Multimed Tools Appl (2015) 74:245–258
DOI 10.1007/s11042-013-1759-3

Directional texture transfer for video

Dongwann Kang · Phutphalla Kong · KyungHyun Yoon · SangHyun Seo

Published online: 20 November 2013

Abstract Texture transfer is a method that copies the texture of a reference image to a target image. This technique has an advantage in that various styles can be expressed according to the reference image, in a single framework. However, in this technique, it is not easy to control the effect of each style. In addition, when this technique is extended to processing video images, maintaining temporal coherence is very difficult. In this paper, we propose an algorithm that transfers the texture of a reference image to a target video while retaining the directionality of the target video. The algorithm maintains the temporal coherency of the transferred texture, and controls the style of the texture transfer.

Keywords Texture transfer · Temporal coherence · Example-based rendering · Video processing

1 Introduction

Non-photorealistic rendering (NPR) is a computer graphics technique that mimics human artistic expression. NPR has been studied since the 1990s [7], and video-based NPR was studied mainly in the 2000s. Most research studies focused on the expression of a specific style: painterly [6, 9, 15], pen and ink [13, 20], and watercolor [2]. These are stroke-based

D. Kang · P. Kong · K. Yoon
ChungAng University, Seoul, Republic of Korea

D. Kang
e-mail: dongwann@cglab.cau.ac.kr

P. Kong
e-mail: kong@cglab.cau.ac.kr

K. Yoon
e-mail: khyoon@cau.ac.kr

S. Seo (✉)
ETRI, Daejeon, Republic of Korea
e-mail: shseo@etri.re.kr

rendering (SBR) methods, which use brush strokes as the basic primitive [10]. Various styles can easily be expressed by modeling each brush stroke according to the style. However, each style requires a distinct painting method, so that representing various styles in a single framework is not easy. To overcome this limitation, a method for transferring texture is suggested.

Texture transfer is a method that copies the texture of a reference image to a target image [4]. This technique has an advantage in that various styles, such as painterly, pen and ink, watercolor, etc., can be expressed according to the reference image, in a single framework. However, this method expresses the textureness by compositing the pixels of the reference image. It is therefore not easy to control the effect of each style. In addition, when this technique is extended to processing video images, maintaining temporal coherence is very difficult. Because of these problems, texture transfer for a single target image is studied widely, but for video images only rarely.

In this paper, we propose an algorithm that transfers the texture of a reference image to a target video while retaining the directionality of the target video. The algorithm maintains the temporal coherency of the transferred texture, and controls the style of the texture transfer. In order to move the transferred texture along the motion of the target video, we synthesize the texture by using the previous frame considering pixels moved along the motion. In addition, we control the textureness and the directionality by using the saliency of each video frame. When the saliency value is high, the detail of the target frame is retained to a greater extent. However, strong textureness is expressed when the saliency value is low. To control the textureness effectively, we employ pyramid dilation based on the saliency.

The main contribution of this paper is to provide a directional texture transfer algorithm for a video target while maintaining the temporal coherency of the transferred texture. Additionally, we propose a technique that estimates some weights using the saliency to control the effect of the texture transfer.

2 Related works

NPR is divided into painterly rendering, pen and ink, watercolor, etc., according to the style. SBR is a popular NPR technique for expressing style, which expresses artistic style by using the brush stroke as the basic primitive. It can easily control stylization by defining the attributes of the brush stroke and painting method. In the 2000s, SBR was extended to animation, and research studies were therefore conducted whose objective was to maintain the temporal coherency of stroke-based animation [8, 12, 19, 23, 24]. The methods used in these studies achieved temporally coherent animations by moving, adding, and modifying the stroke according to the motion between frames. However, the attributes of the stroke and painting method vary according to the style, and therefore different styles cannot be expressed in a single framework. Therefore, techniques that allow different styles to be expressed in single a framework were studied. Texture transfer is one of them.

In the original research on texture transfer, texture synthesis techniques were studied. The purpose of texture synthesis is to create a random very high-quality texture from a tiny example texture. Methods of texture synthesis have been studied [3, 5, 17, 18]; however, these methods do not allow an artistic effect to be expressed, but rather the texture is imitated using self-similarity. We concentrated primarily on an artistic texture transfer algorithm.

