ICIC Express Letters
Volume **13**, Number **12**, December **2019**

ICIC International ©2019 ISSN 1881-803X
pp. 1097–1101

GENERATION OF RIPPLE IMAGES USING INTENSITY GRADIENT

Toru Hiraoka

Department of Information Systems
University of Nagasaki
1-1-1, Manabino, Nagayo-chou, Nishisonogi-gun, Nagasaki-ken 851-2195, Japan
hiraoka@sun.ac.jp

Received May 2019; accepted August 2019

Abstract. *We propose a non-photorealistic rendering method for generating ripple images from gray-scale photographic images. In our method, an iteration processing with intensity gradients is applied to inputting gray-scale photographic images for transforming it to ripple images. In this case, we show how to generate ripple patterns in horizontal and vertical directions. To verify the effectiveness of our method, we conducted experiments using various photographic images by changing the directions and the values of the parameters in our method, and then we visually evaluated the appearance of ripple images. Potential applications of our method range from magazines, TVs, websites, to social networking services.*
Keywords: Non-photorealistic rendering, Ripple pattern, Intensity gradient, Horizontal and vertical directions

1. **Introduction.** There are many active studies on image processing [1, 2, 3]. One of those studies is non-photorealistic rendering. Non-photorealistic rendering is a technology that converts photographic images or three-dimensional data into non-photorealistic images such as pencil drawings, watercolors, and oil paintings [4, 5, 6]. In recent years, many methods for generating non-photorealistic images represented by geometric patterns have been proposed. As examples of geometric non-photorealistic images, there are op-art images [7, 8], reaction-diffusion-pattern images [9], stripe-patchwork images [10], and contour images [11]. Inglis et al. [7] developed an algorithm that converts a coloured planar map into line-based and curve-based op-art images. Ahmed [8] developed an algorithm that generates op-art and labyrinth images based on the original Truchet tiles from input maps of up to 6 colors. Chi et al. [9] developed an algorithm that generates reaction-diffusion-pattern images based on anisotropic reaction diffusion including paper-cut, stylized halftone, and motion illusion. Hiraoka et al. [10, 11] developed algorithms that convert stripe-patchwork and contour images from photographic images using smoothing filter. Such geometric non-photorealistic images can be used in websites and social networking services for entertainment and amusement.

We focus on ripple images as geometric non-photorealistic images, and then propose a simple method for generating ripple images from gray-scale photographic images. Ripple images are expressed by superimposing ripple patterns on photographic images. Ripple patterns imitate a wave on the water surface and are composed of continuous lines with fluctuations. The development of non-photorealistic rendering technology for generating ripple images broadens the range of applications for users. The proposed method is executed by an iterative processing with intensity gradients. The proposed method is easy to implement, and can generate ripple patterns in horizontal and vertical directions. To verify the effectiveness of the proposed method, experiments were conducted using Lenna

DOI: 10.24507/icicel.13.12.1097

image and other photographic images. As a result of the experiments, it was clarified that the proposed method can generate ripple patterns horizontally and vertically.

The rest of this paper is organized as follows. Section 2 describes the proposed method for generating ripple images. Section 3 shows experimental results, and reveals the effectiveness of the proposed method. Finally, Section 4 concludes this paper.

2. **Proposed Method.** The proposed method generates ripple images from gray-scale photographic images. The proposed method is executed in two processes. In the first process, the intensity gradients are calculated. In the second process, images are updated using the intensity gradients. By repeating the two processes, ripple images are generated. The proposed method does not use particularly difficult processing, and then is easy to implement. A flow chart of the proposed method is shown in Figure 1.

