer.*

4.2. User Studies

4.2.1. Methodology

We have also conducted two user studies to measure how our results match users' expectations of a filter associated with that style name. We used a pair of symmetric tests:

- **Study 1:** We showed users an image and its style transfer result. We prepared list of five style names, of which one was used to generate the style transfer result. Users were asked to select the correct name.
- **Study 2:** We showed users an image and a style name. We prepared a list of five style transfer results of the image, of which one was a result for that style name and the other 4 were from other style names. Users were asked to select the image resulting from the given style name transfer result.

For each study, we prepared 20 test cases with 20 different styles. The styles contained common objects and scenes such as *grass* and *desert*, time such as *night*, common styles of photos and movies such as *sepia* and *horror*, places such as *New York*, and abstract concepts such as *happy* and *sweet*.

How to select the *wrong* candidates is not straightforward. In Study 1, if the *wrong* style names are all quite different from the correct one in perception, then the correct one is too easy to select. If they are all highly correlated to the correct one, then it is too difficult. For instance, people may have similar expectations from *sepia* and *death*, because photos with the sepia style are often used to display people who have passed away. Therefore, we randomly selected four of nineteen styles as the wrong candidates. Similarly, in Study 2, we used style transfer results with four randomly selected styles as the wrong candidates. For all test cases, we used the style transfer results of the top-ranked cluster for each keyword. All the candidates are included in our supplementary materials.

Since "style" is a vague concept in people's minds, the feedback from individual users can vary considerably. Therefore, we conducted a relatively large user study on Amazon Mechanical Turk (MTurk) instead of a small-scale in-lab study. In each task on MTurk, we had 15 different test cases. To control the quality of the users, we repeated five of them to check the consistency of answers. In addition, we added five very easy tasks, i.e., tasks hand crafted with wrong candidates that are significantly different and easily distinguished from the correct one. To exclue respondents who simply chose randomly to complete the task, we rejected submissions in which fewer than three of the five repeated tests match, or fewer than three of the five easy tasks match the ground truth. We collected 56 and 60 submissions for Study 1 and Study 2, respectively. Each test case has answers from at least 40 different users.

Using only the top-ranked clusters can fail to produce results with distinctive visual appearance. This happened in our user study for three styles: *rust*, *sunset*, and *nightclub*. For these three styles, we conducted an additional user study with lower-ranked clusters having more distinctive visual appearance.

4.2.2. Results

Table 1 shows the proportion of users that select the correct choice in both tests. We make the following observations:

- Users easily select style transfer results related to common objects and scenes, and common photo and movie styles, with the exception of *candy*.
- For styles related to time, the style transfer results match expectations, e.g., *night*, *sunset*, and *spring*.
- For abstract concepts strongly related to visual appearance, such as *death*, which often has dark tone, reduced local contrast, and reduced saturation, the style transfer

	snow	night	grass	desert	sepia	spring	beach	horror	vintage	death
Study 1	0.927	0.929	0.905	0.800	0.786	0.810	0.634	1.000	0.860	0.595
Study 2	0.978	0.927	0.795	0.795	0.766	0.702	0.864	0.362	0.409	0.553
	wasteland	**avatar**	**rust**	**new york**	**happy**	**sweet**	**sunset**	**candy**	**kyoto**	**nightclub**
Study 1	0.643	0.372	0.349	0.442	0.250	0.256	0.310	0.075	0.167	0.146
Study 2	0.370	0.600	0.500	0.318	0.356	0.349	0.217	0.409	0.295	0.304
	rust*	**sunset***	**nightclub***							
Study 1	0.694	0.531	0.653							
Study 2	0.512	0.659	0.829							

Table 1: *Proportion of correct answers in studies asking users to select the right style name for a stylized image ("Study 1") and the right stylized image for a keyword ("Study 2"). The three results marked '*' come from additional studies that used hand-selected clusters with distinctive appearance instead of the top-ranked cluster for each keyword.*

results match well. But for concepts closely related to image content, such as *happy* and *sweet*, the results do not match as well.