The objective of image analogies [11] is to broaden the range of non-photorealistic rendering techniques and produce results by employing coherent local and optimal global texture synthesis. However, this approach is still rather time consuming, and it requires

users to give additional unfiltered information about the target image/frame in order to achieve a precise correlation with the source texture. Then, a fast texture transfer algorithm was proposed. This technique straightforwardly extends the coherent local texture synthesis technique [25]. This approach can improve the results of image analogies, in terms of imitating artistic painting effects. It is much faster than the image analogies technique of [11]. The most important factor is that it needs only one example image. In the study reported in [22], a patch of the reference image was used as the basic primitive. The authors extracted some patches from the reference image, and generated textures by synthesizing them. In [16], a texture transfer algorithm that has a directional effect was proposed. The algorithm transfers the texture of the reference image into the target image, but retains the direction of the target image. Unlike previous SBR methods, these methods have the advantage that various styles can be expressed according to the style of the reference image. However, it is not easy to control the effect of the texture transfer.

In [25], texture transfer for single images was extended for video images. The texture is synthesized by selecting the best pixels from among the candidates in the reference image. The authors additionally considered the motion between target video frames; their method therefore allows the texture to be transferred along the motion of the target video. However, it cannot express the directional effect of the target video. Like previous texture transfer methods, this method has the limitation that it cannot control various effects of the texture transfer. In this paper, we extend this method to express directional effects and control various effects of the texture transfer.

3 Directional texture transfer for video

3.1 Overview of texture transfer for video

In this section, we represent an overview of a pixel-wise texture transfer algorithm for target video images. The algorithm is divided into two main steps. In the first step, texture transfer is performed by using a reference image and the first frame of the target video. In the second step, a temporary result of the next frame is generated by using the results of the motion and texture transfer of the previous frame, and a final result frame is generated by using our texture transfer algorithm. Table 1 shows the overview of each step.

The quality of the result is dominated by the selection of the best pixel among the candidates. Therefore, the design of the distance function is most important. We design our distance function by considering the texture pattern similarity, intensity similarity, directionality, and temporal coherency. We present the details of the distance function in Section 3.2. In addition, we control the effect of texture transfer by using an adaptive kernel and directional weight control according to the gradient value. We explain this in detail in Section 3.2.

3.2 Distance function

In this paper, the distance function for choosing the best pixel among the candidates is as follows.

$$D(r,q) = D_N(r,q) + D_L(r,q) + D_F(r,q) + w_I D_I(r,q) \quad (1)$$

The equation consists of four terms. The first term is related to the similarity in intensity. This enables the function to select a candidate pixel that is similar to the original pixel in the target image. The second term is related to the texture similarity. This enables the function

Table 1 The pseudo code of directional texture transfer for video 1

Input : Target video frames ($T_1 \sim T_n$), Reference image (S)
Output : Result video frames ($R_1 \sim R_n$)

1st step :
Generating the 1st frame of the result video, R_1

1) Initialize R_1 by filling it with randomly selected pixels on S
Define the mapping function g to indicate the random position of S

2) In scan-line order, for each pixel r in R_1,

Build candidate set,
$Q \leftarrow \{g(t) + (r - t) | t \in L(r)\}$, $L(x)$ is L-shaped neighborhoods
Update $R(r) \leftarrow S(\arg\min_{q \in Q} D(r, q))$

2nd step :
Generating the rest of the frames of the result video, $R_2 \sim R_n$

1) In sequential order, for each frame of R_k,

Initialize R_k by filling it with pixels inherited from R_{k-1} (following the optical flow)

If there is no pixel inherited from R_{k-1}, fill R_k with randomly selected pixels on S, and mark that pixel of R_k

2) In scan-line order, for each marked pixel r in R_k,

Build candidate set,
$Q \leftarrow g(t) + (r - t) | t \in L(r)$, $L(x)$ is L-shaped neighborhoods

Update $R(r) \leftarrow S(\arg\min_{q \in Q} D(r, q))$

to select a candidate pixel whose texture pattern is similar to the current texture pattern. The third term aims to maintain temporal coherency. This allows the function to select a candidate pixel that is similar to the corresponding pixel of the previous frames result. The last term is related to directionality. This makes the transferred texture follow the direction of the target image. In the next subsections, we describe each term in detail.

