Style Transfer Via Image Component Analysis

Wei Zhang, Chen Cao, Shifeng Chen, Jianzhuang Liu, *Senior Member, IEEE*, and Xiaoou Tang, *Fellow, IEEE*

***Abstract*—Example-based stylization provides an easy way of making artistic effects for images and videos. However, most existing methods do not consider the content and style separately. In this paper, we propose a style transfer algorithm via a novel component analysis approach, based on various image processing techniques. First, inspired by the steps of drawing a picture, an image is decomposed into three components: draft, paint and edge, which describe the content, main style, and strengthened strokes along the boundaries. Then the style is transferred from the template image to the source image in the paint and edge components. Style transfer is formulated as a global optimization problem by using Markov random fields, and a coarse-to-fine belief propagation algorithm is used to solve the optimization problem. To combine the draft component and the obtained style information, the final artistic result can be achieved via a reconstruction step. Compared to other algorithms, our method not only synthesizes the style, but also preserves the image content well. We also extend our algorithm from single image stylization to video personalization, by maintaining the temporal coherence and identifying faces in video sequences. The results indicate that our approach performs excellently in stylization and personalization for images and videos.**

***Index Terms*—Example-based stylization, non-photorealistic rendering, video stylization and personalization.**

I. Introduction

IMAGE stylization has been an emerging technique during the past decade [1], [2]. It is categorized into the area of non-photorealistic rendering (NPR) in the graphics community [1]. Much research has been devoted to rendering different styles, e.g., oil painting, watercolor painting [3] and abstract drawing [4]. However, most of these methods support only limited artistic styles. For a new style much human knowledge and experiences are required [2].

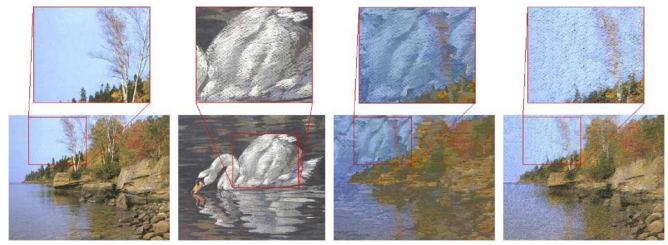


Fig. 1. Comparison of luminance-based style transfer and our method. From left to right: A, B^+, A^+ by image analogies [5], and A^+ by our method. The body of the swan appears in A^+ by image analogies. The source image and style template are downloaded from the project page of image analogies http://mrl.nyu.edu/projects/image-analogies/pastel.

Example-based image stylization provides an easy way of creating stylized images with a number of styles. The stylized image is synthesized from a real image (e.g., a scene photo) with a given style template (e.g., an oil painting). Formally, given two images as input, a source image A and a template image B^+ whose style is to be simulated, the output A^+ is a synthesized image with the main content in A and the similar style to B^+. This process is also called style transfer.

One critical problem of the existing approaches is that they do not separate the style and content in the style transformation process. In [5]–[7], only luminance is transferred from B^+ to A, which brings two drawbacks. First, the luminance of two input images may not be in the same dynamic range. To address this problem, a linear mapping that matches the means and variances of the two luminance distributions is often adopted. But usually good correspondences cannot be found for some input images. We will show some such examples in Section VI for examples. Second, the content of B^+ may appear in the output images. Fig. 1 shows such an example.

In this paper, we propose a novel algorithm to convert a real image to a stylized image with the similar artistic style with an arbitrarily given template image. To break the limitations of the previous approaches, the basic idea of our algorithm is to simulate the artists' drawing procedure, i.e., to decompose both the source image and template image into draft, paint, and edge components, which describe the content, the main style, and the strengthened strokes along boundaries. We introduce the pixel grouping techniques from image analysis to simplify image decomposition greatly. Then the style is transferred from the template image to the source image in the paint and edge components. Style transfer is formulated as a global optimization problem by using Markov random fields (MRFs) [8], and a coarse-to-fine belief propagation algorithm is used to solve the optimization problem. To combine the draft component and

Manuscript received July 31, 2012; revised November 19, 2012; accepted January 02, 2013. Date of publication June 03, 2013; date of current version October 11, 2013. This work was supported in part by Guangdong Innovative Research Team Program No. 201001D0104648280); Science, Industry, Trade, and Information Technology Commission of Shenzhen Municipality (60903117); and Science, Industry, Trade, and Information Technology Commission of Shenzhen Municipality, China (JC201005270357A). The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Nicu Sebe.

W. Zhang is with the Department of Information Engineering, the Chinese University of Hong Kong, Hong Kong (e-mail: zw009@ie.cuhk.edu.hk).

C. Cao is with Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, China (e-mail: chen.cao@siat.ac.cn).

S. Chen, the corresponding author, is with Shenzhen Key Laboratory for Computer Vision and Pattern Recognition, Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, China (e-mail: shifeng.chen@siat.ac.cn).

J. Liu is with Media Laboratory, Huawei Technologies Co., Ltd., Shenzhen 518129, China, and also with the Department of Information Engineering, The Chinese University of Hong Kong, Hong Kong (e-mail: liu.jianzhuang@huawei.com).