- Keywords based on places names usually do not result in distinctive styles, and users have difficulty with them. Since most MTurk users are English speakers, they have more difficulty in identifying the style of places outside English speaking countries, i.e., *kyoto*.
- In general, we find that users select the correct choice significantly more often when a distinctive cluster is selected, which validates our hypothesis that clustering helps create distinctive styles.

4.2.3. Case Studies

We examine several specific cases to better understand our user study results. All these cases demonstrate multiple distinctive yet reasonable styles, validating our motivation of clustering.

Desert. Our analysis starts with *desert*, which gets good results in our user study. Sand in deserts greatly reduces image contrast, and may or may not affect sky color. In Fig. 8, two styles (corresponding to different clusters) are faithfully transferred to the user's photo from Fig 7d: note the significant changes of tone and local contrast on the green bushes, making them look as if they are covered by sand and dust. Also note the distinct differences between the clusters validating the usefulness of clustering target images before transfer.

Sunset. Photos captured at sunset may appear red, orange, or violet, depending on the number of particles in the air and the white balance setting of the camera. Since light is scattered and absorbed over long paths through the air, local contrast is low.

In Fig. 9, the sunset style is transferred from three different sunset clusters to the user's photo in Fig. 7. Our system successfully transfers the hue of sunset photos to the user's photo, and reduces its local contrast. The vignetting layer extracted from the sunset photos helps make the dragon stand out by de-emphasizing the details in the surrounding region.

Fig. 9b is based on the top-ranked *sunset* cluster, and hence was the image used in our user studies. However, this resulted in generally poor performance, so we chose to analyze this test case in more depth. In Study 1, more users choose *vintage* rather than sunset. We believe that this is a reasonable result, because 1) both vintage and sunset styles reduce local contrast; and 2) the tone of sunsets in this cluster is not very distinctive. To verify our hypothesis, we use the result in Fig. 9d, which is based on a more distinctive *sunset* cluster, to conduct an additional user study. The proportion of users selecting *sunset* increases from 31% to 53%. We find that for Fig. 9d, some users select *rust* and *autumn*, because these two styles feature with red and yellow colors, which is emphasized by this style transfer result. We think the reason why there are still users not selecting sunset for this test case is the semantics. Most sunset photos contain sky and sun, but these do not appear in the photos we use in our user studies.

New York. Photos of New York City often emphasize the high contrast of the buildings, so the *New York* style enhances local contrast. Fig. 10 shows two examples with enhanced local contrast and different colors. Fig. 10a shows the top-ranked cluster with blue sky, a green statue, and white, light yellow, and light red buildings. Fig. 10b shows a cluster with even more distinctive visual appearance, in which violet, blue, and yellow colors predominate. These colors show up in our results. The original image is in Fig 7c: note the substantial, yet plausible, shifts in color introduced by our method. Though the original image appears to have relatively uniform color, our color transport algorithm is able to use the subtle variations in hue present in the original to expand the range of colors, introducing different hues for the columns, the curved ceiling at center, and the alcove at right. In both Study 1 and Study 2, though the number of users that make the correct choice is less than half, it is still more than any other wrong style. Due to lack of semantics, it is difficult for users, particularly users outside US, to associate the visual appearance with New York. However, the style still looks visually pleasing.