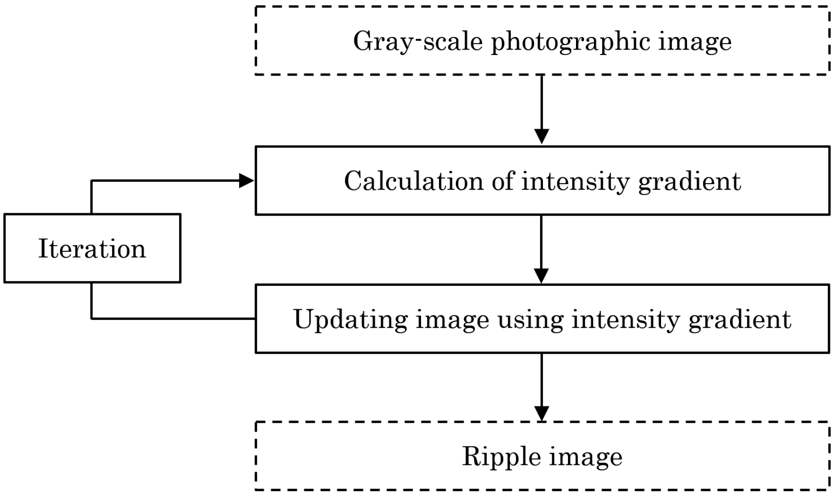


FIGURE 1. Flow chart of the proposed method

The detailed procedure in Figure 1 is shown as follows.

Definition: The input pixel values for spatial coordinates (i, j) of a gray-scale photographic image are defined as $f_{i,j}$. Then, the pixel values of the image at the t-th iteration number are defined as $f_{i,j}^{(t)}$, where $f_{i,j}^{(1)} = f_{i,j}$. The pixel values $f_{i,j}^{(t)}$ have value of U gradation from 0 to $U-1$.

Process 1: The horizontal and vertical intensity gradients at the pixel (i, j) are defined as $g_{x,i,j}^{(t)}$ and $g_{y,i,j}^{(t)}$, respectively. The horizontal and vertical intensity gradients are calculated using the pixel values in the window of W pixels as the following equations.

$$g_{x,i,j}^{(t)} = \sum_{k=-W}^{-1} \sum_{l=-W}^{W} \left(f_{i,j+l}^{(t)} - f_{i+k,j+l}^{(t)} \right) + \sum_{k=1}^{W} \sum_{l=-W}^{W} \left(f_{i+k,j+l}^{(t)} - f_{i,j+l}^{(t)} \right) \tag{1}$$

$$g_{y,i,j}^{(t)} = \sum_{k=-W}^{W} \sum_{l=-W}^{-1} \left(f_{i+k,j}^{(t)} - f_{i+k,j+l}^{(t)} \right) + \sum_{k=-W}^{W} \sum_{l=1}^{W} \left(f_{i+k,j+l}^{(t)} - f_{i+k,j}^{(t)} \right) \tag{2}$$

where k and l are the positions in the window. When ripple patterns are generated horizontally, the intensity gradients $g_{i,j}^{(t)}$ are calculated as the following equation.

$$g_{i,j}^{(t)} = \frac{g_{y,i,j}^{(t)}}{\sqrt{{g_{x,i,j}^{(t)}}^2 + {g_{y,i,j}^{(t)}}^2}} \tag{3}$$

On the other hand, when ripple patterns are generated vertically, the intensity gradients $g_{i,j}^{(t)}$ are calculated as the following equation.

$$g_{i,j}^{(t)} = \frac{g_{x,i,j}^{(t)}}{\sqrt{{g_{x,i,j}^{(t)}}^2 + {g_{y,i,j}^{(t)}}^2}} \quad (4)$$

Process 2: The pixel values $f_{i,j}^{(t)}$ are updated to $f_{i,j}^{(t+1)}$ as the following equation.

$$f_{i,j}^{(t+1)} = f_{i,j} + a\frac{f_{i,j}^{(t)} - g_{\min,i,j}^{(t)}}{g_{\max,i,j}^{(t)} - g_{\min,i,j}^{(t)}} - \frac{a}{2} \quad (5)$$

where a is a positive constant, and $g_{\min,i,j}^{(t)}$ and $g_{\max,i,j}^{(t)}$ are the minimum and maximum values of $g_{i,j}^{(t)}$, respectively. In case $f_{i,j}^{(t+1)}$ is less than 0, then $f_{i,j}^{(t+1)}$ must be set to 0. In case $f_{i,j}^{(t+1)}$ is greater than $U-1$, then $f_{i,j}^{(t+1)}$ must be set to $U-1$. A ripple image is obtained after the above processing of T iterations.

