Generating Striped Animations by Inverse Line Convergence Index Filter

Toru Hiraoka
Department of Information Systems, University of Nagasaki
1-1-1, Manabino Nagayo-chou, Nishisonogi-gun, Nagasaki-ken, 815—2195, Japan
E-mail: hiraoka@sun.ac.jp

Ryosuke Takaki
Division of Computer Science, Graduate School of Regional Design and Creation, University of Nagasaki
1-1-1, Manabino, Nagayo-chou, Nishisonogi-gun, Nagasaki-ken, 815-2195, Japan
E-mail: mc220002@sun.ac.jp

Abstract

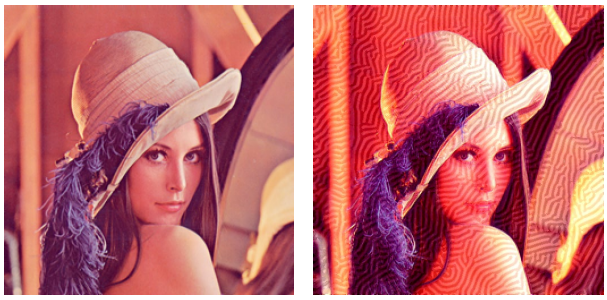
A non-photorealistic rendering method has been proposed for generating a striped image which is overlaid striped patterns in a photograph. The conventional method generates the striped image by an iterative process using an inverse line convergence index filter. When a striped animation is generated by converting each frame of a video by the conventional method, flickering occurs in the generated striped animation. In this paper, we propose a method for generating a striped animation that has characteristic with less flicker from the video. The effectiveness of the proposed method is investigated experimentally. As a result of the experiments, the proposed striped animation had less flicker than the conventional striped animation.

Keywords: Non-photorealistic rendering, animation, striped pattern, inverse line convergence index filter

1. Introduction

Various non-photorealistic rendering methods for a photograph have been proposed[1,2,3,4,5,6]. These methods are used in a wide range of applications, for examples, applications embedded in a personal computer and a portable terminal. Applying these methods to a video is to improve the visual appearance. A Study on non-photorealistic rendering of the video has also been conducted[7], but the conventional method was been known that flickering occurs in the non-photorealistic rendering animation.

In this paper, we focus on the non-photorealistic rendering method for generating a striped image[6] from the photograph. Also we apply the conventional method to the video. The striped image is overlaid the striped patterns in the photograph as shown in Fig. 1. The conventional method generates the striped image by an iterative process using an inverse line convergence index filter, and is characterized by the ability to automatically generate the striped patterns in accordance with the shading and the edge of the photograph. When a striped animation is generated by converting each frames of the video by the conventional method, flickering occurs in the generated striped animation. Therefore, we suppress flicker by using the forward and backward frames of the



(a) Lena image (b) Striped image
Fig. 1. Lena image and striped image

video. The effectiveness of the proposed method is investigated experimentally. By visually and quantitatively comparing the proposed striped animation to the conventional striped animation, we show that the proposed method can suppress flicker better than the conventional method.

2. Method

Let input pixel values of RGB on coordinates (i,j) in o-th frame of the video be $f_{R,i,j,o}$, $f_{G,i,j,o}$ and $f_{B,i,j,o}$ $(i = 1,2,\dots,I; j = 1,2,\dots,J; o = 1,2,\dots,O)$, and let output pixel values after processing with a line convergence index filter[8] be $LF(f_{R,i,j,o})$, $LF(f_{G,i,j,o})$ and $LF(f_{B,i,j,o})$.

The line convergence index filter is executed in the following manner. Consider the straight line $l_{m,o}(m = 1,2,\dots,M)$ inclined $\theta_{m,o}(= 0, \pi/M, 2\pi/M, \dots, (M-1)\pi/M)$ degrees from the x-direction of the target pixel (i,j). Let the length of the straight line $l_{m,o}$ in each direction around the target pixel (i,j) be W_1 pixels. Consider the inclined rectangle which has the center at (i,j) and the length of sides are W_1, W_2. Let N be the number of pixels (k,l) inside the rectangle (Conceptual diagram for W_1, W_2 and N are shown in Fig. 2). Compute a cosine of the angle between a vector perpendicular to a straight line $l_{m,o}$ from a neighboring pixel (k,l) and a vector $((s_{k+2,l+2,o} + s_{k+2,l+1,o} + s_{k+2,l,o} + s_{k+2,l-1,o} + s_{k+2,l-2,o}) - (s_{k-2,l+2,o} + s_{k-2,l+1,o} + s_{k-2,l,o} + s_{k-2,l-1,o} + s_{k-2,l-2,o}), (s_{k+2,l+2,o} + s_{k+1,l+2,o} + s_{k,l+2,o} + s_{k-1,l+2,o} + s_{k-2,l+2,o}) - (s_{k+2,l-2,o} + s_{k+1,l-2,o} + s_{k,l-2,o} + s_{k-1,l-2,o} + s_{k-2,l-2,o}))$ calculated from the density variation. Let the cosine of the angle be $c_{m,n,o}(n = 1,2,\dots,N)$. The term $s_{i,j,o}$ is calculated as the following equation.