X. Tang is with the Department of Information Engineering, The Chinese University of Hong Kong, Hong Kong, and also with Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, China (e-mail: xtang@ie.cuhk.edu.hk).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMM.2013.2265675

the obtained style information, the final artistic result can be achieved via a reconstruction step.

One advantage of our method is its ease of use. For a given arbitrary style template image, it can automatically transform the style of an input image to the template style. It does not require the registered pair of source and stylized images [5], [6], or any user input [9]. The second advantage is that compared with other methods, our algorithm preserves the content of the input image and synthesizes the style, since we solve the problem in different image components. The third advantage is that it can be easily extended to other applications. In this paper, an extension application, video stylization and personalization [10], is proposed. In this application, a video is stylized by preserving temporal coherence between frames, and personalized by making human faces identifiable.

II. Related Work

There has been a rich amount of research on example-based image stylization. The essential problem is to find the mapping from the local pixels/patches of the source image to those of the stylized image. Previous methods assume three different settings: supervised, semi-supervised, and unsupervised, which are terms borrowed from the area of machine learning.

In the supervised setting, the ground-truth image B corresponding to B^+ is given. Hertzmann *et al.* proposed image analogies to estimate the mapping, using the source image B to the stylized image B^+ [5]. Image stylization for highly specialized problems has also been attempted for faces [11]–[13], using very large sets of training data.

In the semi-supervised setting, some parts of A^+ are available as training data, and thus ground-truth, i.e., some correspondences between A and A^+, is known. Cheng *et al.* [6] proposed to use a semi-supervised component to exploit this setting. The similarities between the source patches are utilized to propagate information from the source patches with stylized counterpart to the patches without stylized counterpart.

In the unsupervised setting, the ground-truth source image B is not given. In real-world problems, people usually have a painting without the corresponding real image. Therefore, the unsupervised setting is more user friendly, but more difficult as well. To deal with the difficulty of lacking ground-truth B, Rosales *et al.* used a finite set of patch transformations [7]. They formulated the problem as inferring the latent variables. Wang *et al.*'s method [9] requires the user-specified blocks in B^+ as sample textures, and then the textures are applied on segmented image A. Our method also belongs to this category. Comparing to them, our method requires the least information: neither user input [9] nor assuming a set of transformations [7]. In addition, we utilize the full image of B^+ instead of a very small subset of B^+ [9]. The required supervision information of the above-mentioned approaches is summarized in Table I.

As mentioned in Section I, our approach separates the style and content through decomposition on image components, and thus avoids the severe problems of applying the existing approaches in example-based image stylization [5]–[9] to real-world applications. Drori *et al.*'s method [14] is related to ours in the aspect of exploring content and style decomposition. However, they used a set of images with the same style to learn content and style decomposition. It is not trivial to apply their solution to existing example-based image stylization approaches. The algorithm in [15] also uses the idea of image decomposition. Different from our method, it is not example-based and not suitable for style transfer.

TABLE I
Comparison of the Required Supervision Information in Different Example-Based Image Stylization Approaches. The Input Source Image and the Output Image of all the Methods are Denoted by A and A^+. B is the Ground-Truth Image Corresponding to B^+

Approach	Supervision information	Category
[5]	well aligned B and B^+	Supervised
[6]	available parts of A^+	Semi-supervised
[9]	user-selected parts of B^+	Unsupervised
[7]	B^+ and a set of transformations	Unsupervised
ours	B^+	Unsupervised

Constrained texture synthesis [16], [17] is a topic related to example-based image stylization. However, all existing methods for constrained texture synthesis were supervised. In addition, the mismatching problem between the characteristics of the source image and the template image does not need to be considered in this application.

Extension from single image stylization to video by adding temporal coherence is a natural conception. Previous methods mainly focus on specific style like cartoon [18] and watercolor [19]. Here our video stylization is a framework applicable to example-based arbitrary style. Moreover, inspired by a recent cartoon personalization system called EasyToon [20], we detect human face in the video and prevent it from indistinction, i.e., to personalize a video. The work of video stylization and personalization is an extension from our previous work [10].

III. Algorithm Overview

Our main idea originates from the steps of drawing a picture. First, an artist draws a draft on blank canvas, which means the main content of an image. Then, paint will be laid on the canvas. We define the paint as "style" in this paper, i.e., the type of the paint like oil, watercolor, pastel, etc. At last, the stroke edges will be either blurred or strengthened according to specific style. Therefore, our style transfer approach separates an image into three components, the draft, paint, and edge components. In our method, the draft represents the main content of the image, the paint represents the style, and the edge represents the details along the stroke boundaries.

Such a component analysis scheme solves the style transfer problem for A and B^+ with different luminance distribution (content) elegantly. First, the luminance distributions of A and B^+ are determined by the draft component. Then the paint and edge components of A^+ simulate the characteristics of the corresponding components of B^+. In our algorithm, the simulation step is formulated as a global optimization problem using MRFs [8]. Finally the three components of A^+ are combined to reconstruct the final image A^+. The process of our algorithm is shown in Fig. 2.

