ICIC Express Letters
Volume **16**, Number **12**, December **2022**
 ISSN 1881-803X
pp. **1315–1322**

GENERATING PIXEL ART WITH DECORATIVE PIXEL PATTERNS

Jianwei Nie[1], Wenyi Cui[1], Kohei Inoue[1,*] and Toru Hiraoka[2]

[1]Faculty of Design
Kyushu University
4-9-1 Shiobaru, Minami-ku, Fukuoka 815-8540, Japan
{ nie.jianwei.301; cui.wenyi.333 }@s.kyushu-u.ac.jp
*Corresponding author: k-inoue@design.kyushu-u.ac.jp

[2]Faculty of Information Systems
University of Nagasaki
1-1-1 Manabino, Nagayo-cho, Nishi-Sonogi-gun, Nagasaki 851-2195, Japan
hiraoka@sun.ac.jp

Received May 2022; accepted July 2022

Abstract. *Image pixelization makes pixels visible by enlarging each pixel and reducing the number of pixels in an image. It is a challenging task to represent an image content with a limited number of pixels, and fine pixelization products such as pixel art and icons on computer screens often attract people. In this paper, we propose an expression technique for image pixelization, where each pixel in a pixelized image can be decorated with various styles specified by users. For example, we develop three methods for generating checkered, woven and concentric squares patterns. Experimental results show that a variety of styles of pixelized images are obtained by the proposed methods.*
Keywords: Image pixelization, Pixel art, Image stylization, Pixel decoration, Non-photorealistic rendering

1. **Introduction.** Image pixelization is a fundamental technique in digital image processing used for image obfuscation as well as image blurring to protect privacy, image downscaling to have a low-resolution representation in a content-aware manner, and image abstraction to create fine pixel art which is nowadays recognized as a kind of contemporary art form.

Goldberg and Flegal first published the term 'pixel art' in 1982 [1]. Yu presented a fast-paced tutorial for making pixel art through the creation of a sprite [2]. Fan proposed a private image pixelization which extends differential privacy [3] to image data publication [4]. Kopf et al. proposed a content-adaptive image downscaling method with an adaptive downsampling kernel that combines a spatial and a range kernels following bilateral filter [5] and mean shift [6], and showed that their careful sampling can preserve certain high frequencies features in downscaled image without artifacts [7]. Öztireli and Gross [8] formulated image downscaling as an optimization problem, and derived the closed-form solution, where the error between the input and output images is measured by SSIM (structural similarity) index [9]. Gerstner et al. proposed an automated process that transforms high resolution images into low resolution, small palette outputs in a pixel art style [10], where a superpixel algorithm SLIC (simple linear iterative clustering) [11] and MCDA (mass-constrained deterministic annealing) [12] are utilized. These researches of forty years' duration demonstrate the usefulness of image pixelization and a wide range of interests in pixel art, and, this day, we are free from the technological limitations of the past. Therefore, it would be a meaningful challenge to develop new pixel-artistic expressions.

DOI: 10.24507/icicel.16.12.1315

Inglis and Kaplan [13] presented a pixelation algorithm named Superpixelator that converts vector line art into pixel line art based on Bresenham's algorithm [14]. Kopf and Lischinski proposed an algorithm for extracting a resolution-independent vector representation from pixel art images [15]. Han et al. proposed an unsupervised learning method for pixelization, where preparing the paired training data for supervised learning is impractical due to the difficulty in creating pixel art [16]. These researches also suggest the importance of developing the technologies related to pixel art.

As mentioned above, image pixelization and its related fields attract much attention from researchers. In this paper, we propose a non-photorealistic rendering scheme for image pixelization, where each pixel conveys more information than a single color as in normal color images. That is, each pixel becomes visible by being represented with more than one pixel or a square block of pixels, in which each pixel can have different colors to each other. As a result, we can extend the range of expression in pixelized images, i.e., we can decorate each pixel in image pixelization in various ways. Specifically, we present three types of pixel decoration: checkered, woven and concentric squares patterns. Experimental results demonstrate that the proposed methods can provide novel expressions for pixelized images.