Figure 7: *Photos used as inputs in the case studies.*

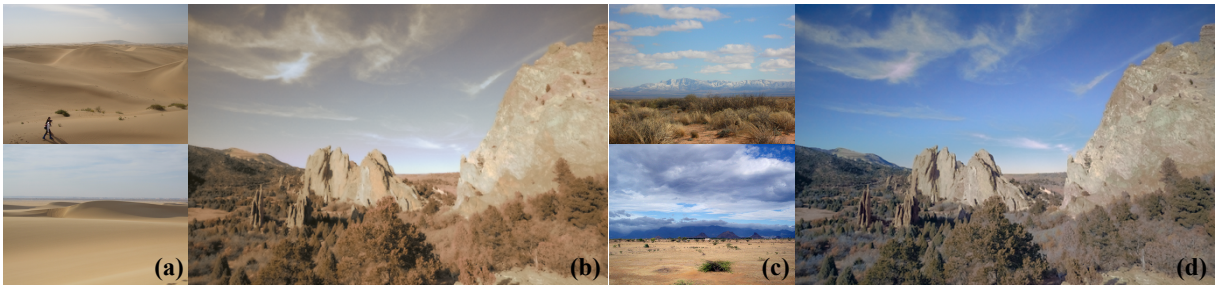

Figure 8: *Case study:* Desert. *(a–b) two images from a cluster, and its style transfer result; (c–d) another result.*

Figure 9: *Case study:* sunset. *(a–b) Three images from the top-ranked cluster, and its style transfer result; (c–f) Images from two other clusters, and their transfer results.*

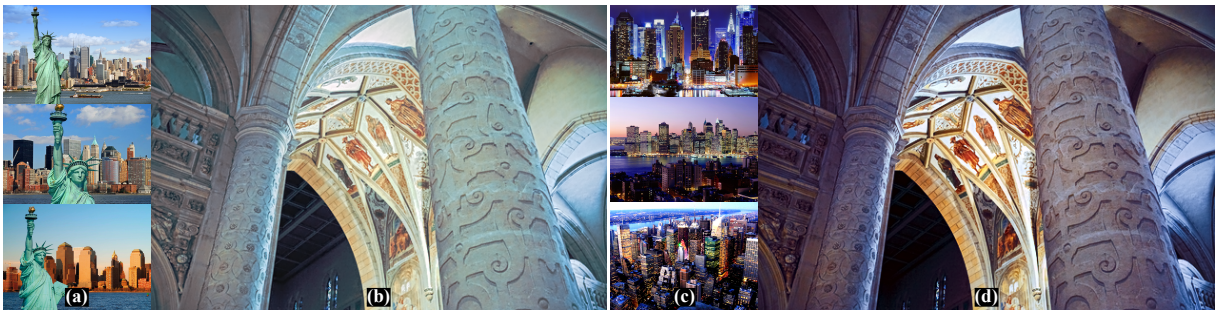

Figure 10: *Case study:* New York. *(a–b) three images from a cluster, and its style transfer result; (c–d) another result.*

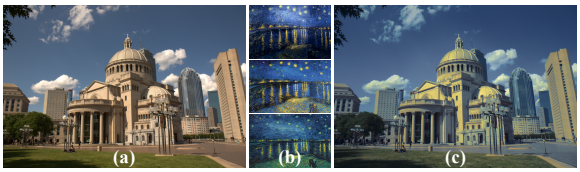

Figure 11: *Failure case of our method. (a) The original image. (b) Three representative images in the cluster with the keyword* Van Gogh. *(c) Style transfer result — note that our method does not transfer local texture.*

5. Discussion

Our user studies seek to find out whether a style name produces expected results, and conversely if a result is produced by the expected style name. A more important question may be whether the filters produced by the method are desirable to users. Crafting filters by hand is tedious and requires expert knowledge. Allowing filters to be automatically produced by keywords may result in many beautiful results independent of their predictability. A further study would be required to assess this type of efficacy. An additional study should also assess how reliably users select keywords to generate expected results.

5.1. Limitations

Our system fails for styles not well defined by their color and contrast distributions. First, for styles strongly dependent on the content in the photos, such as *happy* and *sweet*, it is difficult to consistently produce style transfer results that most users agree on.