IV. Image Style Transfer

Before the processing, we convert two input images A and B^+ from the RGB color space to the YIQ color space. Ex-

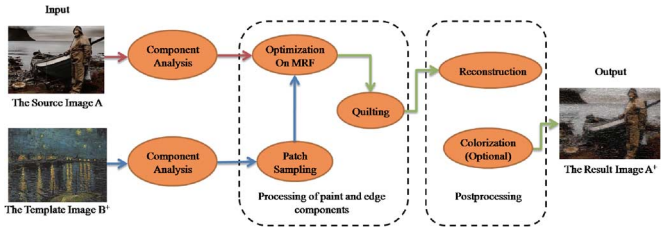


Fig. 2. The framework of image style transfer.

Fig. 3. Two examples of image paint components. From left to right: two template images, the segmentation results, and the paint components.

cept image segmentation, all operations are conducted in the Y channel, and the I and Q channels are stored for the final color restoration. The selection and separate processing of YIQ color space just follows the classic work of Hertzman *et al.* [5], as the artistic style is much more visually sensitive to changes in the Y channel than in the I and Q channels.

A. Component Analysis

In this paper, an image is composed of the draft, paint, and edge component. We notice that different components in an image contain different information. The draft component is affected by the illuminance and radiation of the image formation process and is almost irrelevant with the style. The paint component contains textures and determines the style of the image (see Fig. 3). In our algorithm, the edge component represents information along the boundaries. In addition, the draft component in an image is normally much larger in amplitude than the edge component. So, we argue that it is necessary to perform a component decomposition before transferring the style from the template image to the source image. To our best knowledge, we are the first to consider this problem for example-based image stylization.

In our approach, an image $I(x, y)$ is a combination of draft, paint, and edge components. We denoted it as

$$I(x, y) = I_D(x, y) \oplus I_P(x, y) \oplus I_E(x, y), \tag{1}$$

where I_D, I_P, and I_E are the draft, paint, and edge component.

We design a two-step strategy in our algorithm to decompose the three components. The first step is to separate the paint component from the image. It can be achieved via image segmentation, which is simply a partition of an image into contiguous regions of pixels that have similar appearances. Let the segmented image of A be A_S, in which each segment is represented by its mean luminance. A_S contains $A_D(x, y)$ and $A_E(x, y)$, the draft and edge components of A, which is

$$A_S(x, y) = A_D(x, y) \oplus A_E(x, y). \tag{2}$$

We define

$$A_P(x, y) = A(x, y) - A_S(x, y), \tag{3}$$

where A_P is the paint component of A.

Mean shift is a *nonparametric* data clustering technique, which does not need to specify the number of clusters, and has been successfully applied to image segmentation [21]. In the mean shift image segmentation in this paper, each pixel is assigned a feature point in a five-dimensional space, consisting of two spatial coordinates and three RGB components. The feature points are grouped by the clustering algorithm. Note that image segmentation is conducted in RGB channels, but not only in the luminance channel.

In the second step, we use the gradients of the segmented image A_S to estimate the edge component of A, i.e.,

$$A_E(x, y) = \left[\frac{\partial A_S(x, y)}{\partial x}, \frac{\partial A_S(x, y)}{\partial y}\right]. \tag{4}$$

We also perform the same decomposition on B^+ to obtain B_S^+, B_P^+ and B_E^+.

B. Paint and Edge Component Processing

In this stage, information is propagated from the paint and edge components of B^+ to those of A. Patches are sampled from the components of B^+, and organized to be the components of a new image A^+, which is close to A in some measure, i.e., we obtain A_P^+ and A_E^+ from B^+ and A. In the following description, the reference images are referred to the components of B^+, which the candidate patches come from, and the target images are the corresponding components of A, which need to be covered by patches. Note that the following computation is conducted in the paint and edge components.

1) Global Optimization on Markov Random Fields: We formulate the patch mapping problem as a labeling problem modeled by discrete MRFs [8]. First, the reference image is sampled as a dictionary $\mathcal{P}$ of $w \times h$ patches. Then the target image is divided into overlapping patches with the same size.

Construct an undirected graph $G = (V, E)$, where the node set $V = \{v_1, v_2, \ldots, v_N\}$ contains all the patches in the target image, and E is the set of edges connecting each node to its four neighbors. For each node v_i we assign a patch x_i from the dictionary $\mathcal{P}$. Then the problem is to find the best configuration $X = \{x_1, x_2, \ldots, x_N\}$ to minimize an energy function defined later, where $x_i \in \mathcal{P} (1 \leq i \leq N)$.

The placement of patches should match the target image and have local consistency. So, our energy function is

$$E(X) = \sum_{v_i \in V} E_1(x_i) + \lambda \sum_{(v_i, v_j) \in E} E_2(x_i, x_j), \tag{5}$$

where $E_1(x_i)$ is the penalty cost of assigning the patch x_i to the node v_i, $E_2(x_i, x_j)$ is the consistency cost of a neighboring node pair (v_i, v_j) having labels (x_i, x_j), and λ is a balance factor. We set $\lambda = 5$ in all experiments.