The rest of this paper is organized as follows. Section 2 describes the proposed methods for making checkered, woven and concentric squares patterns. Section 3 shows experimental results with real images. Finally, Section 4 concludes this paper.

2. Proposed Methods.

In this section, we propose three decorative patterns for image pixelization: checkered, woven and concentric squares patterns.

Let $F = [\boldsymbol{f}_{ij}]$ be an RGB color image with $m \times n$ pixels, where $\boldsymbol{f}_{ij}$ denotes the RGB color vector of a pixel in F with coordinates (i, j) for $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, n$. Then we divide the color image F into regular and square blocks of equal size, $h \times h$ pixels, and get $M \times N$ blocks where $M = \lfloor m/h \rfloor$ and $N = \lfloor n/h \rfloor$ with the floor function $\lfloor x \rfloor$ which returns the greatest integer less than or equal to x. We omit the residual rows and columns for simplifying the following procedures. Each block will be converted into different patterns in three ways as described in the following subsections.

2.1. Checkered pattern.

Let $\boldsymbol{f}_k = [r_k, g_k, b_k]$ for $k = 1, 2, \ldots, h^2$ be the kth RGB color vector in a block above, and $x_k \in \{r_k, g_k, b_k\}$ be an element of $\boldsymbol{f}_k$, where we assume that $x_k \in \{0, 1, \ldots, 255\}$. Then we first compute a mean color as $\bar{\boldsymbol{f}} = \sum_{k=1}^{h^2} \boldsymbol{f}_k / h^2$. Next, we expand the distribution of h^2 colors in the block in the RGB color space by

$$\boldsymbol{f}'_k = \alpha \left(\boldsymbol{f}_k - \bar{\boldsymbol{f}} \right) + \bar{\boldsymbol{f}}, \tag{1}$$

where α is a variable to be maximized under the condition that $\boldsymbol{f}'_k \in [0, 255]^3$. For an element x_k in $\boldsymbol{f}_k$, (1) means that

$$x'_k = \alpha \left(x_k - \bar{x} \right) + \bar{x}. \tag{2}$$

If $\boldsymbol{f}'_k$ is on the boundary of the RGB color cube, then there is at least one element of $\boldsymbol{f}'_k$ satisfying $x'_k = 0$ or $x'_k = 255$. Taking this into consideration, we can solve (2) for α as follows:

$$\alpha_{x,k} = \begin{cases} \dfrac{-\bar{x}}{x_k - \bar{x}} & \text{if } x_k - \bar{x} < 0 \\ \dfrac{255 - \bar{x}}{x_k - \bar{x}} & \text{if } x_k - \bar{x} > 0 \\ 0 & \text{if } x_k - \bar{x} = 0. \end{cases} \tag{3}$$

To keep all colors within the color cube, we select the minimal $\alpha_{x,k}$ as

$$\alpha = \min_{x \in \{r,g,b\},\ k \in \{0,1,\ldots,h^2\}} \left\{ \alpha_{x,k} \right\}. \tag{4}$$

Figure 1 illustrates a checkered pattern composed of 3×4 blocks, where each block is indexed by (I, J) for $I = 0, 1, \ldots, M-1$ and $J = 0, 1, \ldots, N-1$ where $M = 3$ and $N = 4$. In this figure, each block is painted white if $I + J$ is an even number, or black otherwise. We decorate this two kinds of blocks in different ways as follows: For each block at (I, J), substituting α in (4) into (1), we have $\boldsymbol{f}'_k = [r'_k, g'_k, b'_k]$, from which we compute $\bar{x}'_k = (r'_k + g'_k + b'_k)/3$. If $I + J$ is an even number, then we sort $\left\{\boldsymbol{f}'_1, \boldsymbol{f}'_2, \ldots, \boldsymbol{f}'_{h^2}\right\}$ in the ascending order of $\left\{\bar{x}'_1, \bar{x}'_2, \ldots, \bar{x}'_{h^2}\right\}$, or in the descending order otherwise. Then we rearrange the colors in the block in a spiral. For example, we show a color block in Figure 2(a). If we rearrange the pixels in descending order from outside to inside, then we have the result in Figure 2(b). On the other hand, if we rearrange them in ascending order from outside to inside, then we have the result in Figure 2(c).