Second, some visual styles are defined not by their color and contrast distributions, but rather by local texture. Because we do not transfer the latter, we do not necessarily produce the expected results for keywords such as *Van Gogh* (see Fig. 11). Though a variety of systems for texture synthesis and "texture painting" have been explored in recent years, we believe that the investigation of *collection-based* texture transfer would make for interesting future work.

Third, our system relies on the power of image search engines. We find that typically many images retrieved are visually similar. However, due to the limitations of the filtering and ranking algorithms of search engines, a small portion of images may be irrelevant, which we call *noise*. Clustering helps group the relevant images into large clusters, and thus reduces the noise. However, some other clusters may be more random due to noise. Since we present results of all clusters to users, users may find that results for these clusters are not as good as others. In the future, we may adopt more sophisticated image understanding algorithms to filter out the noise, and thus further improve the quality of the results presented to users.

Fourth, our system does not consider the existing vignetting in the input image, thus yielding overly strong vignetting in some cases, e.g. Fig. 6(d). This problem may be resolved by estimating the *effective* vignetting in the inputs, and adding only enough additional vignetting to match it to the target. Though algorithms for detecting real vignetting have been explored, the investigation of *effective* vignetting would be interesting future work.

5.2. Running Time

Our system is implemented in Matlab. On a machine with a 2.2GHz CPU, using a single CPU thread, for photos with a 480-pixel longest edge, the time cost of k-means clustering is under 2 seconds. Times for applying the vignetting layer, color transfer, and contrast transfer were 0.02 seconds, 2 seconds, and 1.5 seconds, respectively.

Downloading, resizing, and extracting features and vignetting layers for hundreds of web images consumes tens of minutes in total, depending on the network condition and I/O speed. However, we envision that for common keywords the results could be pre-computed (at a number of resolutions) and distributed with the software. When the user wishes to filter a photo, the images with the closest resolution are used. For relatively unpopular keywords, after the first time downloading, resizing, and extracting features, the clustering result, vignetting layer, and histogram models can be cached and even shared with other users.

6. Conclusion

We have demonstrated a data-driven system to automatically transfer style to a user's photos. To achieve this we download images related to a user-provided keyword, group the image collection into clusters based on their visual appearance, transfer vignetting based on the lightness distribution, and transfer color and contrast by matching 3D CIELab and Laplacian pyramid histograms. Our experiments suggest that the system is able to robustly transfer a variety of styles.

Acknowledgements

We would like to thank all reviewers for their constructive comments, and the members of MSR IVM Group and Princeton Graphics Group for their thoughtful discussion. Part of this work was done when the first author was an intern at MSR. This work is also supported by NSF grants CCF-1012147, CCF-1027962.

References

[Ado07] ADOBE: Adobe Photoshop Lightroom, 2007. 1

[App05] APPLE: Apple Aperture, 2005. 1

[BPCD11] BYCHKOVSKY V., PARIS S., CHAN E., DURAND F.: Learning photographic global tonal adjustment with a database of input / output image pairs. In *Proc. CVPR 2011* (2011). 3

[BPD06] BAE S., PARIS S., DURAND F.: Two-scale tone management for photographic look. *Proc. SIGGRAPH 2006* (2006). 3

[BSPP13] BONNEEL N., SUNKAVALLI K., PARIS S., PFISTER H.: Example-based video color grading. *Proc. SIGGRAPH 2013* (2013). 2

[CKK11] CAICEDO J., KAPOOR A., KANG S. B.: Collaborative personalization of image enhancement. In *Proc. CVPR 2011* (2011). 3

[COSG*06] COHEN-OR D., SORKINE O., GAL R., LEYVAND T., XU Y.-Q.: Color harmonization. *Proc. SIGGRAPH 2006* (2006). 2

[DDS*09] DENG J., DONG W., SOCHER R., LI L.-J., LI K., FEI-FEI L.: Imagenet: A large-scale hierarchical image database. In *Proc. CVPR 2009* (2009). 2

