IEICE TRANS. INF. & SYST., VOL.E103–D, NO.3 MARCH 2020

LETTER

A High-Speed Method for Generating Edge-Preserving Bubble Images

Toru HIRAOKA[†a)], ***Member***

SUMMARY We propose a non-photorealistic rendering method for generating edge-preserving bubble images from gray-scale photographic images. Bubble images are non-photorealistic images embedded in many bubbles, and edge-preserving bubble images are bubble images where edges in photographic images are preserved. The proposed method is executed by an iterative processing using absolute difference in window. The proposed method has features that processing is simple and fast. To validate the effectiveness of the proposed method, experiments using various photographic images are conducted. Results show that the proposed method can generate edge-preserving bubble images by preserving the edges of photographic images and the processing speed is high.

key words: *non-photorealistic rendering, bubble, edge preservation, high speed*

1. Introduction

In recent years, a technique called non-photorealistic rendering (NPR) has attracted attention in the field of computer graphics [1]–[4]. NPR converts photographic images, videos and three-dimensional data to non-photorealistic images. NPR is implemented as applications on mobile terminals such as smartphones or tablets, so that users can easily use the NPR applications. For the NPR applications, high-speed processing is required so that users can comfortably use them. One research on NPR is to generate bubble images from gray-scale photographic images using additive and multiplicative averages in different window sizes [5]. Bubble images are non-photorealistic images embedded in many bubbles. Such bubble images are uploaded by converting photographic images with the recent spread of SNS (Social Networking Service). Also, from the viewpoint of information security, when users want to make it difficult to identify people, products, signboards and locations from photographic images on SNS, users can use non-photorealistic images such as bubble images rather than simply blurring or mosaicing.

In this paper, we focus on NPR generating bubble images from gray-scale photographic images, and propose a simple method that can process faster than the conventional method [5]. In addition, since the conventional method generates bubble pattern across edges, we develop the method to generate edge-preserving bubble (EPB) images with edges preserved. The expression items necessary for expressing EPB images are that it is composed of patterns with a fine curved shape, and the patterns are generated along the edges in photographic images. The proposed method is executed by an iterative processing using absolute difference in window. To visually verify that the edges of photographic images can be preserved, experiment using various images is conducted. And, experiment to compare the calculation times between the proposed and conventional methods is conducted. As a result of the experiments, it is revealed that EPB images can be automatically generated by preserving the edges and the proposed method can process at high speed.

This paper is organized as follows: the second section describes the proposed method for generating EPB images, the third section shows experimental results and reveals the effectiveness of the proposed method, and the conclusion of this paper is given in the fourth section.

2. Proposed Method

The proposed method is executed by an iterative processing consisting of two steps. In the first step, averages of absolute differences in the window are calculated. In the second step, pixel values are updated using the result of the first step. A flow chart of the proposed method is shown in Fig. 1. It turns out that the proposed method does not use particularly difficult processing.

Details of the steps in Fig. 1 are shown below.

Step 0 Let the input pixel values on coordinates (i, j) of a gray-scale photographic image be $f_{i,j}$. The pixel values $f_{i,j}$ have value of 256 gradation from 0 to 255.

Step 1 The averages of the absolute differences $a_{i,j}^{(t)}$ are calculated using pixel values $f_{i,j}^{(t)}$ in the window of $2W+1$

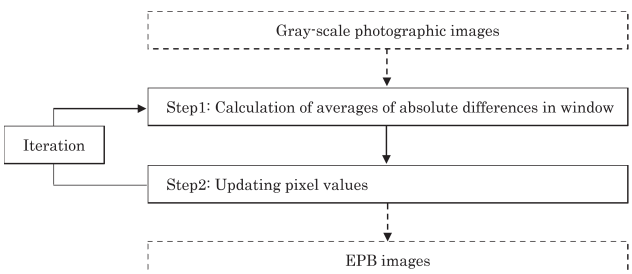


Fig. 1 Flow chart of the proposed method.

Manuscript received October 16, 2019.
Manuscript revised November 6, 2019.
Manuscript publicized November 29, 2019.

