Poisson Vector Graphics (PVG)-Guided Face Color Transfer in Videos

Qian Fu, *Nanyang Technological University, Singapore, 639798, Singapore*

Ying He, *Nanyang Technological University, Singapore, 639798, Singapore*

Fei Hou, *Chinese Academy of Sciences, Beijing, 100049, China*

Qian Sun, *Tianjin University, Tianjin, 300350, China*

Anxiang Zeng, *Alibaba Group, Beijing, 100020, China*

Zhenchuan Huang, *Alibaba Group, Beijing, 100020, China*

Juyong Zhang, *University of Science and Technology of China, Anhui, 230026, China*

Yong-Jin Liu, *Tsinghua University, Beijing, 100084, China*

This article presents a simple yet effective algorithm for automatically transferring face colors in portrait videos. We extract the facial features and vectorize the faces in the input video using Poisson vector graphics, which encodes the low-frequency colors as the boundary colors of diffusion curves, and the high-frequency colors as Poisson regions. Then, we transfer the face color of a reference image/video to the first frame of the input video by applying optimal mass transport between the boundary colors of diffusion curves. Next the boundary color of the first frame is transferred to the subsequent frames by matching the curves. Finally, with the original or modified Poisson regions, we render the video using an efficient random-access Poisson solver. Thanks to our efficient diffusion curve matching algorithm, transferring colors for the vectorized video takes less than 1 millisecond per frame. Our method is particularly desired for frequent transfer from multiple references due to its information reuse nature. The simple diffusion curve matching also greatly improves the performance of video vectorization, since we only need to solve an optimization problem for the first frame. Since our method does not require correspondence between the reference image/video and the input video, it is flexible and robust to handle faces with significantly different geometries and postures, which often pose challenges to the existing methods. Moreover, by manipulating Poisson regions, we can enhance or reduce the highlight and contrast so that the reference color can fit into the input video naturally. We demonstrate the efficacy of our method on image-to-video transfer and color swap in videos.

In the digital age, portrait and self-portrait photographs and videos are extremely popular and have spread to every corner of the world. There are many programs and apps allowing the user to retouch the photos, such as editing eyes, brightening skin, removing wrinkles, reshaping face, etc. However, editing videos is still a challenging task.

In this article, we are interested in automatically transferring face color from a reference image/video to an input video. Though color transfer has been studied in the computer vision and graphics community for almost two decades, there are only a few

0272-1716 © 2020 IEEE
Digital Object Identifier 10.1109/MCG.2020.3024870
Date of publication 18 September 2020; date of current version 10 December 2021.

works for portrait photos. Shu *et al.*[19] developed a method to transfer both lighting and colors for portrait photos. They applied shape from a shading technique to estimate the face normal and then formulated the transfer problem as an eight-dimensional (8-D) optimal mass transport (OMT), considering colors, pixel coordinates, and face normals. Though their method produces excellent results, it takes a few minutes to process an HD image. Such a high computational cost diminishes its application to video color transfer. Fu *et al.*[2] proposed a vectorization-based method that reduces the region color transfer problem to a boundary color transfer, hereby is efficient. They solved a Poisson's equation to render the vector graphics. However, since their results are highly sensitive to the PDE's boundary conditions, directly applying their method to video leads to unpleasing visual artifacts, such as flickering.

To overcome the challenges of the existing work, we propose a simple yet effective method for transferring face color in videos. We extract the facial features and vectorize the face(s) of the input video using Poisson vector graphics, which encodes the low-frequency colors as the boundary colors of diffusion curves (DCs), and the high-frequency colors in Poisson regions (PRs). Then, we transfer the face color of a reference image/video to the first frame of the input video by applying OMT between the boundary colors of DCs. Next the DC boundary color of the first frame is transferred to the subsequent frames by matching the DCs. Finally, with the original or modified PRs, we render the video using an efficient Poisson solver. Thanks to our highly efficient DC matching algorithm, transferring colors for the vectorized video takes less than 1 millisecond per frame. Since we only apply OMT to the first frame, our method is computationally efficient and stable. For a single transfer from one reference image, the absolute performance of our algorithm, including precomputation (i.e., vectorization) and postcomputation (i.e., rendering) is comparable to the conventional raster-image-based method, which simply applies OMT to all frames. For frequent transfer from multiple reference images, our method runs one or two orders of magnitude faster than the image OMT method, due to its information reuse nature. The simple DC matching algorithm also greatly improves the performance of video vectorization, since we only need to solve an optimization problem for the first frame. Since our method does not require correspondence between the reference image/video and the input video, it is flexible and robust to handle faces with significantly different geometries and postures, which often pose challenges to the existing methods. Moreover, by manipulating the PRs, we can enhance or reduce the highlight and contrast so that the reference color can fit into the input video naturally. We demonstrate the efficacy of our method on image-to-video transfer and color swap in videos.

We make the following contributions in this article.

- A simple yet effective method for vectorizing portrait videos using Poisson vector graphics; the method extracts only salient facial features as a set of sparse DCs and encodes the high-frequency details as PRs.
- An efficient method for transferring skin colors between a reference image and the input video; given an accurate skin segmentation, it can produce reliable transfer results and is robust to lighting conditions. It also allows information reuse, and has significant advantages over the existing methods for multiple transfers on high-resolution videos.

RELATED WORK

Color transfer has been studied extensively since 2000. Reinhard *et al.*[15] pioneered the color transfer method that matches the mean and standard deviation of color distributions of the source and reference images. Hwang *et al.*[10] applied probabilistic moving least square for color transfer between images with the same scene but different camera settings and illumination conditions. Shih *et al.*[17] proposed a style transfer technique for portraits that can match the local contrast and the overall lighting direction by transferring the local statistics. Their method can produce visually pleasing results; however, it requires a dense correspondence between the source and the reference image. Shu *et al.*[19] developed a relighting and color transfer method for headshots by solving OMT for 8-D data including RGB colors, pixel positions, and normals. Fu *et al.*[2] proposed a vectorization-based method for transferring color between portrait images. However, directly applying their method to videos is time consuming and produces results with visual artifacts.