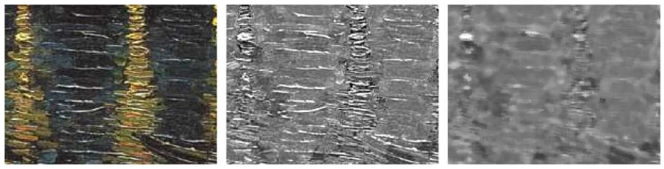


Fig. 4. Components of a template image. Left: the template image. Middle: the paint component. Right: the filtered paint component. The segmented image has strong stroke boundaries, which belong to the edge component. The paint component contains the style characteristic, while a median filter can remove most of the style.

The definition of $E_1(x_i)$ is the sum of the squared differences (SSD) of pixel features between x_i and the region that x_i covers in the target image. $E_2(x_i, x_j)$ is the SSD of pixel features in the overlapping region between x_i and x_j. In summary, E_1 is used to control the reliability of the synthesized image and E_2 helps to produce a seamless synthesized image.

2) Component Image Quilting: After the placement of the patches, the component image quilting from patches is performed via the minimum cut algorithm. Image quilting, which aims to produce a seamless image from overlapping patches, has been extensively studied in texture synthesis [22], [23]. The idea of image quilting is to find a seam with the least inconsistencies between neighboring patches with overlapping regions, which is formulated as a minimum cut problem.

In our implementation, the patches are placed successively. For the placement of each patch, we construct a graph whose nodes are the pixels that are in the overlapping region of the existing patches and the new patch. A source node and a sink node are added to represent the existing patches and the new patch. In the graph, the boundary pixels are connected to the source or sink node with an infinite weight and each node is connected to its four neighbors. The weight of the edge connecting a neighboring pixel pair (i, j) is defined as

$$W(i,j) = \|\mathbf{F}(i) - \mathbf{F}_{new}(i)\|^2 + \|\mathbf{F}(j) - \mathbf{F}_{new}(j)\|^2, \quad (6)$$

where $\mathbf{F}(\cdot)$ and $\mathbf{F}_{new}(\cdot)$ denote the existing and new features of a pixel, and $\|\cdot\|$ is the L2-norm. After a cut is obtained, the existing features are updated according to the cut.

3) Implementation Details: The following implementation details of the above steps are highly related to the quality and speed of the algorithm.

First, to enhance the synthesis quality, we observe that the paint component of the template image usually contains strong style characteristic, which may bring noise in the MRF model. So in the MRF model, the paint component of the template image is processed as (Fig. 4)

$$\widetilde{B}_P^+ = \mathbf{Median}\left(B_P^+\right), \quad (7)$$

and the dictionary $\mathcal{P}$ is taken from $\widetilde{B}_P^+$, where $\mathbf{Median}$ is the median filter. In the quilting step, we use the corresponding patches from the original paint component B_P^+ to keep the style characteristic.

Second, we design several mechanisms to speed up the MRF optimization. In our application, both the size of dictionary $\mathcal{P}$ and the number of nodes in MRFs are large, due to the large size of images used for stylization. Although the popular belief propagation (BP) is adopted to efficiently solve the energy minimization problem, it is still necessary to find some way to speed up the optimization. In this paper, we accelerate the algorithm via three aspects as follows. The first two are for reducing the size of dictionary $\mathcal{P}$, as the computational complexity of BP is the square of the number of the patches in the dictionary, and the last one is for reducing the number of nodes in the MRF.

1) In order to reduce the size of the dictionary, we utilize only 50% of the most representative sampled patches. In our implementation of constructing the paint component dictionary, the quality of a patch in representing the style is

$$d_p = \sum_{(x,y)\in p} \left|\widetilde{B}_P^+(x,y) - B_P^+(x,y)\right|, \quad (8)$$

where $(x, y) \in p$ means that (x, y) is a point in patch p. The larger d_p is, the more style information the patch p contains. We choose the patches with the top 50% d values in the dictionary. It works well for all the artistic styles we test.

2) We use a two-step coarse-to-fine BP algorithm [24] to reduce the computational cost. First, the patches in the dictionary are divided into K clusters with the k-means algorithm. Then, the first BP is applied to find labels in the set of centers of clusters $\mathcal{P}^c = \{c_1, c_2, \ldots, c_K\}$, where c_i is the center of the i-th cluster. Finally, we perform the second BP for each cluster to select the best patches in the cluster. More details about the two-step BP can be found from [24].

3) In the optimization process for the edge component,

$$if\ \forall (x,y) \in p, A_E(x,y) = 0,\ then\ A_E^+(x,y) = 0. \quad (9)$$

It means that, for the patches far away from the edges in the image, we keep their result all zero in the optimization process, since no boundary information should be transferred to regions far away from the boundaries. Thus the number of unlabeled nodes is greatly reduced in the optimization step for the edge component, which greatly reduces the running time.

Besides, for the paint and edge components, the patch sizes are different. The edge component uses a small size to keep more details, while the paint component uses a relatively large size to make the algorithm faster.

