Automatic Motion-Guided Video Stylization and Personalization[*]

Chen Cao[1], Shifeng Chen[1], Wei Zhang[1], Xiaoou Tang[1,2]
[1]Shenzhen Key Laboratory for Computer Vision and Pattern Recognition
Shenzhen Institutes of Advanced Technology,Chinese Academy of Sciences, China
[2]Department of Information Engineering, The Chinese University of Hong Kong, China
{chen.cao, shifeng.chen, zhangwei}@siat.ac.cn, xtang@ie.cuhk.edu.hk

ABSTRACT

Video stylization transfers a source video into an artistic version while maintaining temporal coherence between adjacent frames. In this paper, we formulate the unsupervised example-based video stylization with Markov random field model. In our algorithm, we implement an improved optical flow algorithm to maintain temporal coherence while improve the accuracy of estimation along motion boundaries. We also extend our algorithm to the application of video personalization, in which human faces keep clear and distinguishable. A series of techniques are fused in video personalization, including face detection and alignment, motion flow, skin detection, and illumination blending. Given a source video and a style template image, our algorithm produces the stylized and/or personalized video(s) automatically. Experimental results demonstrate that our algorithm performs excellently in both video stylization and personalization.

Categories and Subject Descriptors

I.4.9 [**Image Processing and Computer Vision**]: Applications

General Terms

Algorithms, Experimentation

Keywords

Non-Photorealistic Rendering, Video Stylization, Video Personalization

1. INTRODUCTION

Video stylization has many applications in multimedia entertainment. It is mainly based on image stylization while maintaining temporal coherence between adjacent frames. Compared with the original video, the stylized one would create a sense of beauty and inspire intense interests in human mind. As increasing attentions have been paid to personal artworks (e.g., home and campus DV), people would like to transfer their self-shot videos into a stylized one. Definitely the video would be blurred or distorted after stylization, people may hope to keep the face in the video clear and distinguishable. We develop an automatic algorithm to achieve these goals. Given a source video and a style template image, the system could produce stylized and/or personalized results. To the best of our knowledge, there is no work focusing on such a video personalization problem.

In stylization, we first consider single image rendering with an artificial style, and then extend image rendering to videos by processing the frames one by one while preserving temporal coherence. There are a number of existing methods in image style transfer. Some of them focus on using particular approaches to obtain specialized styles, such as [3] for line-drawing and [2] for watercolor. These approaches lay emphasis on analyzing the properties of the specified style, and design a particular non-photorealistic rendering model to simulate the artists' painting brush step by step. Another important classification of stylization is the example-based method [5], [7], [10]. Image analogies [5] is a representative algorithm among the example-based approaches. The main idea of example-based image stylization is that a source image B would be transferred into a stylized one B' by given a style template A'. The transfer algorithms could "learn" the style information from A' and then implement it on B to create a new image B' which has the same content as B and the similar style as A'.

Similar to [5], our algorithm is formulated in the model of Markov random fields (MRFs). In [5], both the template A' and its ground-truth image A should be given, while our approach needs only an arbitrary template image A' to create a simulated artwork, which reduces the algorithm limitations greatly. To improve the speed and performance, we use a coarse-to-fine belief propagation algorithm [8] to solve the global optimization of MRF, and use image quilting [4] to generate seamless patch boundaries.

For video stylization if we employ the image stylization algorithm to each frame of the video independently, the "flickering" effect (style textures randomly change in position and appearance frame by frame) would appear and make the result blinking and blurred. In order to preserve temporal coherence, we use the optical flow algorithm to find correspondences between frames.

Video stylization may blur or distort human faces in the video. Inspired by a cartoon personalization system called

[*]Area chair: Lei Chen

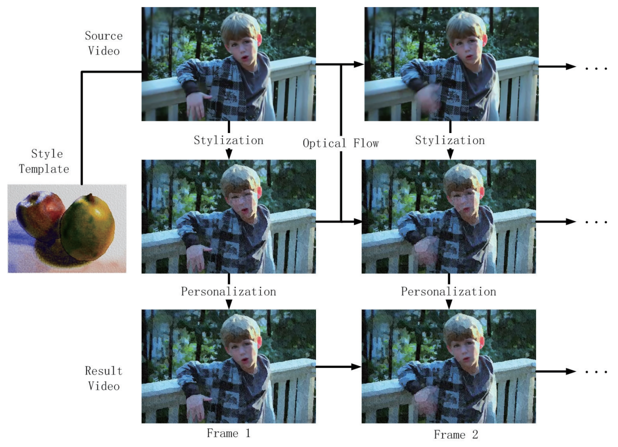

Figure 1: The framework of our algorithm. Top row: the first frame (left) and the second frame (right) of the source video. Middle row: the template image of watercolor style (left), the stylization result of the first frame as initialization (middle), and the stylization result of the second frame via optical flow (right). Faces are blurred in these frames. Bottom row: the final result of personalization.