DC vectorization aims at converting a raster image into a set of curve-based vector primitives. Orazan *et al.*[13] extracted DCs using Canny edges and fitted their boundary colors by a least square method. Xie *et al.*[20] developed an automatic method for vectorizing raster images with hierarchical DCs. Their method is efficient, accurate, and robust, working for both art and natural images. However, it often extracts many curves, making post-editing difficult. The proposed PVG vectorization only takes the salient feature

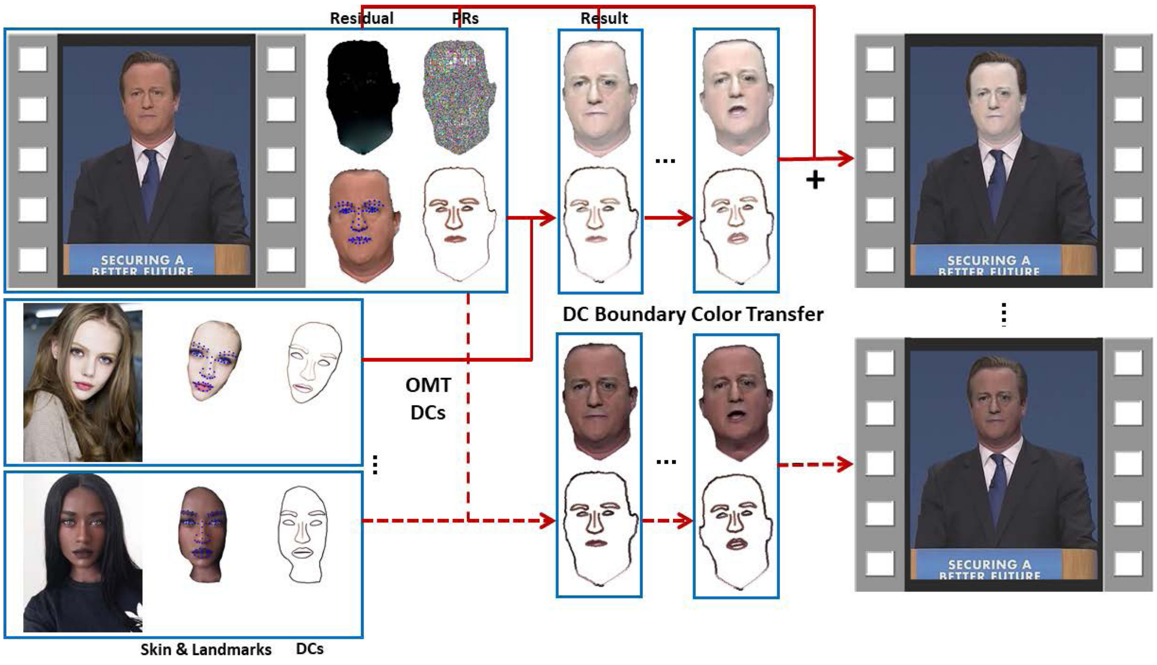

FIGURE 1. Algorithmic pipeline. Thanks to its information reuse nature, our algorithm is highly desired for frequent color transfer from multiple reference images.

curves as DCs, which present the low-frequency skin colors only. Furthermore, our method is also efficient, since we only need to vectorize the first frame of the input video. The boundary colors of DCs for the subsequent frames are obtained by a simple DC matching algorithm.

Style transfer is also a closely related problem. Recent years have witnessed great success of deep learning techniques (e.g., the work by He *et al.*[4]). However, these methods either focus on nonphotorealistic style or require a dense correspondence map between the input and the reference. To our knowledge, there is no deep learning approaches designed for face color transfer in videos.

Generative adversarial network (GAN)[3] is a powerful generative model. Karras *et al.*[11] proposed a style-based generator that uses face color as a high-level attribute and transfers it to target images. The GAN approaches do not require semantic segmentation and can generate realistic images. However, training a GAN is difficult due to large amount of training data and some model-related issues, such as diminished gradient, nonconvergence, and mode collapse. Though some approaches, such as CycleGAN,[9] can be used for transfer color between images by taking color as an attribute, it is unclear whether they can be used for videos. Our method, built upon PVG vectorization and OMT, is conceptually simple and easy to implement.

OVERVIEW

As a precomputation method, our method first vectorizes the input video and then performs color transfer for arbitrary reference image/video.

To vectorize the video, we first detect the facial landmarks, and obtain a face mask by extracting the skin region for each frame. Next we use the extracted facial landmarks and the boundary of the face mask as DCs,[13] whose boundary colors are computed by solving a linear least square problem. We encode the low-frequency skin colors using DCs and the high-frequency details using PRs.[7] We refer the reader to the supplementary material, which is available in the IEEE Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/MCG.2020.3024870, for more details about DCs and PRs. The reference image is also vectorized using Poisson vector graphics.

To transfer face colors, we adopt the OMT model that matches the color distribution of the source and the reference in an exact manner. Note that naïvely applying OMT to all frames is obviously time consuming. In our approach, we apply OMT only to the first frame. Then, taking advantage of the vectorized video, we transfer the DC boundary colors to the subsequent frames by our DC matching algorithm, which is efficient and can also guarantee the smoothness of the DC boundary colors in-between frames, hereby effectively reducing flickering artifacts.

Figure 1 shows the pipeline of our color transfer method and Algorithm 1 presents the pseudocode. The most distinctive feature of our method is information reuse. For frequent transfer from multiple reference images, our method runs one or two orders of magnitude faster than the image OMT method.

VIDEO VECTORIZATION

Vectorization plays a critical role in our method. A naïve method is to simply apply the image vectorization method (e.g., the work by Fu *et al.*[2]) to each frame. Such an approach, however, is time consuming, since one has to solve a least square problem n times. In this article, we propose a simple yet effective method for vectorizing face videos. Our key idea is to solve the optimization problem only for the *first* frame and then transfer the boundary colors of DC to the subsequent frames by matching the curves. This simple strategy kills two birds with one stone. First, since the computational cost of DC matching is significantly less than that of solving the least square problem, our method is much faster than repeating the image vectorization method. Second, it allows us to easily solve some challenging situations, such as large changes of lighting conditions.

Extracting Facial Features

Facial landmark detection, also known as face alignment, is to locate a set of predefined Fiducial points on 2-D faces. As a core component of various face applications, it has been studied extensively in the last decade. In our work, we implement a simple yet effective end-to-end deep neural network for robust facial landmark localization. We adopt MobileNetV2[16] as the feature extractor with regression output layers. To deal with imbalanced facial pose datasets, we use online hard example learning.[18] To further improve the accuracy, we adopt the knowledge distillation method[6] that migrates knowledge from the high-precision model to the real-time model by collecting a large amount of unlabeled data. Our model is compact with a size of 2.2 Mb and can run 125 fps per face for input size 160×160 on iPhone 7 Plus. Figure 2(b) shows the detected 106 facial landmarks. Using the detected facial landmarks, we crop the input video and vectorize only the region of interest.