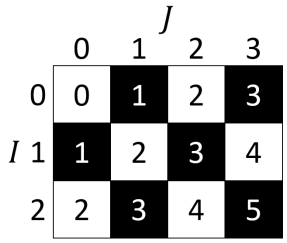
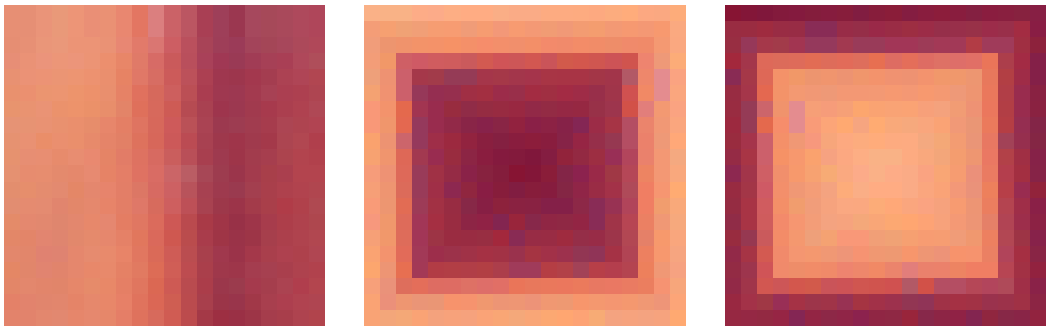


FIGURE 1. A 3×4 checkered pattern with the numbers $I + J$



(a) Color block (b) Descending order (c) Ascending order

FIGURE 2. Pixel rearrangement: (a) A color block of 20×20 pixels, (b) the rearrangement in descending order from the outside to the inside, and (c) the rearrangement in ascending order

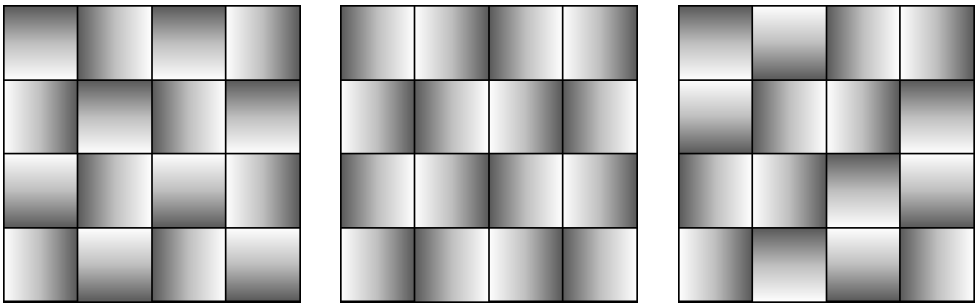
We replace all $M \times N$ blocks in an input image F with the generated patterns by the above procedure to have an image pixelization with a checkered pattern.

2.2. **Woven pattern.** Next, we consider the way to express weave patterns with the technique of decorative pixel patterns. We express the direction of a fiber by the gradient of the intensity x_k, i.e., the sorted colors $\left\{\boldsymbol{f}'_1, \boldsymbol{f}'_2, \ldots, \boldsymbol{f}'_{h^2}\right\}$ based on $\left\{\bar{x}'_1, \bar{x}'_2, \ldots, \bar{x}'_{h^2}\right\}$ are rearranged so that the direction of gradation coincides with that of fiber.