EasyToon [11], we extract faces from the source video and process them separately from the non-facial parts. By using face detection and alignment, motion flow, skin detection, and illumination blending, we can get a result with recognizable faces in stylized video, i.e., the video personalization. The framework of our algorithm is shown in Fig. 1.

2. VIDEO STYLIZATION

The first step of video stylization is to initialize the first frame by using example-based image style transfer algorithm. Then the initialization is propagated to the following frames by optical flow. Our image stylization algorithm is inspired by image analogies [5]. In our algorithm, the well aligned ground-truth of the style template image A is not needed, which is necessary in [5]. We divided both A' and B into small patches and take patches from A' to replace the patches in B according to some rules. To solve this patch replacement problem, we build an MRF model and find its solution as follows. All steps of stylization is processed in luminance channel (Y channel) of YIQ color space. The final result is obtained by combining the changed Y channel and the unchanged I and Q channel.

2.1 Markov Random Field Model

The style template image A' could be divided into $w \times h$ patches and they are denoted as $L = \{l_1, l_2, \cdots, l_m\}$, here each l_i represents a $w \times h$ patch taken from A' and m is the patch number. The target image B (the first frame) is also divided into the same size patches, regarded as $\mathcal{V} = \{v_1, v_2, \cdots, v_n\}$. For every patch region v_i in $\mathcal{V}$, we paste a patch in L on v_i. The goal for style rendering is to find the best configuration $P = \{p_1, p_2, \cdots, p_n \mid p_i \in L\}$ (where p_i is the patch pasted on v_i) to minimize the energy function

$$E(P) = \sum_{v_i \in \mathcal{V}} D(p_i) + \mu \sum_{(v_i, v_j) \in \mathcal{E}} C(p_i, p_j), \qquad (1)$$

where $D(p_i)$ is the data cost of pasting patch p_i to region v_i, $C(p_i, p_j)$ is the consistency cost of a neighboring couple patches (v_i, v_j) to be pasted by (p_i, p_j), μ is a constant to balance D and C, and $\mathcal{E}$ is the set of edges linking each node in $\mathcal{V}$ to its four neighbors.

The data cost term $D(p_i)$ is defined as

$$D(p_i) = \sum_{(x,y) \in p_i} \left\| \left(p_i(x,y) - \overline{p_i}\right) - \left(v_i(x,y) - \overline{v_i}\right) \right\|^2, \qquad (2)$$

where (x, y) is the pixel coordinate in both patch p_i and the overlapping region v_i, $\overline{p_i}$ and $\overline{v_i}$ represent the mean value of all the pixel colors in p_i and v_i, respectively, and $\|\cdot\|$ is the L2 norm. The consistency cost term $C(p_i, p_j)$ is the sum of the squared differences (SSD) of pixel colors in the overlapping region between p_i and p_j.

We choose belief propagation (BP) to solve the energy minimization problem. The computational complexity of BP is the square of the number of label candidates in L. The large size of L makes BP intractable. Therefore, we propose two schemes to speed up the algorithm: selecting representative patches and utilizing two-step BP.

The number of elements in L could always be more than 10000 in our experiment. We select only 5000 representative patches. We define the quality of a patch l_i as

$$Q(l_i) = \sum_{(x,y) \in l_i} \left| Median\left(A'(x,y)\right) - A'(x,y) \right|, \qquad (3)$$

where $(x, y) \in l_i$ means that (x, y) is a pixel in patch l_i from the stylized sample A', and $Median()$ is the median filter. The larger $Q(l_i)$ is, the more style information the patch l_i contains. We take the top 5000 l_i with largest $Q(l_i)$ values as the patch candidates in the BP algorithm.

We utilize a two-step BP algorithm [8] to accelerate the BP process, i.e., to perform BP twice with N_1 and N_2 label candidates each time instead of operating BP once with N candidates, for that N_1 and N_2 are much smaller than N. Refer [8] for the details of two-step BP.