The processing of the edge component described in Section IV-B is used to characterize the fusion of different strokes around the boundaries in the template image. In the paint component, styles are transferred independently for each segment. If the edge component of A is copied to the output image A^+ without any processing, the edges of the output image A^+ are of high contrast (see Fig. 5 for example). Most of the artistic styles do not have such high contrast. The edge component can reduce the contrast and make boundaries look like a paint style much more.

For some styles, the amount of edge components is important. A case in point, would be impressionist art with lots of little strokes. Notice that keeping the paint does not make the points and little strokes disappear if the image is not in a very low resolution. The edge component is contributed by the boundaries of the points and little strokes only, but not by the whole points

Fig. 5. Comparison of the output images with paint and edge processing (the left column) and with only paint processing (the right column). The template image is "Freud" in Fig. 7.

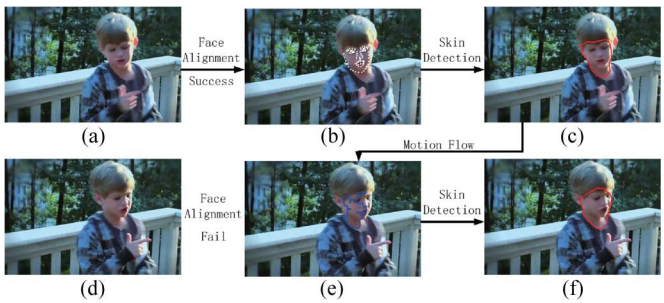


Fig. 6. Example of the face extraction process. (a): The 10th frame in the source video. (b): The landmarks (white points) of the face alignment result. (c): The result of skin detection for (a). (d): The 11th frame. Face alignment fails due to the oblique orientation. (e): The face region obtained from the previous frame via motion flow. (f): The skin detection result based on the region in (e).

and little strokes. Therefore, our algorithm can still handle images with many points and small strokes. One example is the "Rhone" style in Fig. 8.

C. Reconstruction

The reconstruction step is to reconstruct final A^+ from the previous obtained results. First, A_S^+ is obtained from A_S and A_E^+, where A_S^+ corresponds to A_S (the segmentation result of A). Then, from (3) and achieved A_P^+, $A^+ = A_S^+ + A_P^+$ is obtained. As we state previously, the synthesized A^+ is in the luminance channel. Finally, by combining A^+ and the components of the input image A in the I and Q channels, the final colorful stylized result is obtained.

Among these three steps, steps 2 and 3 can be performed straightforwardly. We discuss step 1 in more details here. Denote that $A_E^+ = [G_x, G_y]$. A_S^+ can be achieved by solving such an a least-square (LS) problem that minimizes

$$J\left(A_S^+\right) = \sum_{(x,y)} \left[\gamma \left(A_S^+(x,y) - A_S(x,y)\right)^2 + \left(\left(A_S^+\right)_x - G_x\right)^2 + \left(\left(A_S^+\right)_y - G_y\right)^2 \right], \quad (10)$$

where γ is a constant, and $(A_S^+)_x, (A_S^+)_y$ are partial derivatives of A_S^+ in the x and y direction, respectively. In all experiments, we set $\gamma = 0.01$.

The optimal A_S^+ of the above LS is the solution of

$$\gamma A_S^+ + A_S^+ D_x^T D_x + D_y^T D_y A_S^+ = \gamma A_S + G_x D_x + D_y^T G_y, \quad (11)$$

where D_x and D_y are the 1D differential operators, such that $A_S^+ D_x^T = (A_S^+)_x$ and $D_y A_S^+ = (A_S^+)_y$. This equation is of the form of the Lyapunov matrix equation $AX + XA = B$, where X is unknown and A and B are given.

V. Video Stylization and Personalization

As an application, we extend the above algorithm from single image to video. Especially, inspired by EasyToon [20], we mainly focus on life video which contains personal activities. Because style transfer will partly blur and distort the target image, we also develop methods to keep face clear and distinguishable in the video.

A. Stylization With Temporal Coherence

Separately stylizing each frame of a video will lead to the "flickering" effect and make the result twinkling and vague. To avoid this defect, for two neighbor frames V_{f1} and V_{f2}, where V_{f1} has been stylized while V_{f2} has not, we design and implement the following four steps.

(1) Optical flow [25] is employed to find pixel-level correspondences between neighbor frames. (2) For each image patch in V_{f2}, the corresponding patch in V_{f1} is defined as the same shape patch which contains the most corresponding pixels. (3) The SSD is computed between corresponding patches in V_{f1} and V_{f2}, and a threshold is set to decide the validation of patch-level correspondence. (4) If a patch in V_{f2} has the valid corresponding patch in V_{f1}, it directly keeps the same style component as its corresponding patch. Otherwise, the patch should get component from the dictionary independently as described in Section IV. These four steps successfully solve the "flickering" problem and reduce the running time significantly since a large number of patches only need to copy style components from previous results. Moreover, we adopt the occlusion detection and bilateral diffusion in [26] to deal the problem of the "dragging" effect.