Figure 3 shows three weave patterns: (a) plain, (b) Dutch and (c) twill patterns. Each 4×4 pattern is laid over the image plane without gaps. Actually, in Figure 3(b), it is enough to keep only a 2×2 pattern for constructing the Dutch pattern.

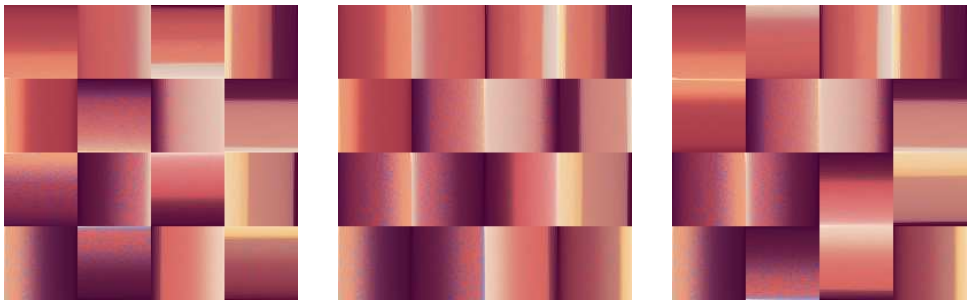
Figure 4 shows an example of generated weave patterns, where the input image is shown in Figure 7(a), the size of which is 256×256 pixels, and that of a block is 64×64 pixels.

We replace all $M \times N$ blocks in an image F with a specified weave pattern to have a decorative image pixelization composed of the weave pattern.



(a) Plain (b) Dutch (c) Twill

FIGURE 3. Weave patterns are expressed by the gradient of intensity x_k.



(a) Plain (b) Dutch (c) Twill

FIGURE 4. Example of weave patterns generated from the image in Figure 7(a)

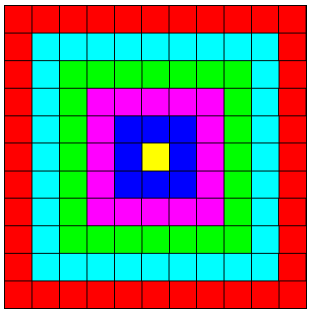


FIGURE 5. (color online) A block composed of six concentric squares, the areas of which are 1, 8, 16, 24, 32 and 40 pixels from the center to outside

2.3. **Concentric squares.** Thirdly, we propose a square pattern with concentric squares, where the colors of six concentric squares are determined so that the mean color in a square block is preserved after the pixel decoration procedure. Figure 5 shows the six concentric squares with different colors, where the numbers of pixels with the same color are 1, 8, 16, 24, 32 and 40 from the center to outside. Therefore, we have three ratios of the areas of adjacent squares as $1:8$, $16:24$ and $32:40$, that determine the colors of squares as follows: Let $\bar{\boldsymbol{f}} = [\bar{r}, \bar{g}, \bar{b}]$ be the mean color of a block with 11×11 pixels as illustrated in Figure 5. Then we compute three ratios as $\gamma_R = \min\left\{\frac{\bar{r}}{255-\bar{r}}, \frac{255-\bar{r}}{\bar{r}}\right\}$, $\gamma_G = \min\left\{\frac{\bar{g}}{255-\bar{g}}, \frac{255-\bar{g}}{\bar{g}}\right\}$ and $\gamma_B = \min\left\{\frac{\bar{b}}{255-\bar{b}}, \frac{255-\bar{b}}{\bar{b}}\right\}$, and assume that $\gamma_X \leq \gamma_Y \leq \gamma_Z$ for $\{X, Y, Z\} = \{R, G, B\}$, which corresponds to $\frac{1}{8} \leq \frac{16}{24} \leq \frac{32}{40}$. Let $\bar{x}$ be the mean value of an X channel in the block, which satisfies the inequality $\bar{x} < 255 - \bar{x}$ or $\bar{x} < \frac{255}{2}$. In this case, if $\frac{\bar{x}-0}{255-\bar{x}} > \frac{1}{8}$, then we solve $\frac{\bar{x}-x}{255-\bar{x}} = \frac{1}{8}$ for x to have $x = \bar{x} - \frac{1}{8}(255 - \bar{x})$, which satisfies $\bar{x} - x : 255 - \bar{x} = 1 : 8$. To keep this ratio on the image plane, we assign a modified version of the mean color $\bar{\boldsymbol{f}}$, the X channel of which is replaced to 255, to the central pixel in the