3.2.1 Intensity similarity

Similarity of the pixel intensity is one of the criteria for choosing the pixel that will be placed on a given location in the target image from among the candidates that consist of pixels selected from the reference image. The pixels from the reference image must accurately express the target image. Therefore, a pixel must be selected whose intensity is similar to that of the pixel of the target image. The equation for measuring the intensity similarity is as follows.

$$D_N(r, q) = ||N_r - Nq||^2 \tag{2}$$

$$N_x = avg(N(x)) \tag{3}$$

Here, $N(x)$ denotes the intensities of the pixels in the predefined kernel. The average intensity of the pixels in a specific kernel is employed for measuring the intensity similarity because it is intended to be less influenced by the noise of the image. In previous research

efforts [1, 16, 25], a rectangular kernel was used. However, averaging the intensities of the pixels in a rectangular kernel produces a blurring effect around the edges in the target image. When the kernel size is increased, this blurring effect becomes more pronounced.

In order to solve this problem, we employ an anisotropic Kuwahara kernel [14] instead of a rectangular kernel. Anisotropic Kuwahara filtering is used for edge-preserving smoothing, which prevents this blurring effect around the edges. We obtain the flow directions of the target image using the edge tangent flow [13]. Then, using this flow, we calculate the Kuwahara kernel [14]. Figure 1 presents the flow of a target image and its corresponding anisotropic Kuwahara kernel.

3.2.2 Texture similarity

In texture transfer methods, the pixel whose texture is similar to the texture of the pixel in the target image is selected from among the candidates. This process is represented by the following equations:

$$D_L(r,q) = (1/L(r))||H_r - Hq||^2 \tag{4}$$

$$H_r = [R(x_i) - \overline{R(x_i)}], x_i \in L(r) \tag{5}$$

$$H_q = [S(x_i) - \overline{S(x_i)}], x_i \in L(q) \tag{6}$$

Here, $||\cdot||$ denotes L_2 norm, $L(x)$ denotes the L-shaped neighbors, $R(\cdot)$ and $S(\cdot)$ denote the intensities of pixels of result and reference image, and $|\cdot|$ denotes the size of the set. Since our algorithm is performed in scan-line order, the L-shaped neighbors indicate the neighbor pixels that were synthesized before the current pixel. By comparing the deviation of the intensities of the L-shaped neighbors, we calculate the texture similarity. When two textures are similar to each other, this term has a low value so that the pixel of that texture can be selected for synthesizing the texture.

3.2.3 Temporal coherency

If the texture transfer, which is based on single target image, is applied to each frame of the target video, flickering occurs in the resulting frames. This is why the motions between frames are not considered. In order to solve this problem, we choose the pixel whose intensity is similar to that of the pixel inherited from the previous result frame. We estimate the motions between frames using an optical flow algorithm. However, if we select the best pixel strictly based on motion, the shower door effect is produces near the occlusion/disocclusion

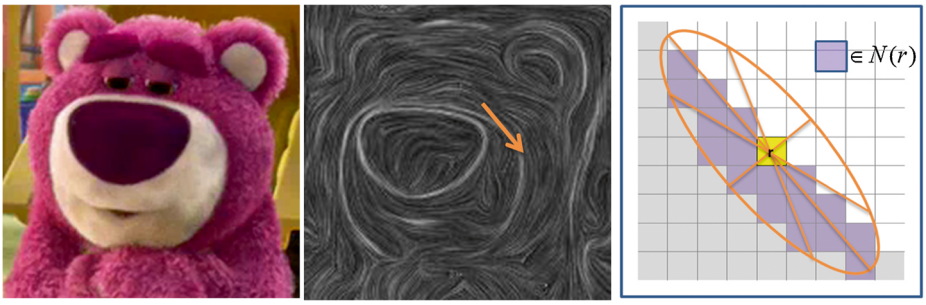

Fig. 1 Flow direction of image and anisotropic kernel for preventing blurring

boundaries [25]. To solve this, we reflect the motion according to the degree of confidence in the optical flow quality [21].

$$D_F(r,q) = C(r) \cdot ||S(q) - S(g(r + F(r)))||^2 \tag{7}$$

Here, F represents the backward optical flow, which is computed using the previous and current frames. $C(r)$ is the confidence at r, which is defined as follows:

$$C(r) = G\left(\nabla \cdot F(r); \sigma_f^2\right) \cdot G\left(T(r) - T_{-1}(r + F(r)); \sigma_I^2\right), \tag{8}$$

where $\nabla \cdot F$ represents the divergence of the flow field, T is the current frame of the target video, T_{-1} is the previous frame of the target video, and $G\left(\cdot; \sigma^2\right)$ represents a zero-mean Gaussian function with variance σ^2. Please refer to [21] for further details.