[†]The author is with the Faculty of Information System, University of Nagasaki, Nagasaki-ken, 851–2195 Japan.
a) E-mail: hiraoka@sun.ac.jp

DOI: 10.1587/transinf.2019EDL8188

Copyright © 2020 The Institute of Electronics, Information and Communication Engineers

centered on the pixel (i, j), where $t(= 1, 2, \cdots)$ is the number of iterations and $f_{i,j}^{(1)} = f_{i,j}$. The averages $a_{i,j}^{(t)}$ are calculated as follows.

$$a_{i,j}^{(t)} = \frac{\sum_{k=-W}^{W} \sum_{l=-W}^{W} |f_{i+k,j+l}^{(t)} - f_{i-k,j-l}^{(t)}|}{(2W+1)^2 - 1} \quad (1)$$

where k and l are the positions in the window.

Step 2 The pixel values $f_{i,j}^{(t)}$ are updated using the averages $a_{i,j}^{(t)}$ as follows.

$$f_{i,j}^{(t+1)} = \begin{cases} f_{i,j} - ba_{i,j}^{(t)} & (t\%2 = 0) \\ f_{i,j} + ba_{i,j}^{(t)} & (t\%2 = 1) \end{cases} \quad (2)$$

where b is a positive constant, and % is the remainder operator. In case $f_{i,j}^{(t+1)}$ is less than 0, then $f_{i,j}^{(t+1)}$ must be set to $-f_{i,j}^{(t+1)}$ and further $f_{i,j}^{(t+1)}$ must be set to 255 in case $-f_{i,j}^{(t+1)}$ is greater than 255. In case $f_{i,j}^{(t+1)}$ is greater than 255, then $f_{i,j}^{(t+1)}$ must be set to $255+(255-f_{i,j}^{(t+1)})/b$. The processing of Steps 1 and 2 is repeated T times, where T is an even number. An image composed of pixel values $f_{i,j}^{(T)}$ is an EPB image.

3. Experiments

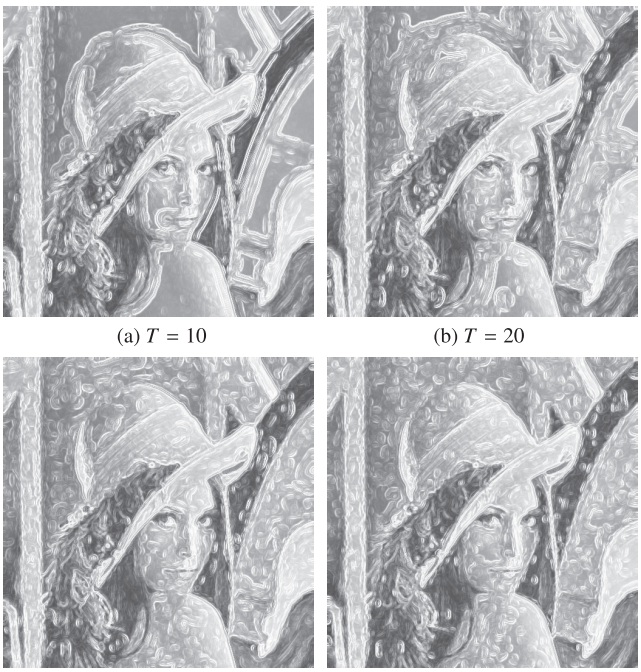
Three experiments were conducted mainly. First, the proposed method was applied to Lenna image shown in Fig. 2. The changes in appearance to EPB images were visually assessed as the values of the parameters were varied. Next, the proposed method were applied to various images. Finally, the calculation times between the proposed and conventional methods were compared. In the experiments, unless otherwise noted, the values of T, W and b were set to 40, 2 and 2.0, respectively. All images used in the experiments were 512 * 512 pixels and 256 gradation. The computing environment for all experiments was Windows 10 Enterprise 2016 LTSB operating system on a computer with 3.20 GHz CPU, 8.00 GB of memory. The programming language used was VC++.