block (the yellow pixel in Figure 5), and the obtained value x is substituted into the X channel of the mean color $\bar{\boldsymbol{f}}$, and the modified mean color is assigned to the eight blue pixels in the block in Figure 5. On the other hand, if $\frac{\bar{x}-0}{255-\bar{x}} \leq \frac{1}{8}$, then we solve $\frac{\bar{x}-0}{x-\bar{x}} = \frac{1}{8}$ for x to have $x = \left(1+\frac{8}{1}\right)\bar{x}$, which satisfies $\bar{x}-0 : x-\bar{x} = 1 : 8$. To keep this ratio on the image plane, we assign a modified version of the mean color $\bar{\boldsymbol{f}}$, the X channel of which is replaced to x, to the central yellow pixel in Figure 5, and another modified version of the mean color $\bar{\boldsymbol{f}}$ where the X channel is replaced to 0 is assigned to the eight blue pixels in the block in Figure 5.

Next, we consider another situation that the mean value $\bar{x}$ satisfies $\bar{x} \geq 255-\bar{x}$ or $\bar{x} \geq \frac{255}{2}$. In this case, if $\frac{255-\bar{x}}{\bar{x}-0} > \frac{1}{8}$, then we solve $\frac{x-\bar{x}}{\bar{x}-0} = \frac{1}{8}$ for x to have $x = \left(1+\frac{1}{8}\right)\bar{x}$, which satisfies $x-\bar{x} : \bar{x}-0 = 1 : 8$. To keep this ratio on the image plane, we assign a modified version of the mean color $\bar{\boldsymbol{f}}$, the X channel of which is replaced to 0, to the central yellow pixel in Figure 5, and the obtained value x is substituted into the X channel of $\bar{\boldsymbol{f}}$, and the modified mean color is assigned to the eight blue pixels in the block in Figure 5. On the other hand, if $\frac{255-\bar{x}}{\bar{x}-0} < \frac{1}{8}$, then we solve $\frac{255-\bar{x}}{\bar{x}-x} = \frac{1}{8}$ for x to have $x = \bar{x} - \frac{8}{1}(255-\bar{x})$, which satisfies $255-\bar{x} : \bar{x}-0 = 1 : 8$.

For the remaining channels Y and Z, we can also determine the colors of squares denoted by magenta and green for Y channel, cyan and red for Z channel in Figure 5. Equations for channel value modification are summarized in Table 1.

TABLE 1. Equations for channel value modification for $\frac{P}{Q} = \frac{1}{8}$, $\frac{16}{24}$ and $\frac{32}{40}$. In the first step, each channel value $\bar{x}$ of a mean color $\bar{\boldsymbol{f}}$ is judged whether it is smaller than $\frac{255}{2}$ or not, and then, in the second step, each situation is further divided into two cases.