3.2.4 Directionality

The directionality of texture exaggerates the shape object in an image and creates a more artistic effect. In order to exaggerate the directionality, [16] considered the flow of textures. They defined I-shape as neighborhoods laid down in the flow direction belonging to the L-shape. By using I-shaped kernel, they find candidate that matches the intensity of pixels on the flow of texture. Similar to this, we choose a pixel whose intensity is similar to that of the pixel in the flow direction using an I-shaped kernel so that the texture has a directional pattern along the gradient direction; this is represented by

$$D_I(r,q) = ||\overline{R(r_i)} - S(q)||^2, r_i \in I(r) \tag{9}$$

Here, $I(x)$ denotes the pixels in an I-shaped kernel.

3.3 Adjustment of texture transfer style

3.3.1 Degree of texture

In artistic painting, brushes of various sizes are used. In the background region, strong textures generated by large brush strokes can be observed. On the contrary, for details of a significant object or complicated region, small brushstrokes are used. Similarly, we can control the degree of texture according to the saliency of the target image, for our algorithm to create a more artistic effect. We transfer a weak texture in the salient region and a strong texture in a relatively less salient region.

We estimate the saliency by using the gradient magnitude of the target image. If we calculate the gradient using a Sobel or Raplcian operator, the magnitude of the gradient varies within a narrow range near the edges. In order to expand the range such that the

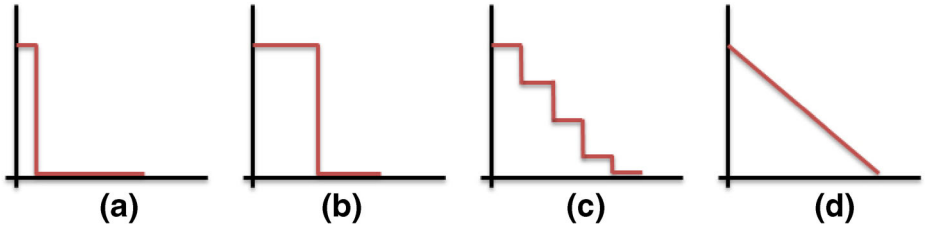

Fig. 2 Change in gradient magnitude: (**a**) original gradient magnitude, (**b**) dilation, (**c**) pyramid dilation, (**d**) pyramid dilation + smoothing

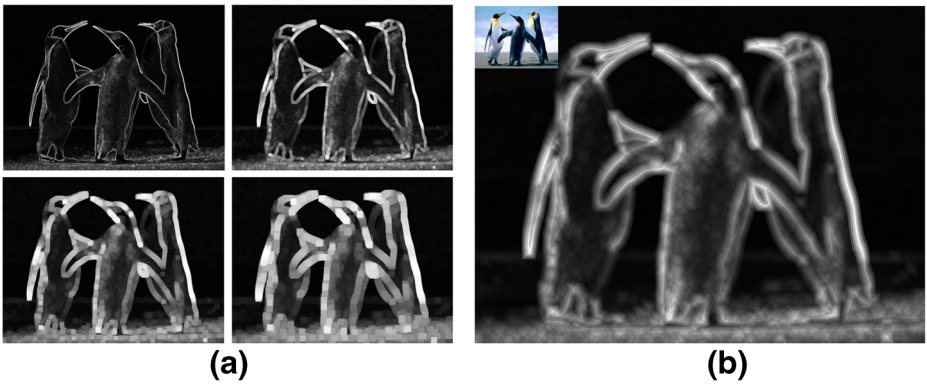

Fig. 3 Pyramid dilation by accumulating several dilation results obtained with different sized kernels: (**a**) dilation with kernel sizes of 1, 4, 8, and 12 and (**b**) pyramid dilation result (saliency map)

gradient values vary to a greater degree, we employ a dilation operation. Although this dilation widens the range, the magnitude of the gradient does not fall off along the direction toward the end of the range. In order to solve this, we employ pyramid dilation, which accumulates the dilation results using various sized kernels. In order to avoid the staircase effect, we smooth the pyramid dilation result using a Gaussian filter (Fig. 2). Figure 3 shows the results of pyramid dilation proposed in this paper. We use this to obtain the saliency map of the target image.

In our texture transfer algorithm, the degree of texture depends on the size of the kernel. When a larger kernel is used, a stronger texture is observed, and vice versa. Figure 4 shows a comparison of different degrees of texture according to kernel size.