B. Face Extraction and Blending

For both stylization and personalization goals, we segment and process face regions specifically to keep human face clean and visible in artistic style. First, a dynamic cascade based face detection [27] is used to detect face region in each frame. Then an active shape model based face alignment [28] is used to locate 87 face landmarks if succeed (see the white points in Fig. 6(b)). To extract the entire area of the face region, we use skin detection [29] to segment the whole face region. The skin detection is formulated in a single Gaussian model that

$$p(c|skin) = \frac{1}{2\pi|\Sigma_S|^{\frac{1}{2}}} e^{-\frac{1}{2}(c-\mu_s)^T \Sigma_S^{-1}(c-\mu_s)}, \quad (12)$$

where c is a color vector of a pixel in the detected area. The distribution parameters are

$$\mu_s = \frac{1}{n}\sum_{i=1}^{n} c_i, \quad \Sigma_S = \frac{1}{n-1}\sum_{i=1}^{n}(c_i - \mu_s)(c_i - \mu_s)^T, \quad (13)$$

where n is the number of training pixel color vector c_i. We set a threshold for $p(c|skin)$ to determine the skin pixel (Fig. 6(c)). It is general and unavoidable that face alignment might fail in some frames. To address this problem, we reuse the optical flow information to trace the corresponding area of the face in the previous frame when face alignment fails in the current frame.

Fig. 7. Two source images and several template images.

Fig. 8. Comparison of the results of image analogies (the upper row) and our method (the under row). The style from left to right: Pastel, Rhone, Craquelure, Freud, and Watercolor.

After this step, we can get some regions of the face parts. We then set these parts as training data and employ skin detection near these regions (see Fig. 6(c)–(f)).

When the face regions are segmented for all frames, the final stylized face F_{final} is obtained by simply combining the completely stylized face F' and not stylized face F as:

$$F_{final} = \gamma F + (1-\gamma)F'. \tag{14}$$

where γ is a coefficient between 0 and 1 to control the amount of personal information on the final stylized face. Finally, face blending is applied to synthesize a personalized artwork from the face and stylized non-facial part. Let p be a pixel belonging to the blending region. The blending result for p is

$$I_{final}(p) = \frac{D_p}{D_p + D'_p} A^+(p) + \frac{D'_p}{D_p + D'_p} F_{final}(p), \tag{15}$$

where D_p is the distance between p and the face region, D'_p is the distance between p and the non-facial region, A^+ is the completed stylized image, and F_{final} is the final stylized face. This method is much simpler than Poisson blending [20].

VI. Experiments

In this section, we test our method on a variety of source images and template styles, and compare it with the image analogies approach [5].[1] There are two reasons to compare our algorithm with image analogies. First, image analogies is known to be the best example-based image stylization algorithm so far. Considering the space limitation, we only compare our algorithm with the best performance. Second, since we focus on the automatic image stylization problem for a given arbitrary style template image, we do not compare our algorithm with the ones such as [6], [9], [14] that need manual input or other different input (please refer to Table I).

[1]To our best knowledge, there is no quantitative metric for evaluating the results of image stylization. We evaluate the results visually.

Two source images and several style template images are given in Fig. 7. There are five template images, called Pastel, Rhone, Craquelure, Freud, and Watercolor, respectively. The Freud style contains many strokes since it is an oil painting, while the Watercolor style contains brushes and diffusion. The difference can be well found on the screen via zooming by, say, 500%. Note that image analogies requires the ground-truth image B, while our method does not.

The comparison results of our algorithm and image analogies for source image 1 are given in Fig. 8.[2] From the results we can see that our results have better appearances than those of image analogies. The content of the source image is changed much more by using image analogies than our algorithm. For example, for the "Pastel" and "Rhone" styles (Fig. 8), image analogies changes the content greatly, while our algorithm preserves the content better. This is because in our algorithm, the draft component representing the image content is extracted and kept unchanged, while in image analogies, no similar scheme is used to preserve the content.

On the other hand, our algorithm synthesizes the style better than image analogies. Because no linear mapping can be found to align the luminance distributions of the input source and style template image perfectly, image analogies cannot synthesize some styles well. The distribution of style patterns relies strongly on the luminance in the result of image analogies. The result on a region with homogenous luminance has the trend of being one with a homogenous pattern. For example, for the style "Craquelure" (see Fig. 8), there are weak patterns in the dark regions in the result of image analogies.

The comparison results for source image 2 are given in Fig. 9. From the results we can see that our algorithm preserves the content better than image analogies after style transfer. It is obviously observed that the color of the flowers in the image is

[2]We use the executable program of image analogies provided by the authors, available at http://www.mrl.nyu.edu/projects/image-analogies/.

Fig. 9. Comparison of the results of image analogies (the upper row) and our method (the under row). The style from left to right: Pastel, Rhone, Craquelure, Freud, and Watercolor.

Fig. 10. Example results by our algorithm. Row 1: The source video frames. Row 2: The stylized output in watercolor, faces are obscure. Row 3: The result artwork in watercolor after stylization and personalization. Row 4: The stylized and personalized artwork in oil-on-canvas style.

better preserved by our algorithm than image analogies, especially for the style "Pastel" and "Watercolor".