To extract the face mask, we adopt the YCbCr color space, where Y is the luma component and Cb and Cr are the blue-difference and red-difference

Algorithm 1. PVG-Guided Face Color Transfer for Videos

```
Require: The input video with $n$ frames $I = \{I_i\}_{i=1}^n$, a
         reference image $R$, and (optional) $\lambda \in [0.7, 1.5]$ the
         PR coefficient.
Ensure: The output video $I' = \{I'_i\}_{i=1}^n$ with transferred face
        color
  ▷ Precomputation: vectorize $I$
  for $i = 1 : n$ do
    Extract facial features and skin mask for $I_i$
    Construct DCs $\gamma_i$ using the extracted features
    if $i > 1$ then
      Compute $\pi_{i-1} : \gamma_{i-1} \to \gamma_i$
    end if
  end for
  Compute $g_1$ for $I_1$ by solving (3)
  Compute PRs $f_1$ for $I_1$ using (4)
  for $i = 2 : n$ do
    $g_i = g_{i-1}\left(\pi_{i-1}^{-1}(\gamma_i)\right)$
    Compute PRs $f_i$ for $I_i$ using (4)
  end for
  ▷ Transfer DC boundary colors
  Transfer color to $I_1$ using OMT $g'_1 = \psi(g_1, g_r)$
  for $i = 2 : n$ do
    Transfer color to $I_i$ using $g'_i = g'_{i-1}\left(\pi_{i-1}^{-1}(\gamma_i)\right)$
  end for
  ▷ (Optional) Edit contrast and highlight
  for $i = 1 : n$ do
    Update PRs for $I_i$ by $f'_i = \lambda f_i$
  end for
  ▷ Render vector primitives
  for $i = 1 : n$ do
    Compute $I'_i$ by solving (1) with $f'_i$ and $g'_i$
  end for
```

chroma components. As pointed out by Hsu *et al.*,[8] skin colors are concentrated in the YCbCr space. Specifically, the pixels in the CrCb subspace follow an approximately elliptical distribution, and the YCbCr color space does not depend on luminance. Based on this observation, we set a range [140,160] for the Cr component to extract the domain of skin. Since the domain may be disconnected and contain nonfacial part, we adopt a simple heuristic by taking the largest connected component. Then, we take its boundary as the face mask [see Figure 2(c)]. Furthermore, the probability edge extraction[12] can help us obtain more robust skin mask from some cases whose skin color is not located in the Cr range but clearly distinguished from the background, e.g., Figure 7 bottom row. We observe this simple technique works very well for face videos with various colors and postures.

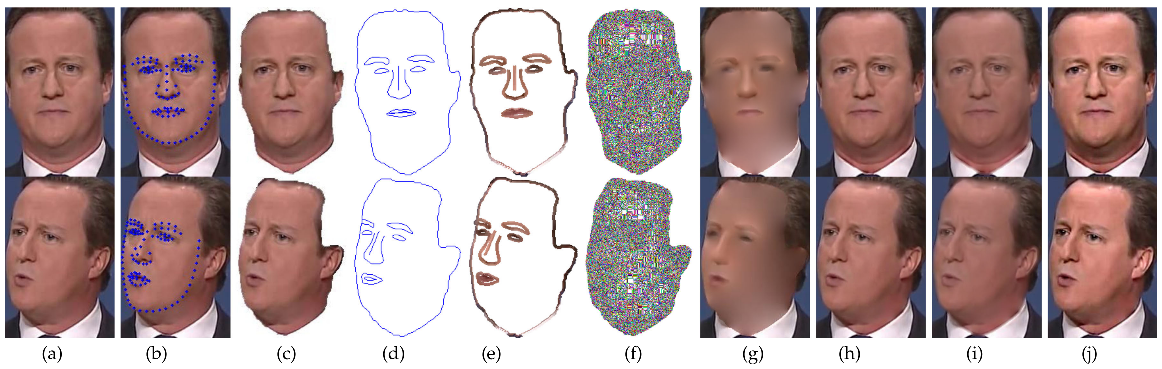

FIGURE 2. Face vectorization. (a) Two frames of an input video. We detect 106 feature points for each frame (b) and obtain a face mask by extracting skin colors (c). We use the feature points and the boundary of the mask as DCs (d). We compute their boundary colors (e) by solving a linear least square problem. To improve the performance, we use harmonic B-spline solver which is built upon a quad-tree data structure. DCs encode the low-frequency (LF) colors (g) and PRs encode the high-frequency (HF) details. (f) Color-coded PRs. Using DCs and PRs, we can faithfully reconstruct the input video (h). Moreover, by changing the coefficients of PRs, we can edit contrast and highlight (i) and (j). (a) Input. (b) Feature points. (c) Skin mask. (d) DCs γ. (e) Boundary colors g. (f) PRs. (g) LF. (h) LF+HF. (i) LF + HF$\times 0.7$. (j) LF + HF$\times 1.5$.

Computing Boundary Color g for DCs

Given an image I and the specified feature curves consisting of the facial landmarks and the boundary of the skin mask, our algorithm assigns colors of DCs and evaluates PRs automatically. The DCs represent significant boundaries containing the main color derived from the image boundaries. The PRs represent highlights/shadows and geometric details derived from image regions (skin mask in our implementation) with pixel Laplacians. Figure 2 illustrates the face vectorization algorithm.

To render and update the vector images, we solve the Poisson's equation

$$\begin{cases} \Delta u(\mathbf{x}) = f, & \mathbf{x} \in \Omega \backslash \partial\Omega \\ u(\mathbf{x})|_{\partial\Omega} = g, & \mathbf{x} \in \partial\Omega \end{cases} \tag{1}$$

where f is the Laplacian constraints (PRs), g is the Dirichlet boundary condition of colors (the boundary colors are defined on the DCs) and Ω is a 2-D compact domain.

Using harmonic B-splines,[1] we can explicitly express the solution of Poisson's equation as

$$u(\mathbf{x}) = \sum_{i=1}^{n} \lambda_i \phi_i(\mathbf{x}) \tag{2}$$

where λ_i is the control coefficient (which will be solved later), and $\phi_i(\mathbf{x})$ is the basis function of harmonic B-splines.

To determine the color g of boundaries γ, we build a quad tree to tessellate the skin region and then construct a harmonic B-spline whose basis functions $\phi_i(\mathbf{x})$ are defined on the nodes of the quad tree. Then, we compute the control coefficients λ_i by fitting the input image I in a linear least square

$$\arg\min_{\lambda_j} \left(\sum \lambda_j \phi_j(\mathbf{x}) - I \right)^2. \tag{3}$$

We obtain the boundary color g by evaluating the color function u on the boundaries $g = u|_\gamma$.

After that, we compute the Laplacian constrains f by applying the Laplacian operator to all interior pixels

$$f = \Delta(I|_\Omega + g). \tag{4}$$

Figure 3 illustrates the idea of Poisson vector graphics guided color editing. The image is vectorized into DCs for low-frequency colors and PRs for high-frequency details. To change the color, we simply modify the boundary colors of the DCs and keep all PRs unchanged. Since there are only a few DCs in PVG vectorized results, editing them is easier than the traditional vectorization approaches (e.g., Xie *et al.*[20]), which usually produce a large amount of DCs.

We apply the above method only to the first frame of the input video and obtain $g_1 : \gamma_1 \rightarrow \mathbb{R}^3$. For the subsequent frames, we do not need to compute the boundary colors g_i, $i > 1$, for the DCs. Instead, we match the DCs and then transfer g from the first frame to the remaining frames. This simple strategy not only reduces computational cost significantly, but also ensures the skin colors are consistent in all frames.