We adjust the kernel size of the texture transfer algorithm using the saliency map so that we can control the degree of texture. The following equation shows the calculation of the kernel size according to the saliency.

$$size = K_{min} + G(r) \cdot (K_{max} - K_{min}) \tag{10}$$

Here, $G(x)$ denotes the value of the saliency map, and K_{min} and K_{max} denote the minimum and maximum sizes of the kernel, respectively.

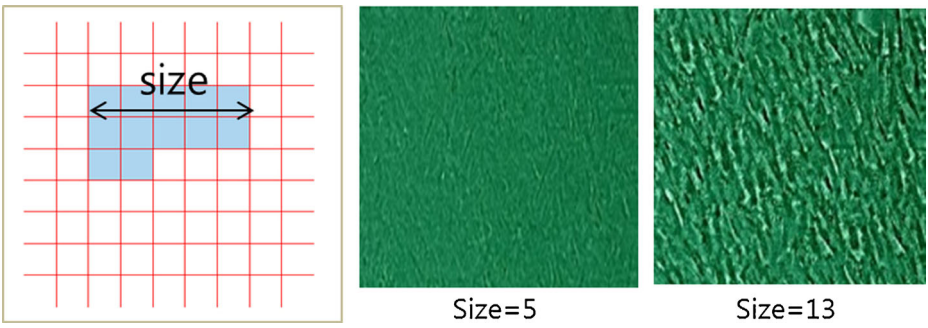

Fig. 4 Comparison of degrees of texture according to kernel size

3.3.2 *Weight of directionality*

In order to control the degree of directionality, a weight of directionality term is used in (1). If the weight is increased, the texture pattern is aligned along the flow direction, but it tends to be different from the pattern in the reference image. On the contrary, if the weight is decreased, the texture pattern reflects the reference image well, but it does not follow the flow of the target image. In [25], the weight is determined by the user.

In this paper, we propose a method of assigning the weight automatically according to the saliency obtained above. In the salient region, we can express the directionality by using a high weight. On the contrary, we can preserve the texture pattern of the reference image by using a low weight, in a less-salient region.

$$w_I = w_{max} \cdot G(r) \tag{11}$$

Here, w_{max} is the maximum weight defined by the user; in this paper, we use a value of 0.6.

4 Experimental results

We conducted experiments using different example source images to examine the corresponding different results. We compared our results with those of previous studies, and found that our method can consider and express the object shape well, whereas previous studies did not. In addition, we can control the various stroke attributes using our pixel-based texture transfer algorithm. Implementation takes about 9 s, depending on the image size and parameters, for a VGA image size with a neighborhood size of 513 pixels. We used a core 2 duo 2.33 GHz CPU PC with 4 GB memory to obtain the results in this paper.

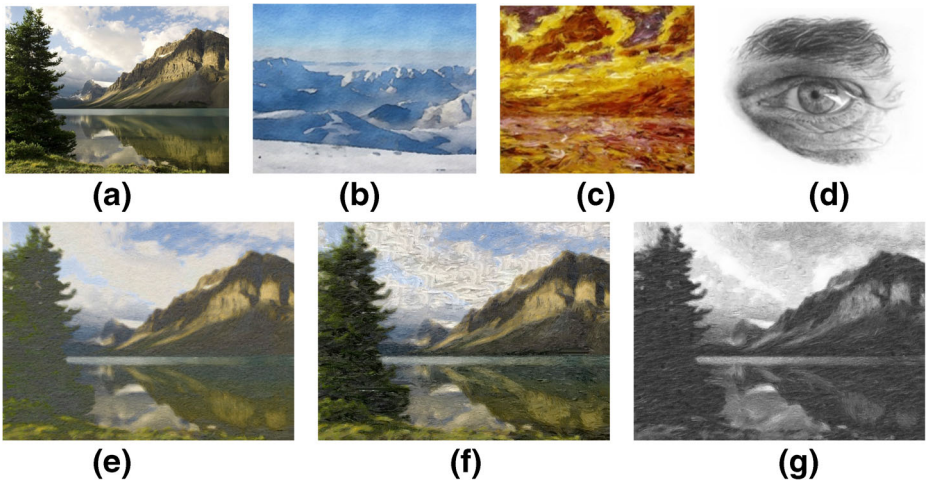

Fig. 5 Results according to the reference image. (**a**) target image, (**b**–**d**) reference images, (**e**–**g**) result images