$$d_{i,j,o} = \frac{f_{R,i,j,o} + f_{G,i,j,o} + f_{B,i,j,o}}{3}. \quad (1)$$

$$s_{i,j,o} = \frac{\sum_{p=-P}^{P} \frac{1}{1+|p|} d_{i,j,o+p}}{\sum_{p=-P}^{P} \frac{1}{1+|p|}}. \quad (2)$$

Where P is a positive constant. Next, compute $C_{m,o}$ that is the average of the absolute value of $c_{m,n,o}$ of N neighboring pixels (k,l) in each straight line $l_{m,o}$ as the following equation.

$$C_{m,o} = \frac{1}{N} \sum_{n=1}^{N} |c_{m,n,o}|. \quad (3)$$

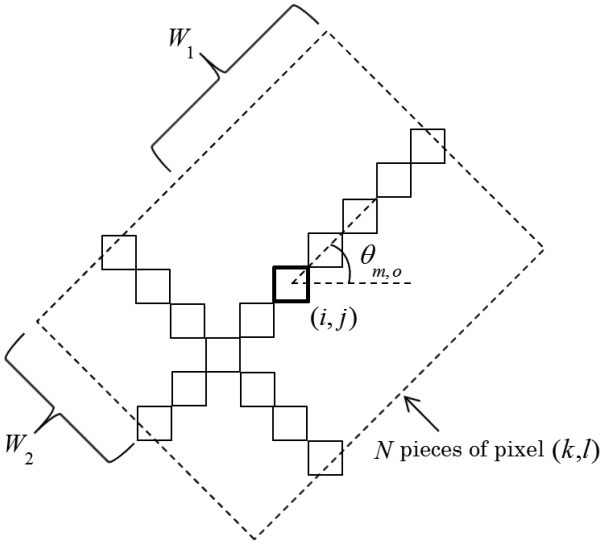


Fig. 2. Conceptual diagram for W_1, W_2 and N

With the maximum value of $C_{m,o}$ in all straight lines $l_{m,o}$ in each pixel (i,j), denote as $g_{i,j,o}$. With the minimum and maximum values of $g_{i,j}$ in all pixels, denoted as $g_{min,o}$ and $g_{max,o}$, respectively. Transform $g_{i,j,o}$ to $h_{i,j,o}$ and thus to the pixel values had from 0 to 255 by thefollowing equation.

$$h_{i,j,o} = 255 \frac{g_{i,j,o} - g_{min,o}}{g_{max,o} - g_{min,o}}. \quad (4)$$

$LF(f_{R,i,j,o})$, $LF(f_{R,i,j,o})$ and $LF(f_{R,i,j,o})$ are the same value as $h_{i,j,o}$.

Compute the pixel value $f_{R,i,j,o}^{(t)}$, $f_{G,i,j,o}^{(t)}$ and $f_{B,i,j,o}^{(t)}$ by using the inverse line convergence index filter as

$$f_{R,i,j,o}^{(t)} = a(f_{R,i,j,o}^{(t-1)} - LF(f_{R,i,j,o}^{(t-1)})) + f_{R,i,j,o}. \quad (5)$$

$$f_{G,i,j,o}^{(t)} = a(f_{G,i,j,o}^{(t-1)} - LF(f_{G,i,j,o}^{(t-1)})) + f_{G,i,j,o}. \quad (6)$$

$$f_{B,i,j,o}^{(t)} = a(f_{B,i,j,o}^{(t-1)} - LF(f_{B,i,j,o}^{(t-1)})) + f_{B,i,j,o}. \quad (7)$$