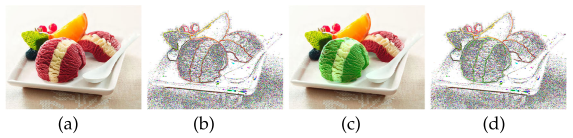

FIGURE 3. Poisson vector graphics guided color editing. The ice cream image (a) is vectorized into a set of DCs (colored, solid lines) and PRs (dotted regions) (b), where the former encodes the low-frequency color and the latter encodes the high-frequency details. To change the ice cream's color, we only need to edit the boundary colors of the DCs (d). Diffusing the new boundary colors produces an image with a totally new looking (c). Intuitively speaking, we formulate the region-based color editing into a problem of editing the color of its boundary. The only difference between (b) and (d) is the boundary colors of DCs. (a) Original. (b) DCs and PRs. (c) Result. (d) DCs with new boundary colors.

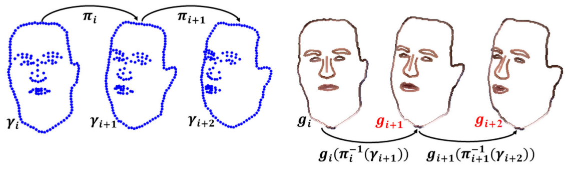

FIGURE 4. DC boundary color matching. The boundary color of the first frame g_1 is computed by (3). With the boundary color g_i for the ith frame and the correspondence map $\pi_i : \gamma_i \to \gamma_{i+1}$, we compute the boundary color g_{i+1} for the $(i+1)$th frame by the inverse map π_i^{-1}.

The 106 facial landmarks are semantically labeled, hence providing a one-to-one correspondence between frames. Denote by $\pi_i : \gamma_i \to \gamma_{i+1}$ the map between two consecutive frames. Given the boundary color $g_i : \gamma_i \to \mathbb{R}^3$ of the ith frame, we compute the boundary color of the $(i+1)$th frame by

$$g_{i+1} = g_i\left(\pi_i^{-1}(\gamma_{i+1})\right) \tag{5}$$

where $i = 1, 2, \ldots, n$. Figure 4 illustrates the above method for matching DCs' boundary color.

PVG-GUIDED COLOR TRANSFER

OMT is a powerful technique for image-to-image color transfer. Denote by I and J the input and reference images with N_I and N_J pixels, respectively. It computes a transport map between the input color distribution $I = \{c_i | c_i \in \mathbb{R}^{3\times1}\}_{i=1}^{N_I}$ and the reference color distribution $J = \{c_j | c_j \in \mathbb{R}^{3\times1}\}_{j=1}^{N_J}$ with a minimum

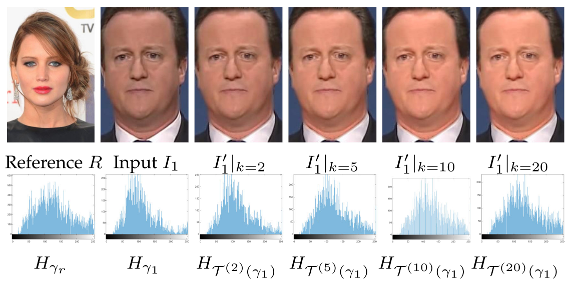

FIGURE 5. Iterative computation of the OMT map $\mathcal{T}$. We show a few representative intermediate results $I_1'|_k$ of the first frame in the top row and their corresponding histograms $H_{\mathcal{T}^{(k)}(\gamma_1)}$ in the bottom row. The iterative algorithm converges quickly and produces the final result in only 20 iterations.

cost, where c_i and c_j are the pixel colors. To ensure all the instances of a color in the input are transferred to a single color, we adopt the Monge's formulation to compute the transport $\mathcal{T} : \{c_i\} \to \{c_j\}$:

$$\arg\min_{\mathcal{T}} \sum_{i=1}^{N_I} H_{c_i} \|c_i - \mathcal{T}(c_i)\|^2 \tag{6}$$

with constraints $H_{\mathcal{T}(I)} = H_J$, where H_I and H_J are the normalized histograms of I and J.

Since Monge's formulation may not always produce a solution, we adopt the sliced Wasserstein distance algorithm[14] to obtain an approximate solution $\psi(I, J)$ such that $H_{\mathcal{T}(I)} \approx H_J$

$$\begin{aligned} \mathcal{T}^{(k+1)}(I) &= (1-\beta)\mathcal{T}^{(k)}(I) \\ &\quad + \beta\theta M(\theta^{-1}\mathcal{T}^{(k)}(I), \theta^{-1}J) \end{aligned} \tag{7}$$

with initial condition $\mathcal{T}^{(1)}(I) = I$, where k is the iteration count, $\beta \in [0, 1]$ is a linear parameter balancing convergence speed and stability of the result, $\theta \in \mathbb{R}^{3\times3}$ is a randomly generated orthogonal matrix, and $M : \mathbb{R}^{3\times N_I} \times \mathbb{R}^{3\times N_J} \to \mathbb{R}^{3\times N_I}$ is the histogram matching operator.

A naïve implementation is to simply compute the OMT map between each frame and the reference image. This is obviously time consuming. In our method, we compute the OMT map only for the DCs of the first frame as follows:

$$\mathcal{T}^{(k+1)}(\gamma_1) = (1-\beta)\mathcal{T}^{(k)}(\gamma_1) + \beta\theta M(\theta^{-1}\mathcal{T}^{(k)}(\gamma_1), \theta^{-1}\gamma_r) \tag{8}$$

with initial condition $\mathcal{T}^{(1)}(\gamma_1) = \gamma_1$, where γ_1 and γ_r, respectively, represent the DCs of the first frame I_1 and the reference image R. In our experiment, we set

TABLE 1. Statistics. The timings were measured in milliseconds.

Video	Size	T_{R-OMT}	T_V	T_{V-OMT}	T_{DC-M}	T_R
	200×274	3262	3501	339	< 1	78
	206×307	2858	3482	348	< 1	93
	386×503	6640	5975	426	< 1	197
	316×457	4876	4976	353	< 1	172
	202×390	3412	3460	364	< 1	109
	322×392	4622	4559	356	< 1	141
	340×447	5640	5315	360	< 1	172

$\beta = 0.2$ and $N = 20$ which is sufficient to produce visually pleasing result (see Figure 5). Then, we compute the transferred boundary color g_1 as $g_1 = \psi(\gamma_1, \gamma_r) \triangleq T^{(N)}(\gamma_1)$.

The DC boundary colors of the next frame are obtained by transferring the boundary colors from the the previous frame $g_i = g_{i-1}(\pi_{i-1}^{-1}(\gamma_i))$, $2 \leq i \leq n$.

RESULTS

We implemented our algorithm in C++ and evaluated it on a PC with an Intel i7 CPU2.80 GHz. Table 1 reports the running time of our method and the raster-image-based method.