2.2 Image Quilting

After getting the result of P, we employ image quilting [4] to produce a seamless image from overlapping patches. The purpose of image quilting is to stitch together small patches by finding a ragged edge with the least inconsistencies between adjacent patches. The image quilting step is formulated as a min-cut problem. Suppose that a source node $\mathcal{S}$ and a sink node $\mathcal{T}$ represent the two adjacent patches. Then a graph is constructed whose nodes are the pixels in the overlapping region of the two adjacent patches. Each node (pixel) is connected to its four neighbors while the pixels in the boundaries are connected to $\mathcal{S}$ or $\mathcal{T}$. The weight of the edge connecting two neighbor pixels (s, t) is

$$W(s,t) = \|F_{sc}(s) - F_{sk}(s)\|^2 + \|F_{sc}(t) - F_{sk}(t)\|^2, \qquad (4)$$

where F_{sc} and F_{sk} represent the feature vectors of a pixel in source and sink patches, respectively.

2.3 Propagating to Succeeding Frames

After obtaining the initialization for the first frame, we propagate the stylized result to succeeding frames via motion flow. In this process, we use the optical flow algorithm in [1] to estimate the pixel-level correspondences between successive frames. Then we divided the currently processing frame into patches as in the image stylization step. For the patches that could be found corresponding patches in the

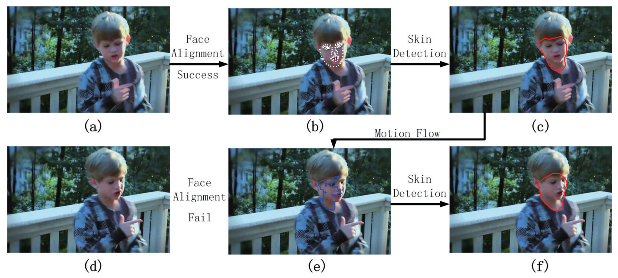

Figure 2: Example of the face extraction process. (a): The 10th frame in the source video. (b): The landmarks (white points) of the face alignment result. (c): The result of skin detection for (a). (d): The 11th frame. Face alignment fails due to the oblique orientation. (e): The face region obtained from the previous frame via motion flow. (f): The skin detection result based on the region in (e).

previous frame, we directly copy the patches in the previous frame into the current frame in the related positions. For the patches that has no relevant regions in its previous frame, we re-render them by using our image stylization approaches. To evaluate the reliability of correspondences between two patches, we calculate the SSD between them and set a threshold to judge the validation of correspondence.

In addition, a conspicuous artifact of optical flow is the over-smoothing or randomly dragging effect at occluded regions, which would visibly impair the quality of output video as mentioned in [12]. We adopt the algorithm of occlusion detection and bilateral diffusion in [12] to deal with this unsatisfactory phenomenon.

3. VIDEO PERSONALIZATION

3.1 Face Extraction

At first, a dynamic cascade method based face detection [13] is utilized to detect faces in frames. Then we employ an active shape model based face alignment [14] to locate the face feature points and a single Gaussian model based skin detection [9] to get the whole face region. The face extraction process is shown in Fig. 2.

Face alignment returns 87 landmarks of face features if succeed (see the white points in Fig. 2(b)). With these feature points, we could locate the face in the frame. However, these points do not surround the entire area of the face region. Thus we use a skin detection algorithm to obtain the whole face region after getting the face alignment results. We only detect skin in the area near the 87 landmarks of the face for high accuracy.

We use a Gaussian model for skin detection, which is:

$$p\left(c \mid skin\right) = \frac{1}{2\pi\left|\Sigma_S\right|^{1/2}} e^{-\frac{1}{2}\left(c-\mu_s\right)^T \Sigma_S^{-1}\left(c-\mu_s\right)}, \qquad (5)$$

where c is a color vector of a pixel in the detected area. We perform the detection in YCrCb color space and c represents a two-dimensional vector indicates the values of Cr and Cb. The distribution parameters are

$$\mu_s = \frac{1}{n}\sum_{i=1}^{n} c_i, \quad \Sigma_S = \frac{1}{n-1}\sum_{i=1}^{n}\left(c_i-\mu_s\right)\left(c_i-\mu_s\right)^T, \qquad (6)$$

where n is the number of skin color training data c_i. The pixels in the surrounded area of the landmarks of the face alignment result are set to be the training data. Of course we should firstly eliminate the dark points which might be the eyes or other shaded regions of the face in the training data set. $p\left(c \mid skin\right)$ represents the probability of a point with color vector c being a skin point. We set a threshold for $p\left(c \mid skin\right)$ to judge whether the point we detect belongs to skin or not. The red-line surrounded region in Fig. 2(c) is the result of skin detection.