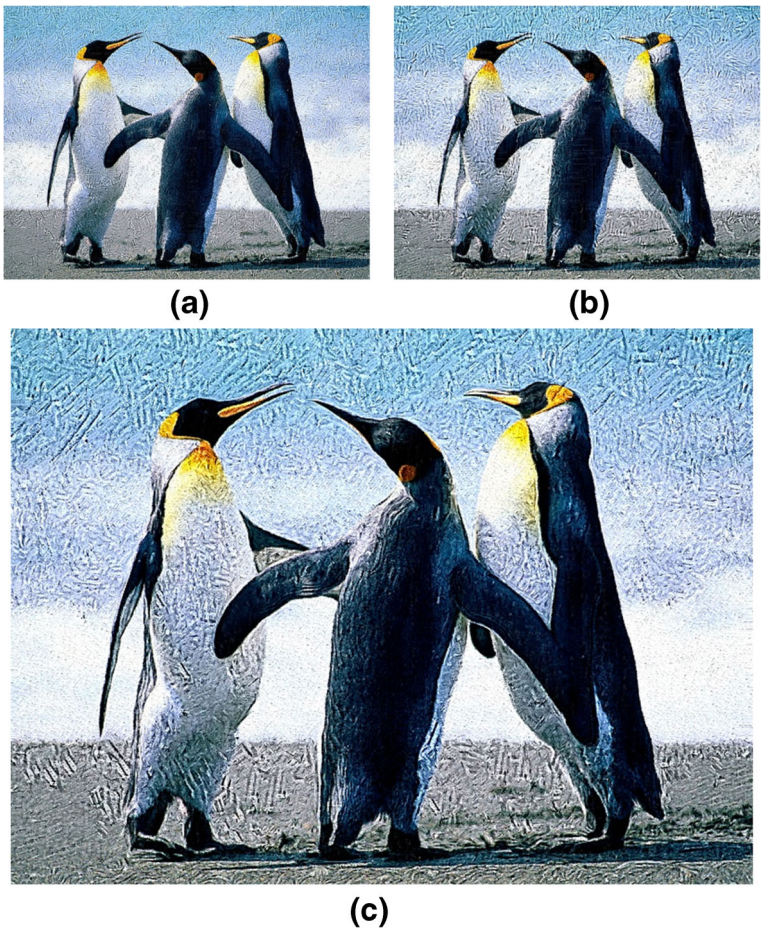

Fig. 6 The results with fixed sized kernel (**a** and **b**) and adaptive kernel (**c**)

Figure 5 shows the different results according to the different reference image. We use painterly, pen and ink, and water color style images as the reference images. By transferring

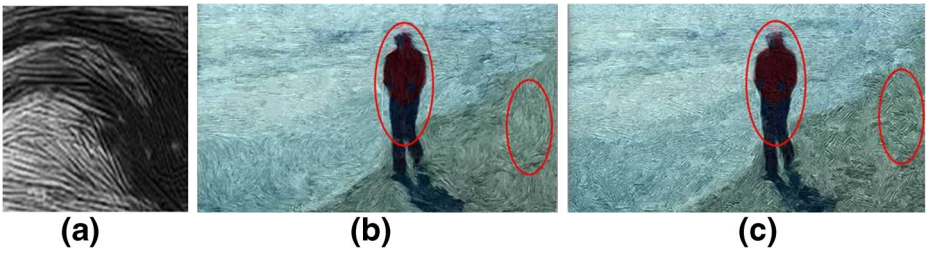

Fig. 7 The effect of directional weight control: (**a**) Reference image; (**b**) the result without directional weight control; (**c**) the result with directional weight control

Fig. 8 Comparison with previous work

the texture of the reference image, we converted the style of the target image to that of the reference image.

Figure 6 shows a comparison of the results when a fixed-size or adaptive kernel was applied. Figure 8a, b shows the results when a fixed-size kernel of size 5 (Fig. 6a) and 13 (Fig. 6a) was applied. When the larger kernel was used, the textureness is exaggerated, but the detail of the target image is lost. In contrast, when the smaller kernel was used, the detail

Fig. 9 The result videos by composing with various reference images

of the target image is maintained well, but the textureness is degraded. In Fig. 6c, which is the result when an adaptive kernel was applied, the textureness is exaggerated in the region where the gradient value is low, and the detail is maintained in the region where the gradient value is high. Therefore, the detail and textureness are both expressed well.