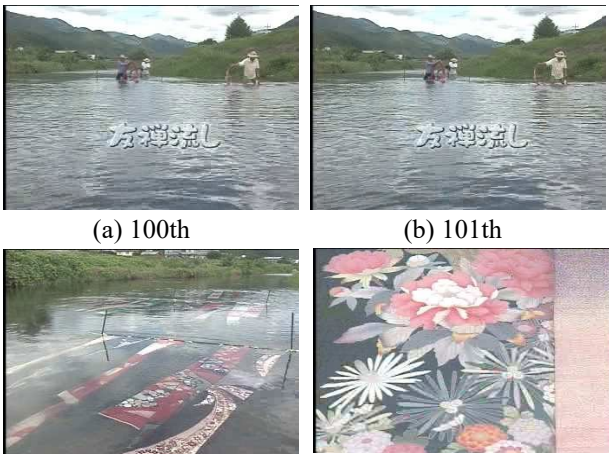
Where a is a positive constant and t is the number of iterations. Let the initial value $f_{R,i,j,o}^{(0)}$, $f_{G,i,j,o}^{(0)}$ and $f_{B,i,j,o}^{(0)}$ be $f_{R,i,j,o}$, $f_{G,i,j,o}$ and $f_{B,i,j,o}$, respectively. $f_{R,i,j,o}^{(t)}$, $f_{G,i,j,o}^{(t)}$ and $f_{B,i,j,o}^{(t)}$ set to 0 if their values are less than 0, and set to 255 if their values are greater than 255.

Finally, the N striped images are obtained after processing of the inverse line convergence index filter of T generated from these striped images.

3. Experiments

We applied the proposed method to the Yuzenzome video[9] which consists of 703 frames, 30 frames / second, $352 * 240$ pixels and 256 tone. The 100, 101, 387 and 388th frames of the Yuzenzome video are shown in Fig. 3. The 387 and 388 th ftames are the scene change frames. In the following experiments, referring to the literature 6), we set the values of the parameters M, W_1, W_2, a and T to 8, 4, 4, 0.4 and 30, respectively.

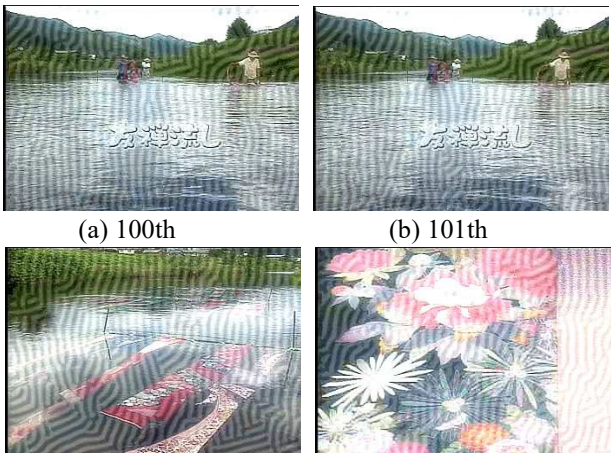
First, we visually compared the proposed striped animation to the conventional striped animation. The 100, 101, 387 and 388th frames of the conventional striped animation and the proposed striped animation are shown in Fig. 4 and Fig. 5, respectively. We set the value of the parameter P to 3. Observing these animations, the proposed striped animation were suppressed flicker more than the conventional striped animation. Observing Fig. 4 and Fig. 5, the proposed striped animation had less changes in the striped patterns than the conventional striped animation.



(a) 100th (b) 101th

(c) 387th (d) 388th

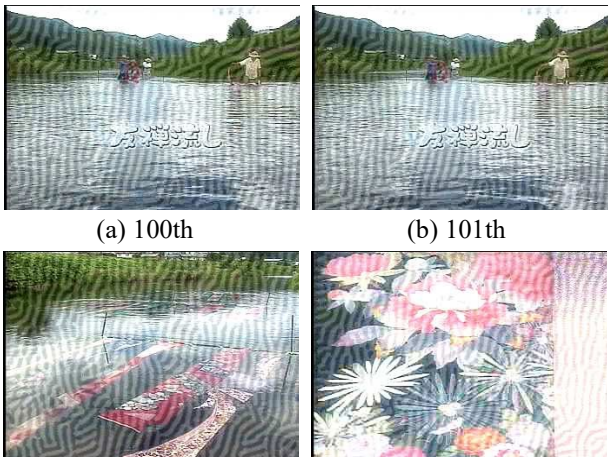
Fig. 3. The frames of the Yuzenzome video