First	$\bar{x} < \frac{255}{2}$		$\bar{x} \geq \frac{255}{2}$	
Second	$\frac{\bar{x}-0}{255-\bar{x}} > \frac{P}{Q}$	$\frac{\bar{x}-0}{255-\bar{x}} \leq \frac{P}{Q}$	$\frac{255-\bar{x}}{\bar{x}-0} > \frac{P}{Q}$	$\frac{255-\bar{x}}{\bar{x}-0} \leq \frac{P}{Q}$
Result	$x = \bar{x} - \frac{P}{Q}(255-\bar{x})$	$x = \left(1+\frac{Q}{P}\right)\bar{x}$	$x = \left(1+\frac{P}{Q}\right)\bar{x}$	$x = \bar{x} - \frac{Q}{P}(255-\bar{x})$

Figure 6 shows an example of the above concentric square pattern, where a block with 11×11 pixels shown in Figure 6(a) is extracted from the image in Figure 7(a), and the mean color is shown in Figure 7(b). Figure 6(c) shows the obtained pattern of concentric squares.

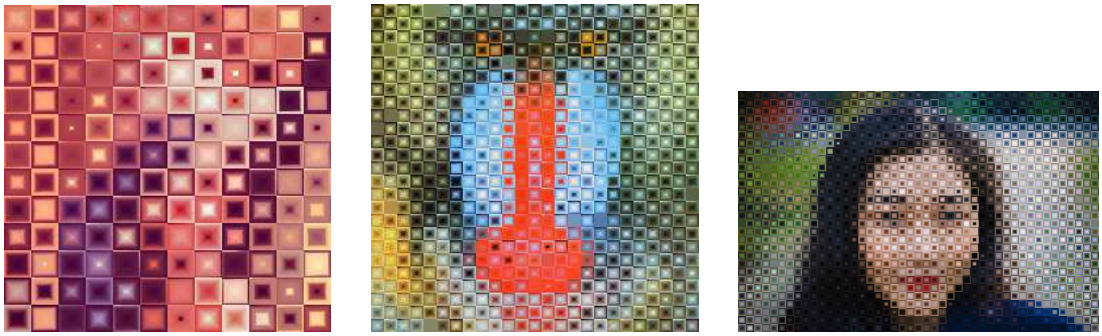
(a) Extracted block (b) Mean color (c) Concentric squares

FIGURE 6. Example of concentric squares generated from a block in Figure 7(a)

3. **Experimental Results.** In this section, we show the results of the proposed image pixelization with real images. First, we convert three images in Figure 7 into checkered patterns as shown in Figure 8. The size of input images in Figures 7(a) and 7(b) is 256×256 pixels, to which we set the parameter for block size as $h = 20$ and $h = 10$, respectively. The larger value of h makes each block more noticeable, and image contents obscurer, and vice versa. Figure 7(c) shows an example of larger image of 618×900 pixels, which is converted into the checkered image in Figure 8(c) for $h = 20$. Changing the order of pixels based on the intensity values between descending and ascending orders, we can visually enhance the checker pattern.