An important and unavoidable issue is that face alignment might fail in some frames (see Fig. 2(d)) for the reason that a face is posed in an oblique orientation, or a part of the face is out of the camera screen, or the face is partially shaded by hands or other objects. This phenomenon is general in many cases. Address to this problem, we reuse the optical flow result obtained in the video stylization stage to trace the corresponding area of the face in the previous frame when face alignment fails in the current frame. The blue-line surrounded region in Fig. 2(e) is the corresponding area of face mask in Fig. 2(c) obtained via motion flow. After this step, we can get some regions of the face parts. We then set these parts as training data and conduct skin detection near these regions. Fig. 2(f) is the detection result.

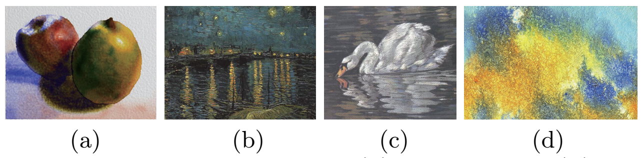

Figure 3: Template images: (a) Watercolor; (b) Oil on canvas; (c) Pastel; (d) Natural texture.

3.2 Face Blending

After the skin detection step, we obtain face masks in all frames. Then we simply combine the completely stylized face F' and not stylized face F with a coefficient α between 0 and 1 to control the amount of personal information on the final stylized face. The final stylized face F_{final} is:

$$F_{final} = \alpha F + \left(1-\alpha\right) F'. \qquad (7)$$

At last, We put the final stylized faces on the extracted face regions in the completely stylized result. Obviously, the seam boundary would occur between the face regions and the non-facial regions. To settle this problem, we use Poisson blending [6] to create merged and seamless boundaries between the faces and the other parts in the video.

4. EXPERIMENTAL RESULTS

To demonstrate the performance of our algorithm, we test it on some child-centered videos with four style templates (watercolor, oil-on-canvas, pastel, and natural texture in Fig. 3).

The first source video is about an active boy with exaggerated poses. Three frames from the video are shown in Fig. 4(a). The corresponding frames with watercolor style are given in Fig. 4(b). Notice that face regions are blurred after stylization. The final result of personalization is given in Fig. 4(c). Fig. 4(d) shows the personalized result on oil-on-canvas style. This style has a strong texture. Therefore we can see clearly that the temporal coherence is kept well: the texture on the same object (especially the background) maintained unchanged in different frames.

Fig. 5 is another result on the video about a girl who is chatting online. Fig. 5(b) includes the pastel-stylized frames

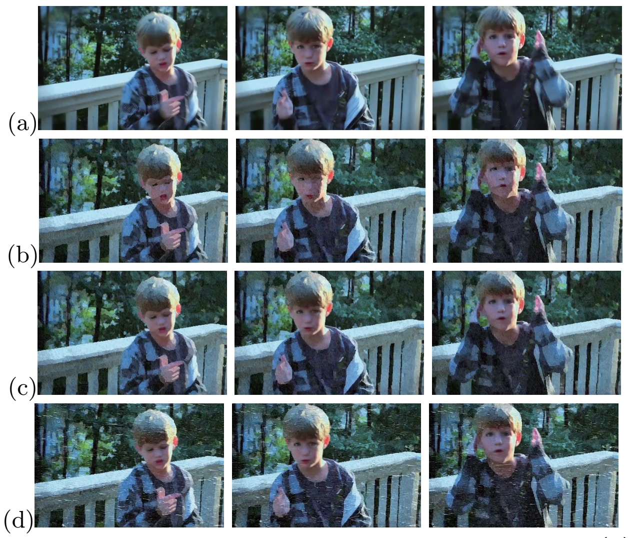
Figure 4: Example results by our algorithm. (a) The source video frames. (b) The stylized output in watercolor, faces are obscure. (c) The result artwork in watercolor after stylization and personalization. (d) The stylized and personalized artwork in oil-on-canvas style.

with obscure faces. The corresponding personalized result are given in Fig. 5(c). Fig. 5(d) is the result with natural texture style. From the result we can see that the faces are recognizable in a pleasing appearance and the non-facial parts show conspicuous texture similar to the style template.