(a) 100th (b) 101th

(c) 387th (d) 388th

Fig. 4. The frames of the conventional striped animation



(a) 100th (b) 101th

(c) 387th (d) 388th

Fig. 5. The frames of the proposed animation

Next, we quantitatively compared the proposed striped animation to the conventional striped animation. We calculated the average of the absolute difference value of pixel values between front and rear frames of these animations. As the average become smaller, flickering is less. The average of the conventional striped animation is shown in Table 1. On the other hand, the average of the proposed striped animation is shown in Table 2, when we changed the values of parameter P from 1 to 10. Observing Table 2, as the values of parameter P became bigger, the average became small. Observing Table 1 and Table 2, the average of the proposed striped animation were smaller than the average of the conventional striped animation. Thus, the proposed method can suppress flicker better than the conventional method.

Lastly, we compared the calculation time of the proposed method to that of the conventional method. The calculation time of the conventional method is shown in Table 3. On the other hand, the calculation time of the proposed method is shown in Table 4, when we changed the values of parameter P from 1 to 10. The calculation time in Table 3 and Table 4 is one iteration calculation

Table 1. The average of the conventional striped animation

Average
23.856

Table 2. The average of the proposed striped animation when the value of the parameter P is changed

P	Average
1	20.403
2	20.264
3	20.190
4	20.151
5	20.118
6	20.100
7	20.087
8	20.075
9	20.067
10	20.062

Table 3. The calculation time of the conventional method [second]

Average
4.377

Table 4. The calculation time of the proposed method when the value of the parameter P is changed [second]

P	Average
1	4.381
2	4.390
3	4.394
4	4.399
5	4.401
6	4.405
7	4.410
8	4.413
9	4.420
10	4.426

[second]. Observing Table 4, the calculation time became big as the values of parameter P became bigger, but there are no big differences in these calculation times. Observing Table 3 and Table 4, there was no big difference in the calculation time between the proposed method and the conventional method. Thus, the proposed method is considered to be an effective method compared with the conventional method on the calculation time. However, considering real-time use on portable terminals, it is necessary to speed up the proposed method. Thecomputing environment of the experiment is Windows 7 Professional for OS, 3.40 GHz for CPU, 8.00 GB for RAM and C language.

4. Conclusion

We proposed a method for generating a striped animation that has characteristic with less flicker from the video. The effectiveness of the proposed method was investigated experimentally. As a result of the experiments, the proposed striped animation had less flicker than the conventional striped animation. The future tasks are to speed up the proposed method and to apply the proposed method to other videos.

Acknowledgements

This work was supported by JSPS KAKENHI Grant Number JP19K12664.

References

1. D. Nakagawa, K. Yamaguchi, K. Muraoka, and N. Chiba, Generating Pen and Ink Illustrations from Gray-scale Images : Stippling and Line Drawing Style, *Journal of IIEEJ*. **30**(4), 2001, pp. 350-361.
2. J. Sugita and T. Takahashi, A Generation Method for Colored-Paper-Mosaic Images Based on Hierarchial Poisson Disk Sampling, *Journal of IIEEJ*. **36**(4), 2007, pp. 407-416.
3. Y. Takada and K. Urahama, Dendritic Line Images Generated from Distance Transform and its Application to NPR, *Journal of IIEEJ*. **38**(5), 2009, pp. 791-794.
4. T. Saito, Non-photorealistic Rendering, *Journal of IIEEJ*. **39**(6), 2010, pp. 837-839.
5. R. Ando and R. Tsuruno, An Interactive Brush Stroke Synthesis Using Exemplar Images, *Journal of IIEEJ*. **40**(4), 2011, pp. 578-586.
6. T. Hiraoka and K. Urahama, Generating Striped Color Images by Inverse Line Convergence Index Filter, *IIEEJ Transactions on Image Electronics and Visual Computing*. **2**(2), 2014, pp. 190-194.
7. B. J. Meier, Painterly Rendering for Animation, *Proceedings of ACM SIGGRAPH 1996*, 1996, pp. 477-484.
8. H. Kobatake and S. Hashimoto, Convergence Index Filter for Vector Fields, *IEEE Transactions Image Processing*. **8**, 1999.
9. The Video-Database-Working Group of PRMU, Video Database for Evaluating Video Processing, http://www.nii.ac.jp/dsc/idr/video/VDB/documentary.html, Yuzenzo-me, 2002.