We examine the time complexity of our method, the naïve image-based OMT that computes OMT for all frames, and the extended Fu *et al.*'s method[2] that applies their vectorization-based method for all frames. Denote by n the number of frames, m the number of reference images, T_{R-OMT} the time for computing image-based OMT for one frame, T_V the time for vectorizing one frame, T_{V-OMT} the time for computing vectorization-based OMT for one frame, T_{DC-M} the time for matching DCs between adjacent frames, and T_R the time for rendering PVG.

The image-based OMT method takes mnT_{R-OMT} time, since it computes the OMT map for every frame and every reference. The extended Fu's method consists of three steps: vectorization takes nT_V time, since it solves the linear least square problem n times; color transfer takes mnT_{V-OMT} time, since it computes an OMT map for every frame and every reference; PVG rendering takes mnT_R time.

Our method also consists of three steps: vectorization takes only $T_V + (n-1)T_{DC-M}$ time, since we only solve the linear square problem for the first frame; color transfer takes $mT_{V-OMT} + m(n-1)T_{DC-M}$, since we compute the OMT map only for the first frame; PVG rendering takes mnT_R time. Since $T_{DC-M} \ll T_{R-OMT}$, $T_{DC-M} \ll T_{V-OMT}$, and $T_R \ll T_{R-OMT}$, our method has significant advantage for long videos (i.e., with large n) and multiple references (i.e., $m > 1$).

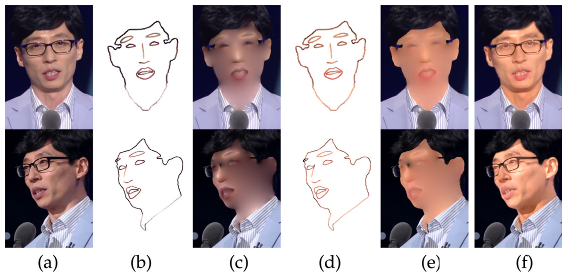

FIGURE 6. Transferring face colors to person wearing glasses. We extract the facial features to form the DCs and then compute their boundary colors by solving a linear least square for the first frame. DCs encode the low-frequency skin colors and PRs encode the high-frequency details (including glasses). Given a vectorized reference image (with or without wearing glasses), we only need to transfer the boundary colors between the DCs and keep the PRs unchanged. Due to change of camera positions, the resolutions of the two facial regions are different, which are 318×449 and 512×689, respectively. Our method can directly apply to the raw images and is insensitive to the resolution. For better visualization, we scaled the images here. (a) Input. (b) DCs w/ g. (c) LF. (d) DCs with g'. (e) LF. (f) LF+HF.

We evaluate our method on various face videos. Figure 7 shows our results on several celebrities. Since our method does not require correspondence between the input video and the reference, it is flexible and robust to handle persons wearing glasses (see Figure 6).

Our method can also handle the cases with changing lighting conditions (see Figure 8). Note that the skin colors are extracted from the *first* frame and encoded as DCs, and the light changes in the subsequent frames are all encoded as PRs. Since PVG explicitly separates hues (DCs) and tones (PRs), it can automatically blend PRs (which are the relative or offset values) with DCs, no matter how their boundary colors are changed. This again demonstrates the advantages of PVG vectorization and justifies the effectiveness of our strategy for obtaining DC colors from only the first frame. Figure 9 shows the color swap result in a video. See also the supplementary material for video demonstrations, available online, due to the space limitation.

Now we examine the quality of the results. We observe that both our method and the image-based OMT obtain results of similar quality (which is hard to distinguish by naked eyes), but our method runs significantly faster. The extended Fu's method has a similar

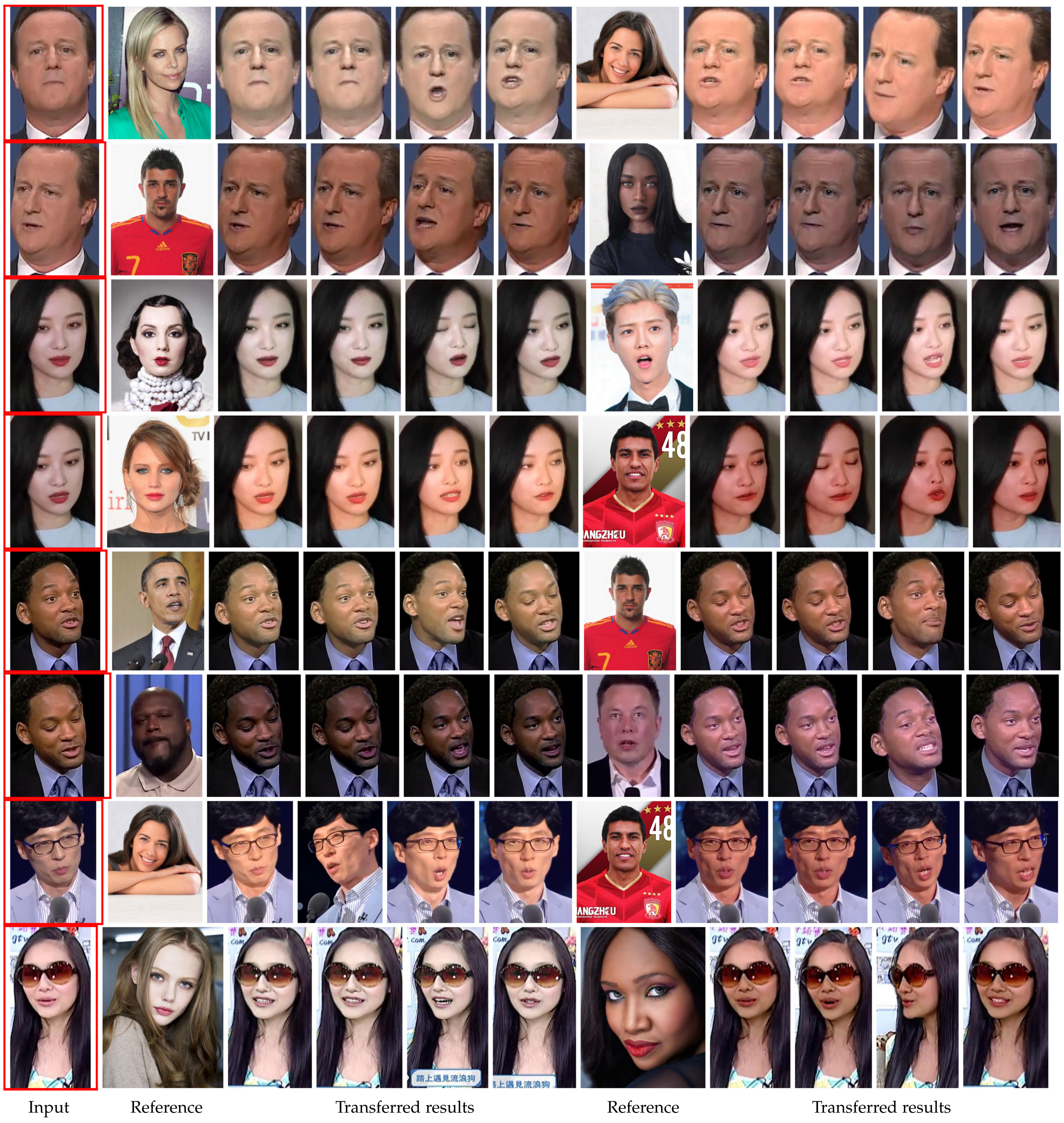

FIGURE 7. Transferred results of celebrities. Each person is transferred from four references. We observe that the transferred colors are highly consistent and show the first, the second, the middle, and the last frames as the representative frames. Note that the first frame is computed via OMT and the subsequent frames are obtained via DC matching. The last two rows show the transferred results of persons wearing glasses. See Parts 3 and 4 in the accompanying video.