3. **Experiments.** Two experiments were conducted mainly. The first experiment with changing the values of the parameters in the proposed method was conducted using Lenna image shown in Figure 2. The second experiment was conducted to verify that ripple patterns can be generated using various photographic images. All photographic images used in all experiments were 512×512 pixels and 256 gradation. Since all ripple images converged with 10 iterations, the value of the iteration number T was set to 10 in all experiments.

FIGURE 2. Lenna image

3.1. **Experiment with different parameter values.** Ripple images by changing the window size W were visually confirmed using Lenna image. The window size W was set to 1, 2, 3, 4, and 5. The parameter a was set to 128. The results of the experiment are shown in Figure 3. As the value of the window size W was larger, the width of ripple patterns became wider.

Ripple images by changing the parameter a were visually confirmed using Lenna image. The parameter a was set to 32, 64, 96, 128, and 160. The window size W was set to 3. The results of the experiment are shown in Figure 4. As the value of the parameter a was larger, ripple patterns became clear.

3.2. **Experiment using various photographic images.** The proposed method was applied to five photographic images shown in Figure 5. Since ripple patterns were visually recognized well in the previous experiments, the parameters W and a were set to 3 and 128, respectively. The results of the experiment are shown in Figure 6. In all cases, ripple patterns could be automatically generated in horizontal and vertical directions on the whole image.

(1) $W = 1$ (2) $W = 2$ (3) $W = 3$ (4) $W = 4$ (5) $W = 5$

(a) Case that generated ripple patterns horizontally

(1) $W = 1$ (2) $W = 2$ (3) $W = 3$ (4) $W = 4$ (5) $W = 5$

(b) Case that generated ripple patterns vertically

FIGURE 3. Ripple images in the case of the window size $W = 1, 2, 3, 4$, and 5

(1) $a = 32$ (2) $a = 64$ (3) $a = 96$ (4) $a = 128$ (5) $a = 160$

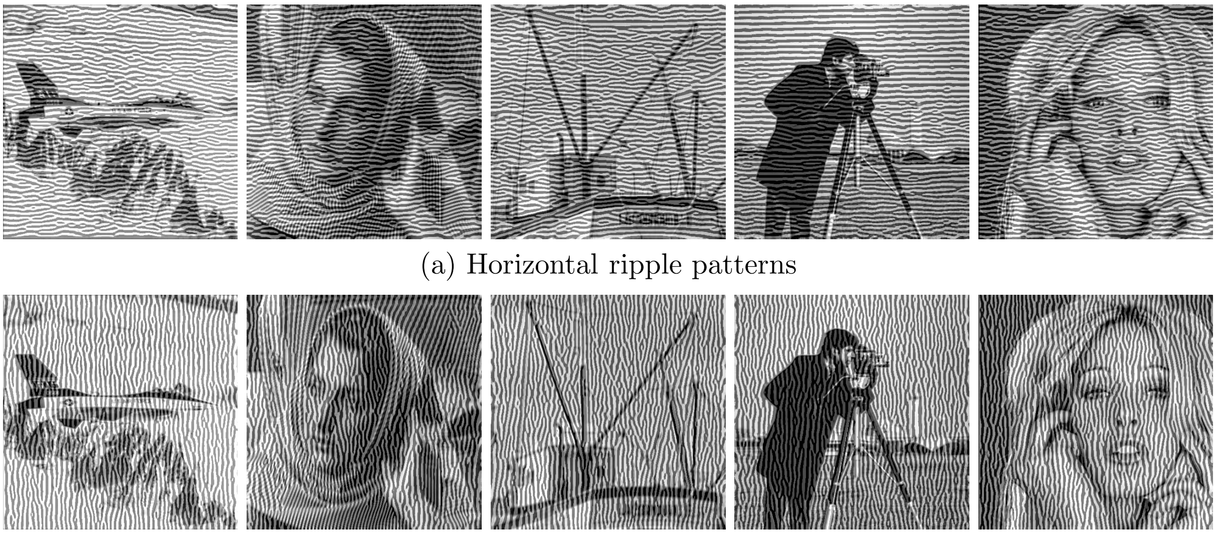
(a) Case that generated ripple patterns horizontally