First, EPB images by varying the value of the iteration number T were confirmed visually using Lenna image. The value of T was set to 10, 20, 30 and 40. The results of the experiment are shown in Fig. 3. As the value of T was larger, EPB patterns changed to the patterns with the fine curved shape and were generated along the edges in Lenna image.

Fig. 2 Lenna image.

Next, EPB images by varying the value of the window size W were confirmed visually using Lenna image. The value of W was set to 1, 2, 3 and 4. The results of the experiment are shown in Fig. 4. As the value of W was larger, the size of EPB patterns became bigger.

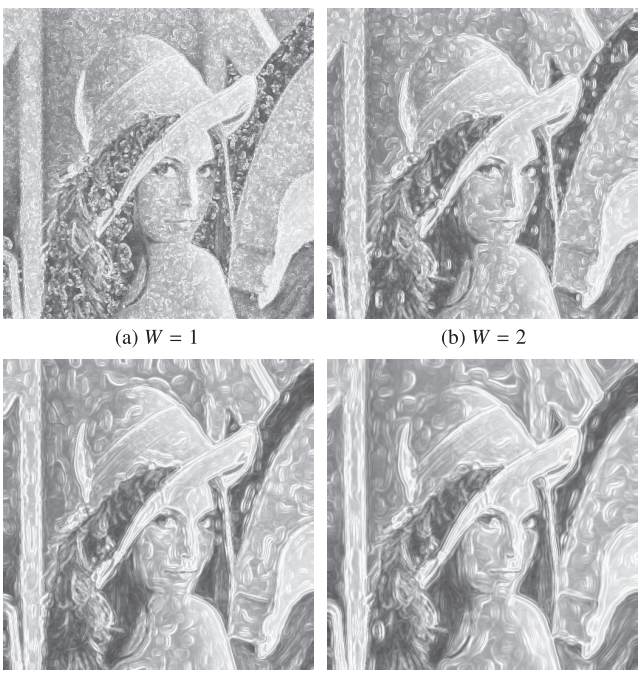
Next, EPB images by varying the value of the parame-



(a) $T = 10$ (b) $T = 20$

(c) $T = 30$ (d) $T = 40$

Fig. 3 EPB images for $T = 10, 20, 30$ and 40.



(a) $W = 1$ (b) $W = 2$

(c) $W = 3$ (d) $W = 4$

Fig. 4 EPB images for $W = 1, 2, 3$ and 4.

(a) $b = 1.5$ (b) $b = 2.0$

(c) $b = 2.5$ (d) $b = 3.0$

Fig. 5 EPB images for $b = 1.5, 2.0, 2.5$ and 3.0.

ter b were confirmed visually using Lenna image. The value of b was set to 1.5, 2.0, 2.5 and 3.0. The results of the experiment are shown in Fig. 5. As the value of b was larger, more EPB patterns were generated. On the other hand, as the value of b became too small, EPB patterns were not generated. If users want to generate EPB images so that photographic images can be recognized, the value of b should be set between 2.0 and 3.0. On the other hand, if users want to make it difficult to recognize photographic images from the viewpoint of information security, the value of b should be set to 2.0 or more.

Next, the proposed method was applied to four photographic images shown in Fig. 6. The results are shown in Fig. 7. In all cases, EPB images were composed of patterns with the fine curve shape, and EPB patterns could be automatically generated by preserving the edges in photographic images. However, EPB patterns were less likely to occur in areas where textures are fine such as the upper right area of the upper right image in Fig. 7. In addition, as a result of evaluating EPB images by a researcher who studied in art such as watercolor painting and ink painting at university and now specializes in design and chromatics, the following comments were received.

- It has a unique touch that is different from any filters such as watercolors, oil paintings, stained glass painting and Fresco that are embedded in Adobe Photoshop and other applications.
- It is balanced with the eye-catching effect while maintaining the original information.

Finally, the calculation times between the proposed and conventional methods were compared quantitatively using

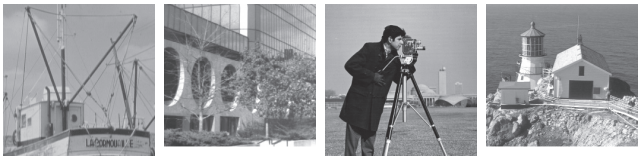


Fig. 6 Various photographic images.