5. CONCLUSION

In this paper, we propose an automatic algorithm to stylize and personalize videos into artistic works. It comprises two stages: video stylization and video personalization. Video stylization stylizes the first frame of the video for a given style template image and then propagates the result to all frames via motion flow. Video personalization extracts face regions in the video frames and processes them separately from the non-facial regions. Our stylization algorithm is example based and formulated with an MRF model. A two-step BP is used to solve the MRF energy minimization problem efficiently. We utilize an improved optical flow estimation method with occlusion detection and bilateral filtering to preserve temporal coherence and overcome the unsatisfactory artifacts in video stream. In video personalization stage, a method by combining face detection and alignment with skin detection is used to extract face region. Then the algorithm processes the face exclusively to make face clear and distinguishable. Our algorithm is robust even in the case that face alignment fails in some frames since we proposes a motion flow based face detection method to handle the problem. To the best of our knowledge, there is no existing work about such video personalization.

6. ACKNOWLEDGEMENTS

This work was supported by Introduced Innovative R&D Team of Guangdong Province (Robot and Intelligent Information Technology); Natural Science Foundation of China (60903117); and Science, Industry, Trade, and Information Technology Commission of Shenzhen Municipality, China (JC201005270357A, ZYC201006130314A).

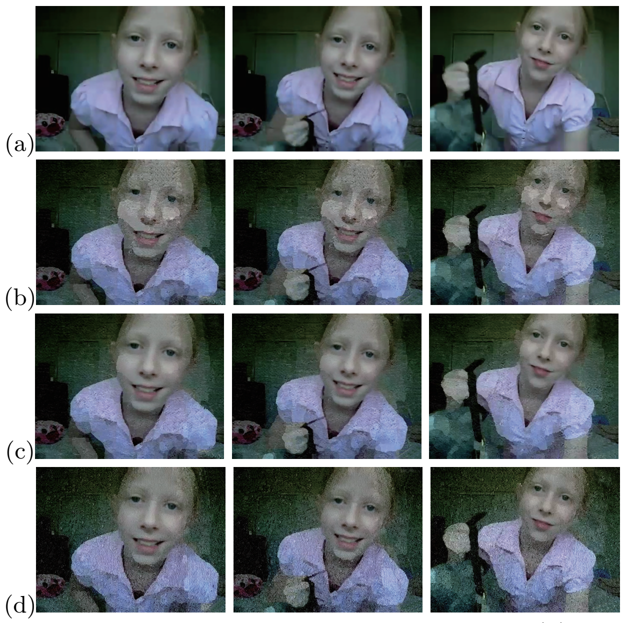
Figure 5: More results of our algorithm. (a) The source video frames. (b) The stylized but blurred faces results in pastel style. (c) The stylized and personalized artwork in pastel style. (d) The stylized and personalized artwork in natural texture style.

7. REFERENCES

[1] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. In *ECCV*, 2004.

[2] C. Curtis, S. Anderson, J. Seims, K. Fleischer, and D. Salesin. Computer-generated watercolor. In *SIGGRAPH*, 1997.

[3] D. DeCarlo and A. Santella. Stylization and abstraction of photographs. *ACM Trans. on Graphics*, 21(3):769–776, 2002.

[4] A. Efros, W. Freeman, et al. Image quilting for texture synthesis and transfer. In *SIGGRAPH*, 2001.

[5] A. Hertzmann, C. Jacobs, N. Oliver, B. Curless, and D. Salesin. Image analogies. In *SIGGRAPH*, 2001.

[6] P. Pérez, M. Gangnet, and A. Blake. Poisson image editing. *ACM Trans. on Graphics*, 22(3):313–318, 2003.

[7] R. Resales, K. Achan, and B. Frey. Unsupervised image translation. In *ICCV*, 2003.

[8] H. Ting, S. Chen, J. Liu, and X. Tang. Image inpainting by global structure and texture propagation. In *ACM MM*, 2007.

[9] V. Vezhnevets, V. Sazonov, and A. Andreeva. A survey on pixel-based skin color detection techniques. In *Graphicon*, 2003.

[10] B. Wang, W. Wang, H. Yang, and J. Sun. Efficient example-based painting and synthesis of 2d directional texture. *IEEE Trans. on Visualization and Computer Graphics*, 10(3):266–277, 2004.

[11] F. Wen, S. Chen, and X. Tang. Easytoon: cartoon personalization using face photos. In *ACM MM*, 2008.

[12] J. Xiao, H. Cheng, H. Sawhney, C. Rao, and M. Isnardi. Bilateral filtering-based optical flow estimation with occlusion detection. In *ECCV*, 2006.

[13] R. Xiao, H. Zhu, H. Sun, and X. Tang. Dynamic cascades for face detection. In *ICCV*, 2007.

[14] Y. Zhou, L. Gu, and H. Zhang. Bayesian tangent shape model: estimating shape and pose parameters via Bayesian inference. In *CVPR*, 2003.