(1) $a = 32$ (2) $a = 64$ (3) $a = 96$ (4) $a = 128$ (5) $a = 160$

(b) Case that generated ripple patterns vertically

FIGURE 4. Ripple images in the case of the parameter $a = 32$, 64, 96, 128, and 160

FIGURE 5. Various photographic images



(a) Horizontal ripple patterns

(b) Vertical ripple patterns

Figure 6. Ripple images generated from various photographic images in Figure 5

4. **Conclusions.** We proposed a new non-photorealistic rendering method for ripple images from gray-scale photographic images. The proposed method was executed by an iterative processing with intensity gradients. The proposed method had two features that it is easy to process and can generate ripple patterns in horizontal and vertical directions. To verify the effectiveness of the proposed method, experiments were conducted using various photographic images. The experiments results show that the proposed method satisfies two features.

A subject for future study is to expand the proposed method for application to color photographic images and videos.

REFERENCES

[1] K. S. Sim and F. Sammani, Deep convolutional networks for magnification of DICOM brain images, *International Journal of Innovative Computing, Information and Control*, vol.15, no.2, pp.725-739, 2019.

[2] P. Maniriho and T. Ahmad, High quality PVM based reversible data hiding method for digital images, *International Journal of Innovative Computing, Information and Control*, vol.15, no.2, pp.667-680, 2019.

[3] J. Fu, Y. Huang, J. Xu and H. Wu, Optimization of distributed convolutional neural network for image labeling on asynchronous GPU model, *International Journal of Innovative Computing, Information and Control*, vol.15, no.3, pp.1145-1156, 2019.

[4] P. Haeberli, Paint by numbers: Abstract image representations, *ACM SIGGRAPH Computer Graphics*, vol.24, no.4, pp.207-214, 1990.

[5] J. Lansdown and S. Schofield, Expressive rendering: A review of nonphotorealistic techniques, *IEEE Computer Graphics and Applications*, vol.15, no.3, pp.29-37, 1995.

[6] D. Martin, G. Arroyo, A. Rodriguez and T. Isenberg, A survey of digital stippling, *Computers & Graphics*, vol.67, pp.24-44, 2017.

[7] T. C. Inglis, S. Inglis and C. S. Kaplan, Op art rendering with lines and curves, *Computers & Graphics*, vol.36, no.6, pp.607-621, 2012.

[8] A. G. M. Ahmed, Line-based rendering with Truchet-like tiles, *Proc. of the Workshop on Computational Aesthetics (CAe'14)*, pp.41-51, 2014.

[9] M. T. Chi, W. C. Liu and S. H. Hsu, Image stylization using anisotropic reaction diffusion, *The Visual Computer*, vol.32, no.12, pp.1549-1561, 2016.

[10] T. Hiraoka, H. Nonaka and Y. Tsurunari, A method for controlling patterns of stripe-patchwork images, *ICIC Express Letters*, vol.13, no.3, pp.175-180, 2019.

[11] T. Hiraoka, H. Nonaka and Y. Tsurunari, Generation of contour-like images by smoothing filter and Laplacian filter, *ICIC Express Letters*, vol.13, no.4, pp.263-268, 2019.