We have conducted a user study with 20 users (students). For each user, we have given the style transfer results of our algorithm and image analogies with 7 source images and 7 template images. Therefore, each user has 98 result images (49 our results and 49 image analogies') to evaluate. The users do not know which algorithm does each result is obtained by. Then each user is required to score each result from 1–5 (from bad to excellent). Finally, the average score is 4.4 for our algorithm, and 4.0 for image analogies. Our algorithm obtains the score 0.4 higher than image analogies.

Besides, an example of video stylization and personalization is shown in Fig. 10. It is evidently shown that the same object in different frames keeps the constant style texture, and the human face is clear and smoothly transiting along boundaries. More results of our algorithm are provided in our website.[3]

[3]http://mmlab.siat.ac.cn/personal/style/supply/results

In our experiments, our algorithm is implemented in Matlab. In the PC with AMD Athlon dual core 2.5 GHz and 1 GB RAM, the average running time for a 640 $\times$ 480 image is about 1 minute for the non-optimized Matlab code. Without the dictionary reduction of representative sample selection, it takes more than 5 minutes to process the images with the same size. Without the coarse-to-fine BP, the needed running time is 10000 times that of our implementation, without considering the memory. This is impractical.

VII. Conclusion

In this paper, we have proposed an image component analysis based approach to transferring the style of an artistic image to real photographs. In our approach, three components, including the draft, paint, and edge component, are used to describe the content, main style information, and information along the boundaries of an image, respectively. Our algorithm preserves the content of the source image and synthesizes the style by copying style patches from the template. The patch copying process is formulated as a global optimization problem using Markov random fields, and the optimization problem is solved using a coarse-to-fine belief propagation.

We further extend our approach to create video stylization and personalization artwork. Inspired by EasyToon [20], we propose a general framework to create personalized artworks with temporal coherence from videos.

We find some styles that our current framework does not work well on. One example is highly abstract artworks, e.g., the style of Picasso's surrealistic paintings. To deal with these problems, human interaction may be incorporated. It is attractive to continue our research in this direction.

References

[1] B. Gooch and A. Gooch, *Non-Photorealistic Rendering*. Natick, MA, USA: AK Peters, 2001.
[2] A. Hertzmann, "A survey of stroke-based rendering," *IEEE Comput. Graph. Applicat.*, vol. 23, no. 4, pp. 70–81, Jul.–Aug. 2003.
[3] C. Curtis, S. Anderson, J. Seims, K. Fleischer, and D. Salesin, "Computer-generated watercolor," in *Proc. ACM Conf. Computer Graphics and Interactive Techniques (SIGGRAPH)*, 1997.
[4] D. DeCarlo and A. Santella, "Stylization and abstraction of photographs," in *Proc. ACM Conf. Computer Graphics and Interactive Techniques (SIGGRAPH)*, 2002.
[5] A. Hertzmann, C. E. Jacobs, N. Oliver, B. Curless, and D. H. Salesin, "Image analogies," in *Proc. ACM Conf. Computer Graphics and Interactive Techniques (SIGGRAPH)*, 2001.