Figure 7 shows the effect of directional weight control. In Fig. 7b, which is the result when a constant directional weight was applied, the directionality is expressed well. However, the patterns of the texture are confused in some regions (red circles in the figure). In Fig. 7c, which is the result when directional weight control was used, directionality is expressed well in high gradient regions, and textureness is maintained well in low gradient regions.

Figure 8 shows a comparison of the results of a previous study [25] and our study. We generated our results using the same reference image and target video as in [25]. As seen in the figure, in contrast to the results of [25], the directionality is well expressed in our results. Moreover, as in [25], the temporal coherency is well maintained.

Figure 9 shows the videos that result when various reference images are used for the composition. Please see the experimental results in the accompanying videos. Our algorithm can generate various results according to the reference image, in a single framework.

5 Conclusions and future work

In this paper, we proposed a directional texture transfer method for video targets. First, we improved the distance function by employing a directional and a temporal coherency term. By using this function, we generated a stylized video having a directional effect based on the image gradient of the target video frame. In addition, the temporal coherency of the transferred texture is well maintained. Next, we controlled the effect of texture transfer. By using adaptive kernel, we controlled the textureness. In addition, we controlled the directionality by using directional weight control. In our study, in a single framework, we were able to generate videos in various styles according to the reference image.

Our algorithm has some limitations. In this study, we employed optical flow [21] for estimating the motion between video frames. However, the motion estimation is not accurate, and therefore estimation error can accumulate. This will generate an incorrect texture pattern. Therefore, it is necessary to enhance the motion estimation to obtain better results. In addition, our algorithm is pixel-based, so that it is easy to convert into parallel processing. Therefore, we are planning future studies on parallel processing using GPU.

Acknowledgments This work (Grants No. C0004960) was supported by Business for Cooperative R & D between Industry, Academy, and Research Institute funded Korea Small and Medium Business Administration in 2013.

References

1. Ashikhmin M (2003) Fast texture transfer. IEEE Comput Graph Appl 23(4):38–43. doi:10.1109/MCG.2003.1210863
2. Bousseau A, Neyret F, Thollot J, Salesin D (2007) Video watercolorization using bidirectional texture advection. ACM Trans Graph 26(3). doi:10.1145/1276377.1276507
3. De Bonet JS (1997) Multiresolution sampling procedure for analysis and synthesis of texture images. In: Proceedings of the 24th annual conference on computer graphics and interactive techniques, SIGGRAPH '97. ACM Press/Addison-Wesley Publishing Co., New York, pp 361–368. doi:10.1145/258734.258882