(a) 256×256 (b) 256×256 (c) 618×900

FIGURE 7. Input images

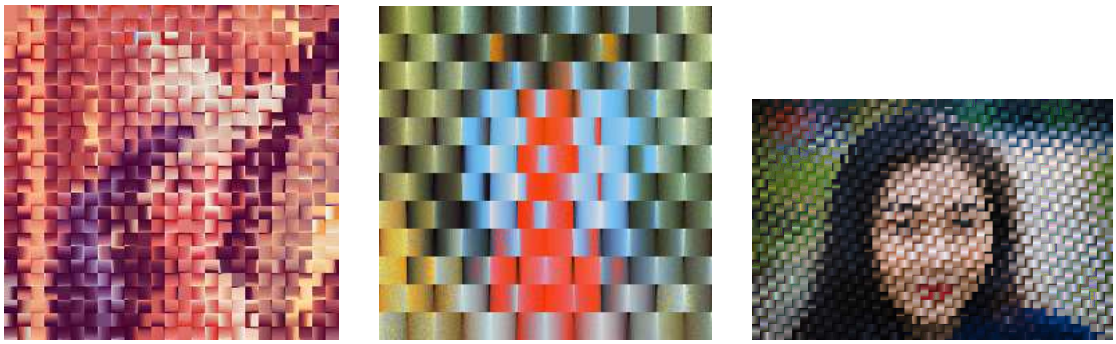


(a) $h = 20$ (b) $h = 10$ (c) $h = 20$

FIGURE 8. Checkered patterns

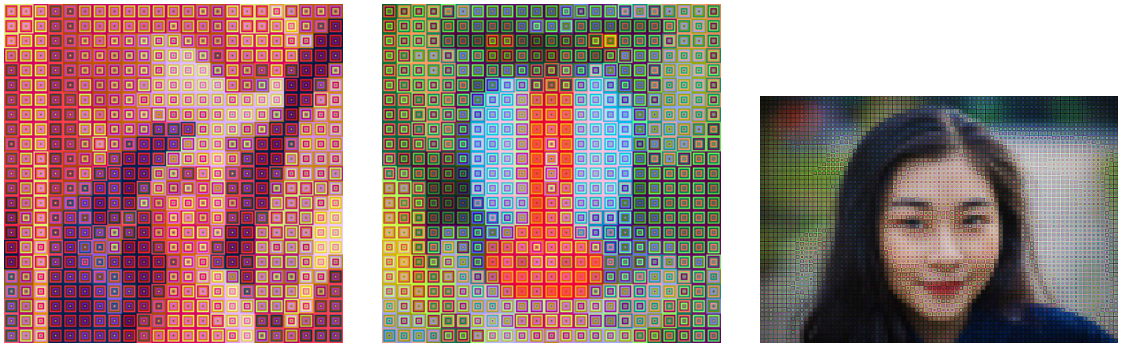
Next, Figure 9 shows the results of the second decorative image pixelization with woven patterns generated from the images in Figure 7. Figure 9(a) shows the result of image pixelization with plain weave pattern generated from the input image in Figure 7(a) for $h = 10$, where we can see a vertically and horizontally-woven pattern in a low resolution image of Figure 7(a). Figures 9(b) and 9(c) show the results with Dutch and twill styles, respectively. Although these results obtained with the same parameter $h = 20$, the densities of the woven patterns are different from each other since they have different image resolutions. The Dutch and twill styles in Figures 9(b) and 9(c) exhibit the bumpiness of the image surfaces more than the plain style in Figure 9(a) as well as their real woven products.

Figure 10 shows the results of the third image pixelization with concentric squares generated from the images in Figure 7. In this method, the size of a block is fixed to 11×11 pixels. Therefore, for smaller images, each concentric square pattern becomes more visible than larger images. However, it is straightforward to make the concentric square pattern visible in larger images by enlarging the block size to 22×22, 33×33 and so on.



(a) Plain ($h = 10$) (b) Dutch ($h = 20$) (c) Twill ($h = 20$)

FIGURE 9. Woven patterns



(a) $M \times N = 23 \times 23$ (b) $M \times N = 23 \times 23$ (c) $M \times N = 56 \times 81$

FIGURE 10. Concentric squares

4. **Conclusions.** In this paper, we proposed a method for generating pixel art with decorative pixel patterns, which is an extension of the concept of image pixelization by decorating each pixel of a pixelized image to various styles or primitive square patterns. Specifically, we demonstrated three styles of decorative pixels with checkered, woven and concentric squares patterns in this paper. To make the checkered patterns visually distinctive, we enhanced the colors in each block, which expresses a pixel in generated pixel art, by expanding the color distribution in an RGB color cube. Similar color enhancement technique is also used in the style of concentric squares. For woven patterns, we applied three weaving patterns: plain, Dutch and twill weavings. The proposed method for making weaving patterns is also applicable to other weaving patterns expressed on any square grids.

Our future work will include the generation of "Tetris"-like pixel art, in which adjoining pixels constitute tetrominoes colored in a limited number of colors.

Acknowledgment. This work was supported by JSPS KAKENHI Grant Numbers JP19 K12664 and JP21K11964.