run-time performance as ours for a single transfer, but their method produces results with flickering artifacts, since it computes the DC boundary color g separately and there is not guarantee of smoothness between adjacent frames. Our method does not compute g for frames $i \geq 2$, instead it simply transfers the g from the previous frame. As a result, our method is not only stable but also efficient.

To quantitatively evaluate the quality, we compute the PSNR metric P for each frame of the original video and the transferred video. Then, we measure their difference as shown in Figure 10. We observe that the difference curve of our method is almost a constant, meaning that our result consistently mimics the input, differing only by a constant which is independent of the frame index. In contrast, we observe the extended

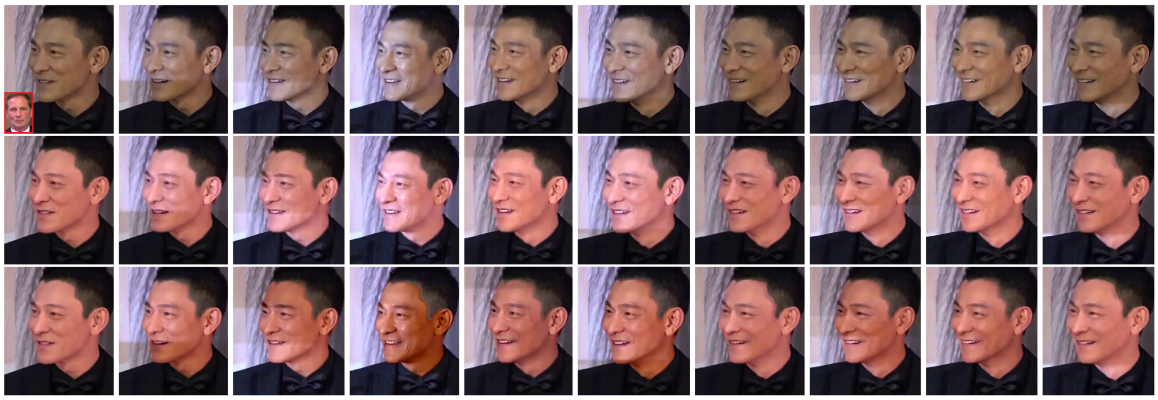

FIGURE 8. Varying lighting conditions. Row 1 shows 10 consecutive frames of a video with large color changes due to flash. Rows 2 and 3 show our results and the results by Fu *et al.*,[2] respectively. The reference image is shown in an inset with red frame. Since their method vectorizes each frame separately, the light changes are taken as low-frequency signals, which are encoded by DCs. Therefore, their method does not compute a stable skin color and their transferred results have undesired skin color changes (e.g., column 4, row 3). In our approach, skin colors are obtained in the first frame only, and all the color changes in the subsequent frames caused by the varying lighting conditions are encoded as high-frequency PRs. With DC matching strategy, the skin colors remain stable in all frames, and they can naturally be blended with the PRs (which encode lighting changes). See Part 5 of the accompanying video.

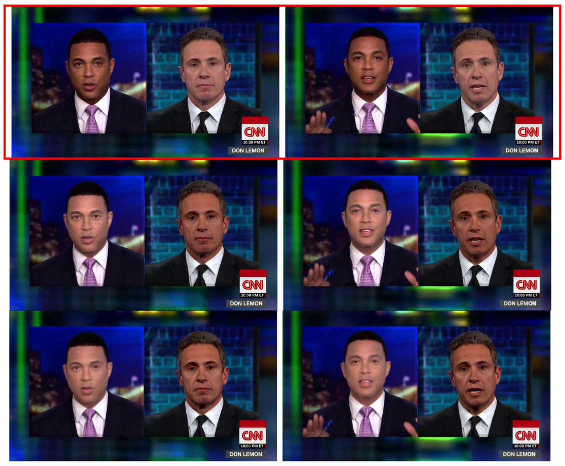

FIGURE 9. Face color swap. Row 1 shows two frames of the original video. Row 2 shows the results by swapping the boundary colors of DCs and keeping the PRs unchanged. Row 3 shows the results by updating PRs with coefficients 0.8 (Lemon) and 1.2 (Cuomo), respectively. We observe that the results with updated PR coefficients have more natural contrast and highlight. See also the accompanying video.

Fu's method has large changes frame by frame, implying that their results are not stable.

Our method requires the input video has a minimal resolution for extracting the salient facial features. We observe that DCs can be accurately extracted when the facial regions have size $> 200 \times 200$. Our DC matching algorithm efficiently transfers the boundary color of the first frame to the subsequent frames. This strategy works well for short video clips. Since vectorization errors often accumulate in long video clips, an easy way to reduce the error is to "reset" the DC boundary colors by solving the least square (3) after a certain number of frames. In our implementation, we set the default clip length as 120 to balance efficiency and accuracy. Resetting DCs may cause a slight color jump in the video. However, for a typical video sequence with frame rate 25fps, the jump lasts only 0.04 s, which is too short to be observed by naked eyes. Figure 11 illustrates the vectorization errors of a 600-frame video clip with different resolutions. We can also see that our method is insensitive to video resolution, as long as each frame is large enough to extract salient edges and facial landmarks.