4. Efros AA, Freeman WT (2001) Image quilting for texture synthesis and transfer. In: Proceedings of the 28th annual conference on computer graphics and interactive techniques, SIGGRAPH '01. ACM, New York, pp 341–346. doi:10.1145/383259.383296
5. Eisenacher C, Lefebvre S, Stamminger M (2008) Texture synthesis from photographs. In: Proceedings of the Eurographics conference. http://www-sop.inria.fr/reves/Basilic/2008/ELS08
6. Gooch B, Coombe G, Shirley P (2002) Artistic vision: painterly rendering using computer vision techniques. In: NPAR, pp 83–90
7. Haeberli P (1990) Paint by numbers: abstract image representations. In: Proceedings of the 17th annual conference on computer graphics and interactive techniques, SIGGRAPH '90. ACM, New York, pp 207–214. doi:10.1145/97879.97902
8. Hays J, Essa I (2004) Image and video based painterly animation. In: Proceedings of the 3rd international symposium on non-photorealistic animation and rendering, NPAR '04. ACM, New York, pp 113–120. doi:10.1145/987657.987676
9. Hertzmann A (1998) Painterly rendering with curved brush strokes of multiple sizes. In: Proceedings of the 25th annual conference on computer graphics and interactive techniques, SIGGRAPH '98. ACM, New York, pp 453–460. doi:10.1145/280814.280951
10. Hertzmann A (2003) A survey of stroke based rendering. IEEE Comput Graph Appl 23:70–81
11. Hertzmann A, Jacobs CE, Oliver N, Curless B, Salesin DH (2001) Image analogies. In: Proceedings of the 28th annual conference on computer graphics and interactive techniques, SIGGRAPH '01. ACM, New York, pp 327–340. doi:10.1145/383259.383295
12. Hertzmann A, Perlin K (2000) Painterly rendering for video and interaction. In: Proceedings of the 1st international symposium on non-photorealistic animation and rendering, NPAR '00. ACM, New York, pp 7–12. doi:10.1145/340916.340917
13. Kang H, Lee S, Chui CK (2007) Coherent line drawing. In: Proceedings of the 5th international symposium on non-photorealistic animation and rendering, NPAR '07. ACM, New York, pp 43–50. doi:10.1145/1274871.1274878
14. Kyprianidis JE, Kang H, Döllner J (2009) Image and video abstraction by anisotropic Kuwahara filtering. Comput Graph Forum 28(7):1955–1963. Special issue on Pacific Graphics 2009
15. Lee H, Lee C, Yoon K (2009) Motion based painterly rendering. Comput Graph Forum 28(4):1207–1215. http://doi.wiley.com/10.1111/j.1467-8659.2009.01498.x
16. Lee H, Seo S, Ryoo S, Yoon K (2010) Directional texture transfer. In: Proceedings of the 8th international symposium on non-photorealistic animation and rendering, NPAR '10. ACM, New York, pp 43–48. doi:10.1145/1809939.1809945
17. Lefebvre S, Hoppe H (2006) Appearance-space texture synthesis. ACM Trans Graph 25(3):541–548. doi:10.1145/1141911.1141921
18. Liang L, Liu C, Xu YQ, Guo B, Shum HY (2001) Real-time texture synthesis by patch-based sampling. ACM Trans Graph 20(3):127–150. doi:10.1145/501786.501787
19. Litwinowicz P (1997) Processing images and video for an impressionist effect. In: Proceedings of the 24th annual conference on computer graphics and interactive techniques, SIGGRAPH '97. ACM Press/Addison-Wesley Publishing Co., New York, pp 407–414. doi:10.1145/258734.258893
20. Salisbury MP, Wong MT, Hughes JF, Salesin DH (1997) Orientable textures for image-based pen-and-ink illustration. In: Proceedings of the 24th annual conference on computer graphics and interactive techniques, SIGGRAPH '97. ACM Press/Addison-Wesley Publishing Co., New York, pp 401–406. doi:10.1145/258734.258890
21. Sand P, Teller S (2008) Particle video: long-range motion estimation using point trajectories. Int J Comput Vision 80(1):72–91. doi:10.1007/s11263-008-0136-6
22. Wang B, Wang W, Yang H, Sun J (2004) Efficient example-based painting and synthesis of 2d directional texture. IEEE Trans Vis Comput Graph 10(3):266–277. doi:10.1109/TVCG.2004.1272726
23. Wang J, Bhat P, Colburn RA, Agrawala M, Cohen MF (2005) Interactive video cutout. ACM Trans Graph 24(3):585–594. doi:10.1145/1073204.1073233
24. Wang J, Xu Y, Shum HY, Cohen MF (2004) Video tooning. ACM Trans Graph 23(3):574–583. doi:10.1145/1015706.1015763
25. Ye N, Sim T, Miao X (2010) Video stylization by single image example. In: Proceedings of ICIP'10, pp 3993–3996

Dongwann Kang received his Ph.D. degree in Chung-Ang University, South Korea in 2013. He received his B.S. and M.S. degrees in Computer Science and Engineering from Chung-Ang University in 2006 and 2008. Now, he is working for same university as a research professor. His research interests include NPR, GPU and image manipulation.

Phutphalla Kong received his M.S degree in Department of Computer Science and Engineering from Chung-Ang University, Seoul, Rep. of Korea. Now, he is working for Institute of Technology of Camboida as Lecturer, His research interests include Non-Photorealistic Rendering and Animation and Image manipulation.

KyungHyun Yoon received his B.S. and M.S. degree from Chung-Ang University. Also He received his PhD from University of Connecticut in 1991. From 1991, He is currently a tenured Associate Professor in the Department of Computer Science & Engineering at the Chung-Ang University in Korea. His research interests are in the area of non-photorealistic rendering, color theory, image processing.

SangHyun Seo received his B.S. degrees in Computer Science and Engineering from Chung-Ang University, Seoul, Rep. of Korea, in 1998 and M.S. and Ph.D. degrees in GSAIM Dep. at Chung-Ang University in 2000 and 2010 respectively. He was senior researcher at G-Inno System from 2002 to 2005. He was the post-doctoral researcher at Chung-Ang University, in 2010 and the post-doctoral researcher at LIRIS Lab, Lyon 1 University from Feb. 2011 to Feb 2013. Now, he is working at the ETRI (Electronics and Telecommunications Research Institute). He research interests are in the area of computer graphics and non-photorealistic rendering, 3D GIS system and game technology.