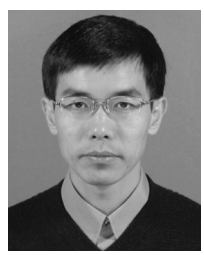
[6] L. Cheng, S. Vishwanathan, and X. Zhang, "Consistent image analogies using semi-supervised learning," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2008.
[7] R. Resales, K. Achan, and B. Frey, "Unsupervised image translation," in *Proc. Int. Conf. Computer Vision*, 2003.
[8] W. Freeman, E. Pasztor, and O. Carmichael, "Learning low-level vision," *Int. J. Comput. Vision*, vol. 40, no. 1, pp. 25–47, 2000.
[9] B. Wang, W. Wang, H. Yang, and J. Sun, "Efficient example-based painting and synthesis of 2D directional texture," *IEEE Trans. Visualiz. Comput. Graph.*, vol. 10, no. 3, pp. 266–277, 2004.
[10] C. Cao, S. Chen, W. Zhang, and X. Tang, "Automatic motion-guided video stylization and personalization," in *Proc. ACM Int. Conf. Multimedia*, 2011.
[11] X. Tang and X. Wang, "Face sketch synthesis and recognition," in *Proc. Int. Conf. Computer Vision*, 2003.
[12] X. Wang and X. Tang, "Face photo-sketch synthesis and recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 11, pp. 1955–1967, 2009.
[13] W. Zhang, X. Wang, and X. Tang, "Lighting and pose robust face sketch synthesis," in *Proc. Eur. Conf. Computer Vision*, 2010.
[14] I. Drori, D. Cohen-Or, and H. Yeshurun, "Example-based style synthesis," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2003.
[15] S. Bae, S. Paris, and F. Durand, "Two-scale tone management for photographic look," *ACM Trans. Graph.*, vol. 25, pp. 637–645, 2006.
[16] G. Ramanarayanan and K. Bala, "Constrained texture synthesis via energy minimization," *IEEE Trans. Visualiz. Comput. Graph.*, vol. 13, no. 1, pp. 167–178, 2007.
[17] K. Zhou, P. Du, L. Wang, Y. Matsushita, J. Shi, B. Guo, and H. Shum, "Decorating surfaces with bidirectional texture functions," *IEEE Trans. Visualiz. Comput. Graph.*, vol. 11, no. 5, pp. 519–528, 2005.
[18] J. Wang, Y. Xu, H. Y. Shum, and M. F. Cohen, "Video tooning," in *Proc. ACM Conf. Computer Graphics and Interactive Techniques (SIGGRAPH)*, 2004.
[19] A. Bousseau, F. Neyret, J. Thollot, and D. Salesin, "Video watercolorization using bidirectional texture advection," in *Proc. ACM Conf. Computer Graphics and Interactive Techniques (SIGGRAPH)*, 2007.
[20] S. Chen, Y. Tian, F. Wen, Y. Xu, and X. Tang, "EasyToon: An easy and quick tool to personalize a cartoon storyboard using face photos," in *Proc. ACM Int. Conf. Multimedia*, 2008.
[21] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 5, pp. 603–619, 2002.
[22] A. A. Efros and W. Freeman, "Image quilting for texture synthesis and transfer," in *Proc. ACM Conf. Computer Graphics and Interactive Techniques (SIGGRAPH)*, 2001.
[23] V. Kwatra, A. Schodl, I. Essa, G. Turk, and A. Bobick, "Graphcut textures: Image and video synthesis using graph cuts," in *Proc. ACM Conf. Computer Graphics and Interactive Techniques (SIGGRAPH)*, 2003.
[24] T. Huang, S. Chen, J. Liu, and X. Tang, "Image inpainting by global structure and texture propagation," in *Proc. ACM Int. Conf. Multimedia*, 2007.
[25] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert, "High accuracy optical flow estimation based on a theory for warping," in *Proc. Eur. Conf. Computer Vision*, 2004.
[26] J. Xiao, H. Cheng, H. Sawhney, C. Rao, and M. Isnardi, "Bilateral filtering-based optical flow estimation with occlusion detection," in *Proc. Eur. Conf. Computer Vision*, 2006.
[27] R. Xiao, H. Zhu, H. Sun, and X. Tang, "Dynamic cascades for face detection," in *Proc. Int. Conf. Computer Vision*, 2007.
[28] Y. Zhou, L. Gu, and H. J. Zhang, "Bayesian tangent shape model: Estimating shape and pose parameters via Bayesian inference," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2003.
[29] V. Vezhnevets, V. Sazonov, and A. Andreeva, "A survey on pixel-based skin color detection techniques," in *Proc. Graphicon*, 2003.

Wei Zhang received the B.Eng. degree in electronic engineering from the Tsinghua University, Beijing, China, in 2007, the M.Phil. degree in 2009, and Ph.D. degree in 2012, both in information engineering from The Chinese University of Hong Kong. He was among the top ten Kaggle data scientists from July to November, 2012. His research interests include machine learning and computer vision.

Chen Cao received the B.E. degree from the University of Science and Technology of China, Hefei, in 2011. From July 2011 to July 2012, he was a research assistant at the Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, China. He is currently an M.S. student at the University of Florida, Gainesville, FL. His research interests include image/video processing and pattern recognition.

Shifeng Chen received the B.E. degree from the University of Science and Technology of China, Hefei, in 2002, the M.Phil. degree from City University of Hong Kong, Hong Kong, in 2005, and the Ph.D. Degree from the Chinese University of Hong Kong, Hong Kong, in 2008. He is now an Associate Professor in the Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, China. His research interests include image processing and computer vision.

Jianzhuang Liu (M'02–SM'02) received the Ph.D. degree in computer vision from The Chinese University of Hong Kong, Hong Kong, in 1997. From 1998 to 2000, he was a research fellow with Nanyang Technological University, Singapore. From 2000 to 2012, he was a postdoctoral fellow, then an assistant professor, and then an adjunct associate professor with The Chinese University of Hong Kong. He joined Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, as a professor, in 2011. He is currently a chief scientist with Huawei Technologies Co. Ltd., Shenzhen, China. He has published more than 100 papers, most of which are in prestigious journals and conferences in computer science. His research interests include computer vision, image processing, machine learning, multimedia, and graphics.



Xiaoou Tang (S'93–M'96–SM'02–F'09) received the B.S. degree from the University of Science and Technology of China, Hefei, in 1990, and the M.S. degree from the University of Rochester, Rochester, NY, in 1991. He received the Ph.D. degree from the Massachusetts Institute of Technology, Cambridge, in 1996.

He is a Professor in the Department of Information Engineering, Chinese University of Hong Kong. He worked as the group manager of the Visual Computing Group at the Microsoft Research Asia from 2005 to 2008. His research interests include computer vision, pattern recognition, and video processing.

Dr. Tang received the Best Paper Award at the IEEE CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION (CVPR) 2009. He is a program chair of the IEEE INTERNATIONAL CONFERENCE ON COMPUTER VISION (ICCV) 2009 and an Associate Editor of IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE (PAMI) and International Journal of Computer Vision (IJCV).