CONCLUSIONS

We developed a simple yet effective method for automatically transferring face colors in videos. Taking advantage of PVG guided vectorization, our method computes only an OMT map for the first frame and then transfers the DC boundary colors for the subsequent frames by matching the DCs. Thanks to its information reuse nature, our method is highly desired in

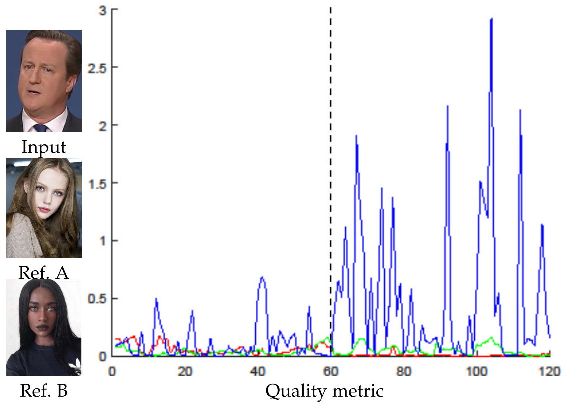

FIGURE 10. Quality analysis. We transfer Reference A and Reference B to the first and second half of the input video and compute the difference of PSNR between the original video and the transferred video. The more constant the difference, the more consistence with the input video, hence the less dependence of the frame index and the less flickering artifacts. The x-axis is the frame index and the y-axis is the squared difference of PSNR between the original video and the transferred video. Both our curve (red) and the curve of the image-based OMT (green) are almost constant, indicating the comparable quality of the results. However, our method runs significantly faster. We observe that simply applying Fu's method[2] to the video frames produces results with flickering artifacts. The large variation of the blue curve confirms Fu's results are unstable. The PSNR deviations of our method, the image-based OMT, and Fu's method are 0.0656, 0.0838, and 0.6248, respectively. See Part 2 of the accompanying video.

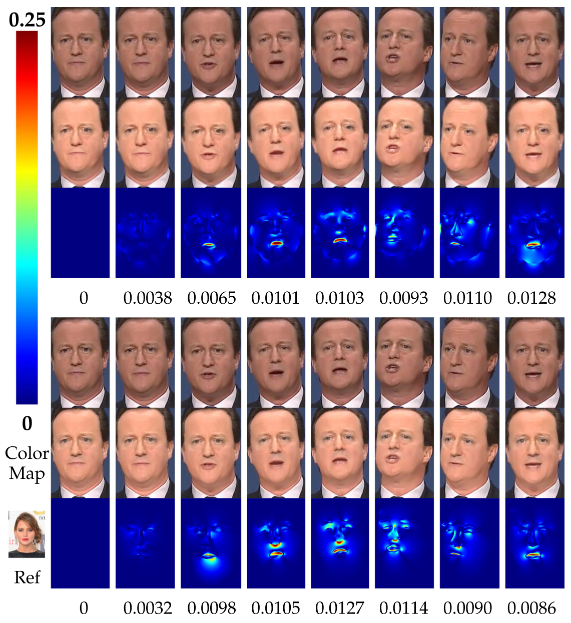

FIGURE 11. Accuracy. We evaluate the vectorization accuracy on a 600-frame video with two different resolutions and visualize the vectorization errors using heat color map, where warm colors denote large error and cold colors small error. The frame ids are 1, 5, 30, 60, 120, 240, 480, and 600, respectively. The top (resp. bottom) 3 rows show the low-resolution (resp. high-resolution) video, in which the size of the facial regions is 200×274 (resp. 800×1096). The value below each figure is the mean error. We observe that the high-resolution video has slightly lower errors than the low-resolution input. In this figure, we do not reset DCs so that accumulation errors can be visualized. In other figures, we reset DCs every 120 frames to stop error propagation.

frequent color transfer from multiple references. Since our method does not require correspondence between the input video and the reference, it is flexible and robust to handle faces with significantly different geometries and postures. Moreover, it also allows the user to edit the highlight and contrast by changing the coefficient of PRs. Experimental results demonstrate the effectiveness of our method.

Limitations. High-quality color transfer requires accurate skin mask and facial feature extraction. For the skin mask, we adopted Hsu *et al.*'s algorithm[8] for skin extraction. We observed that Hsu's method works well if skin colors are clearly different from the background. However, this method often computes wrong masks if the assumption does not hold. Figure 12 shows a failed example where skin and hair have similar colors. In the future, we will improve our method by adopting state-of-the-art deep learning techniques (e.g., He *et al.*[5]) for skin mask extraction. Our facial feature extraction neural network works for a wide range of head poses and camera angles. However, when there are large topological changes of features in a sequence of video frames (e.g., the number of features drop significantly from front face to side face), DC matching may fail due to many unmatched features. A robust *partial* DC matching algorithm is demanded to solve this issue.

ACKNOWLEDGMENTS

We would like to thank the anonymous reviewers for their constructive comments. This project was supported in part by Alibaba-NTU Joint Research Institute, MOE Grant RG20/20, NSFC Grants (61672481, 61702363, 61725204, 61872347), in part by Special Plan

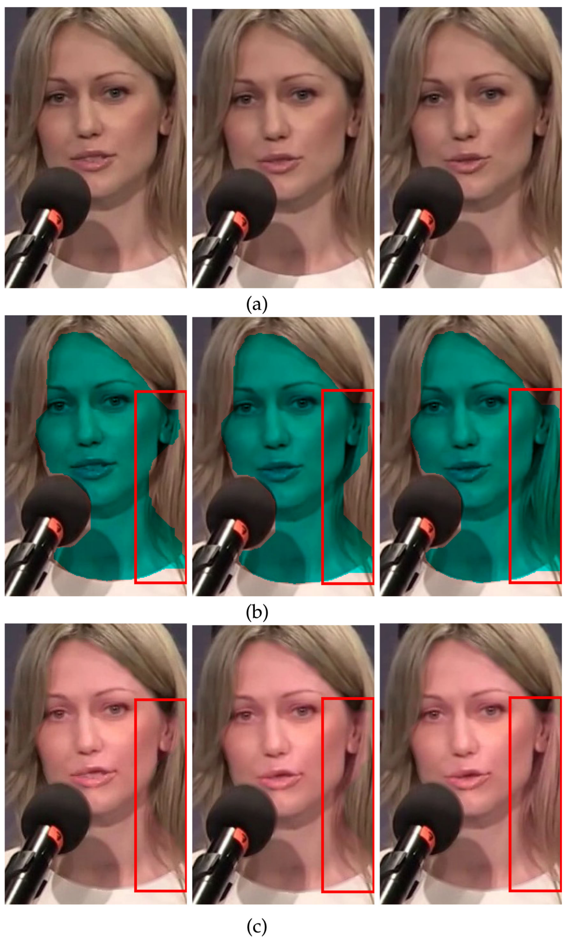

FIGURE 12. Failed case due to wrong skin extraction. The skin color clustering algorithm computes wrong masks (green) which include part of her hair. As a result, skin colors are also transferred to the hair. The artifacts are highlighted in the red boxes. See also the accompanying video. (a) Input. (b) Skin masks (green). (c) Transferred results.

for the Development of Distinguished Young Scientists of ISCAS (Y8RC535018), and in part by Youth Innovation Promotion Association CAS (No. 2018495). This article has supplementary downloadable material at http://ieeexplore.ieee.org, provided by the authors.