REFERENCES

[1] A. Goldberg and R. Flegal, ACM president's letter: Pixel art, *Communications of the ACM*, vol.25, no.12, pp.861-862, doi: 10.1145/358728.358731, 1982.

[2] D. Yu, *Pixel Art Tutorial*, 2020, https://www.derekyu.com/makegames/pixelart.html, Accessed on 21-April-2022.

[3] C. Dwork, F. McSherry, K. Nissim and A. Smith, Calibrating noise to sensitivity in private data analysis, in *Theory of Cryptography. TCC 2006. Lecture Notes in Computer Science*, S. Halevi and T. Rabin (eds.), Springer, Berlin, Heidelberg, doi: 10.1007/11681878_14, 2006.

[4] L. Fan, Image pixelization with differential privacy, in *Data and Applications Security and Privacy XXXII. DBSec 2018. Lecture Notes in Computer Science*, F. Kerschbaum and S. Paraboschi (eds.), Springer, Cham, doi: 10.1007/978-3-319-95729-6_10, 2018.

[5] C. Tomasi and R. Manduchi, Bilateral filtering for gray and color images, *The 6th International Conference on Computer Vision*, Bombay, India, pp.839-846, doi: 10.1109/ICCV.1998.710815, 1998.

[6] D. Comaniciu and P. Meer, Mean shift: A robust approach toward feature space analysis, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.24, no.5, pp.603-619, doi: 10.1109/34.1000236, 2002.

[7] J. Kopf, A. Shamir and P. Peers, Content-adaptive image downscaling, *ACM Transactions on Graphics*, vol.32, no.6, doi: 10.1145/2508363.2508370, 2013.

[8] A. C. Öztireli and M. Gross, Perceptually based downscaling of images, *ACM Transactions on Graphics*, vol.34, no.4, pp.1-10, doi: 10.1145/2766891, 2015.

[9] Z. Wang, A. C. Bovik, H. R. Sheikh and E. P. Simoncelli, Image quality assessment: From error visibility to structural similarity, *IEEE Transactions on Image Processing*, vol.13, no.4, pp.600-612, doi: 10.1109/TIP.2003.819861, 2004.

[10] T. Gerstner, D. DeCarlo, M. Alexa, A. Finkelstein, Y. Gingold and A. Nealen, Pixelated image abstraction, *Proc. of the 10th International Symposium on Non-Photorealistic Animation and Rendering (NPAR2012)*, doi: 10.5555/2330147.2330154, 2012.

[11] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua and S. Süsstrunk, Slic superpixels compared to state-of-the-art superpixel methods, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.34, no.11, pp.2274-2282, doi: 10.1109/TPAMI.2012.120, 2012.

[12] K. Rose, Deterministic annealing for clustering, compression, classification, regression, and related optimization problems, *Proceedings of the IEEE*, vol.86, no.11, pp.2210-2239, doi: 10.1109/5.726788, 1998.

[13] T. C. Inglis and C. S. Kaplan, Pixelating vector line art, in *International Symposium on Non-Photorealistic Animation and Rendering*, P. Asente and C. Grimm (eds.), The Eurographics Association, doi: 10.2312/PE/NPAR/NPAR12/021-028, 2012.

[14] J. E. Bresenham, Algorithm for computer control of a digital plotter, *IBM Systems Journal*, vol.4, no.1, pp.25-30, doi: 10.1147/sj.41.0025, 1965.

[15] J. Kopf and D. Lischinski, Depixelizing pixel art, *ACM SIGGRAPH 2011*, vol.30, no.4, pp.1-8, doi: 10.1145/2010324.1964994, 2011.

[16] C. Han, Q. Wen, S. He, Q. Zhu, Y. Tan, G. Han et al., Deep unsupervised pixelization, *ACM Transactions on Graphics*, vol.37, no.6, pp.1-11, doi: 10.1145/3272127.3275082, 2018.