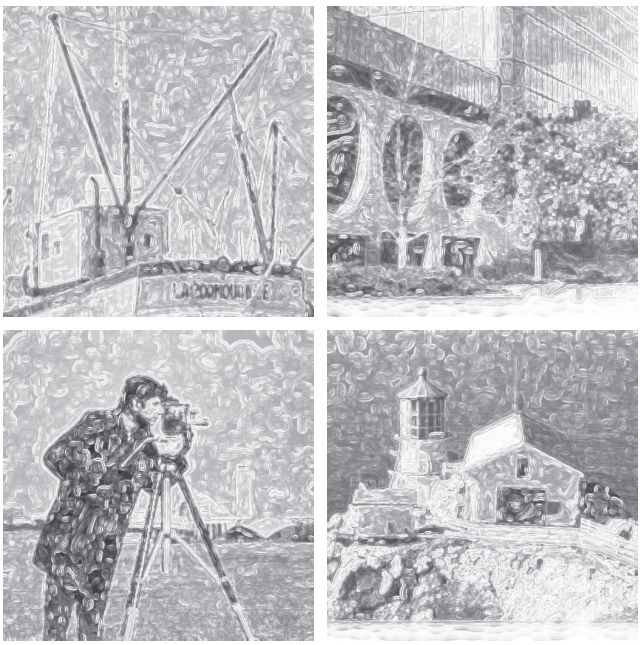


Fig. 7 EPB images.

Table 1 Calculation times of the proposed method.

Window size W	Calculation time [seconds]
1	1.618
2	4.106
3	7.932
4	12.940

Table 2 Calculation times of the conventional method.

Window size W	Calculation time [seconds]
1	9.344
2	14.857
3	22.437
4	31.832

Lenna image. In the experiments of the proposed and conventional methods, the value of the window size W was set to 1, 2, 3 and 4. The results of the proposed and conventional methods are shown in Tables 1 and 2, respectively. The calculation times of the proposed method were 1.618, 4.106, 7.932 and 12.940 seconds for W =1, 2, 3 and 4, respectively. The calculation times of the conventional method were 9.344, 14.857, 22.437 and 31.832 seconds for W =1, 2, 3 and 4, respectively. Thus, the proposed method could speed up the processing approximately three times as compared with the conventional method.

4. Conclusion

We proposed a method for generating EPB Images from gray-scale photographic images by an iterative processing using absolute difference in window. The proposed method had features that processing is fast and EPB patterns can be automatically generated by preserving the edges in photographic images. In the experiments using Lenna image and other photographic images, it was clarified that the proposed method can practically realize these features. In addition, it was clarified how to change EPB patterns when the values of the parameters in the proposed method were varied. A subject for future study is to make it possible to generate EPB patterns in fine texture areas as well.

Acknowledgments

This work was supported by JSPS KAKENHI Grant Number JP19K12664.

References

[1] D. Jönsson, E. Sundén, A. Ynnerman, and T. Ropinski, "A survey of Volumetric Illumination Techniques for Interactive Volume Rendering," Comput. Graph. Forum, vol.33, no.1, pp.27–51, 2014.

[2] L.A. Gatys, A.S. Ecker, and M. Bethge, "Image Style Transfer Using Convolutional Neural Networks," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.2414–2423, 2016.

[3] W. Qian, D. Xu, K. Yue, Z. Guan, Y. Pu, and Y. Shi, "Gourd Pyrography Art Simulating Based on Non-Photorealistic Rendering," Multimed. Tools. Appl., vol.76, no.13, pp.14559–14579, 2017.

[4] T. Wu, "Saliency-aware generative art," Proc. 2018 10th International Conference on Machine Learning and Computing, pp.198–202, 2018.

[5] T. Hiraoka and K. Urahama, "Generation of bubble images using additive and multiplicative averages in different window sizes," ICIC Express Letters, vol.13, no.6, pp.469–474, 2019.