REFERENCES

1. P. Feng and J. Warren, "Discrete bi-laplacians and biharmonic b-splines," *ACM Trans. Graph.*, vol. 31, no. 4, pp. 115:1–115:11, Jul. 2012.
2. Q. Fu, Y. He, F. Hou, J. Zhang, A. Zeng, and Y.-J. Liu, "Vectorization based color transfer for portrait images," *Comput.-Aided Des.*, vol. 115, pp. 111–121, 2019.
3. I. Goodfellow *et al.*, "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.
4. M. He, J. Liao, D. Chen, L. Yuan, and P. V. Sander, "Progressive color transfer with dense semantic correspondences," *ACM Trans. Graph.*, vol. 38, no. 2, 2019, Art. no. 13.
5. Y. He *et al.*, "Semi-supervised skin detection by network with mutual guidance," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 2111–2120.
6. G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," 2015, *arXiv:1503.02531*.
7. F. Hou *et al.*, "Poisson vector graphics (PVG)," *IEEE Trans. Visual. Comput. Graph.*, vol. 26, no. 2, pp. 1361–1371, Feb. 2020.
8. R.-L. Hsu, M. Abdel-Mottaleb, and A. K. Jain, "Face detection in color images," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 5, pp. 696–706, May 2002.
9. X. Huang, M.-Y. Liu, S. Belongie, and J. Kautz, "Multimodal unsupervised image-to-image translation," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 172–189.
10. Y. Hwang, J.-Y. Lee, I. S. Kweon, and S. J. Kim, "Color transfer using probabilistic moving least squares," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 3342–3349.
11. T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 4401–4410.
12. M. Leordeanu, R. Sukthankar, and C. Sminchisescu, "Efficient closed-form solution to generalized boundary detection," in *Proc. Eur. Conf. Comput. Vis.*, 2012, pp. 516–529.
13. A. Orzan, A. Bousseau, H. Winnemöller, P. Barla, J. Thollot, and D. Salesin, "Diffusion curves: A vector representation for smooth-shaded images," *ACM Trans. Graph.*, vol. 27, no. 3, pp. 92:1–92:8, 2008.
14. J. Rabin, G. Peyré, J. Delon, and M. Bernot, "Wasserstein barycenter and its application to texture mixing," in *Proc. Int. Conf. Scale Space Variational Methods Comput. Vis.*, 2011, pp. 435–446.
15. E. Reinhard, M. Adhikhmin, B. Gooch, and P. Shirley, "Color transfer between images," *IEEE Comput. Graph. Appl.*, vol. 21, no. 5, pp. 34–41, Jul./Aug. 2001.
16. M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 4510–4520.
17. Y. Shih, S. Paris, C. Barnes, W. T. Freeman, and F. Durand, "Style transfer for headshot portraits," *ACM Trans. Graph.*, vol. 33, no. 4, pp. 148–171, 2014.

18. A. Shrivastava, A. Gupta, and R. Girshick, “Training region-based object detectors with online hard example mining,” in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2016, pp. 761–769.
19. Z. Shu, S. Hadap, E. Shechtman, K. Sunkavalli, S. Paris, and D. Samaras, “Portrait lighting transfer using a mass transport approach,” *ACM Trans. Graph.*, vol. 36, no. 4, pp. 1–15, 2017.
20. G. Xie, X. Sun, X. Tong, and D. Nowrouzezahrai, “Hierarchical diffusion curves for accurate automatic image vectorization,” *ACM Trans. Graph.*, vol. 33, no. 6, pp. 1–11, 2014.

QIAN FU is currently working toward the Ph.D. degree with the School of Computer Science and Engineering, Nanyang Technological University, Singapore. Her research interests fall into the areas of computer graphics, computer-aided design, and computational geometry. She received the bachelor’s and master’s degrees from the College of Information Science and Technology, Beijing Normal University, Beijing, China. She is the corresponding author of this article. Contact her at qfu004@e.ntu.edu.sg.

YING HE is currently an Associate Professor with the School of Computer Science and Engineering, Nanyang Technological University, Singapore. His research interests fall into the general areas of visual computing and he is particularly interested in the problems which require geometric analysis and computation. He received the B.S. and M.S. degrees in electrical engineering from Tsinghua University, Beijing, China, and the Ph.D. degree in computer science from Stony Brook University, Stony Brook, NY, USA. Contact him at yhe@ntu.edu.sg.

FEI HOU is currently a Research Associate Professor with the Institute of Software, Chinese Academy of Sciences, Beijing, China. His research interests include geometry processing, image-based modeling, data vectorization, medical image processing, etc. He received the Ph.D. degree in computer science from Beihang University, Beijing, China, in 2012. He was a Postdoctoral Researcher with Beihang University from 2012 to 2014 and a Research Fellow with the School of Computer Science and Engineering, Nanyang Technological University, from 2014 to 2017. Contact him at houfei@ios.ac.cn.

QIAN SUN is currently an Associate Professor with the College of Intelligence and Computing, Tianjin University, Tianjin, China. Her current research interests include human–computer interaction and computer graphics. She received the Ph.D. degree in computer science from Nanyang Technological University, Singapore. Contact her at qian.sun@tju.edu.cn.

ANXIANG ZENG is currently a Senior Staff Algorithm Engineer and the Director of Alibaba, Hangzhou, China, who is head of the globalization search and recommendation. He is currently working toward the Ph.D. degree with Nanyang Technological University, Singapore. He focuses on the field of search and recommendation and reinforcement learning, and has authored or coauthored more than ten papers in top conferences. He has been working in the searching and recommendation field for more than 10 years. Contact him at renzhong@taobao.com.

ZHENCHUAN HUANG is currently an Algorithm Engineer with the content algorithm team of Search and Recommend Business Unit with Alibaba Group, Hangzhou, China. His research interests are object keypoint detection, metric learning, and applications of computer vision. He received the B.Sc. and M.Sc. degrees in computer science from Nanjing University, Nanjing, China, in 2014 and 2017, respectively. Contact him at zhenchun.hzc@alibaba-inc.com.

JUYONG ZHANG has been a Faculty Member with the School of Mathematical Sciences, since August 2012. His research interests fall into the areas of computer graphics, computer vision, and machine learning. He received the bachelor’s degree in computer science and engineering from the University of Science and Technology of China, Hefei, China, in 2006, and the Ph.D. degree from the School of Computer Science and Engineering, Nanyang Technological University, Singapore, under the supervision of Prof. Jianfei Cai and Prof. Jianmin Zheng. From 2011 to 2012, he first worked as an Intern Student and then a Postdoctoral Research Fellow in LGG of EPFL, Switzerland. Contact him at juyong@ustc.edu.cn.

YONG-JIN LIU is currently a tenured Full Professor with the Computer Science Department, Tsinghua University. His research interests include computational geometry, computer graphics, computer-aided design, and pattern analysis. He received the Ph.D. degree from the Department of Mechanical and Aerospace Engineering, Hong Kong University of Science and Technology, Hong Kong. He received the Ph.D. degree in 2004 from Hong Kong University of Science and Technology. Contact him at liuyongjin@tsinghua.edu.cn.