[DF11] DESELAERS T., FERRARI V.: Visual and semantic similarity in imagenet. In *Proc. CVPR 2011* (2011). 2

[Fac10] FACEBOOK: Instagram, 2010. 1

[FK10] FREEDMAN D., KISILEV P.: Object-to-object color transfer: Optimal flows and SMSP transformations. In *Proc. CVPR 2010* (2010). 3, 5

[GC05] GOLDMAN D., CHEN J.-H.: Vignette and exposure calibration and compensation. In *Proc. ICCV 2005* (2005). 4

[HJO*01] HERTZMANN A., JACOBS C. E., OLIVER N., CURLESS B., SALESIN D. H.: Image analogies. In *Proc. SIGGRAPH 2001* (2001). 3

[KKL10] KANG S. B., KAPOOR A., LISCHINSKI D.: Personalization of image enhancement. In *Proc. CVPR 2010* (2010). 3

[LBP*12] LAFFONT P.-Y., BOUSSEAU A., PARIS S., DURAND F., DRETTAKIS G.: Coherent intrinsic images from photo collections. *Proc. SIGGRAPH 2012* (2012). 3

[LDW*13] LAPUT G. P., DONTCHEVA M., WILENSKY G., CHANG W., AGARWALA A., LINDER J., ADAR E.: PixelTone: A multimodal interface for image editing. In *Proc. SIGCHI 2013* (2013). 2

[LSA05] LI Y., SHARAN L., ADELSON E. H.: Compressing and companding high dynamic range images with subband architectures. *Proc. SIGGRAPH 2005* (2005). 3

[LSBS12] LINDNER A., SHAJI A., BONNIER N., SÜSSTRUNK S.: Joint statistical analysis of images and keywords with applications in semantic image enhancement. In *Proc. ACM Multimedia 2012* (2012). 2, 4

[MSMP11] MURRAY N., SKAFF S., MARCHESOTTI L., PERRONNIN F.: Towards automatic concept transfer. In *Proc. NPAR 2011* (2011). 2

[OT01] OLIVA A., TORRALBA A.: Modeling the shape of the scene: A holistic representation of the spatial envelope. *IJCV* (2001). 4

[PKD07] PITIÉ F., KOKARAM A. C., DAHYOT R.: Automated colour grading using colour distribution transfer. *CVIU* (2007). 2

[PR10] POULI T., REINHARD E.: Progressive histogram reshaping for creative color transfer and tone reproduction. In *Proc. NPAR 2010* (2010). 2

[RAGS01] REINHARD E., ASHIKHMIN M., GOOCH B., SHIRLEY P.: Color transfer between images. *IEEE CG&A* (2001). 2

[RTG00] RUBNER Y., TOMASI C., GUIBAS L. J.: The earth mover's distance as a metric for image retrieval. *IJCV* (2000). 5

[SAHO13] SHAPIRA D., AVIDAN S., HEL-OR Y.: Multiple histogram matching. In *Proc. ICIP 2013* (2013). 3

[SJMP10] SUNKAVALLI K., JOHNSON M. K., MATUSIK W., PFISTER H.: Multi-scale image harmonization. *Proc. SIGGRAPH 2010* (2010). 3

[SPDF13] SHIH Y., PARIS S., DURAND F., FREEMAN W. T.: Data-driven hallucination of different times of day from a single outdoor photo. *Proc. SIGGRAPH Asia 2013* (2013). 3

[SSCO09] SHAPIRA L., SHAMIR A., COHEN-OR D.: Image appearance exploration by model-based navigation. *Computer Graphics Forum* (2009). 2

[WYW*10] WANG B., YU Y., WONG T.-T., CHEN C., XU Y.-Q.: Data-driven image color theme enhancement. *Proc. SIGGRAPH 2010* (2010). 2

[WYX11] WANG B., YU Y., XU Y.-Q.: Example-based image color and tone style enhancement. *Proc. SIGGRAPH 2011* (2011). 3