



An art-oriented pixelation method for cartoon images

Peng Lei¹ · Shuchang Xu^{2,3} · Sanyuan Zhang¹

Accepted: 22 December 2022 / Published online: 7 January 2023
© The Author(s), under exclusive licence to Springer-Verlag GmbH Germany, part of Springer Nature 2023

Abstract

Pixel art has evolved from a primitive computer image presentation to an independent digital art style. It is widely used on the internet, for graphic user interface (GUI) design, and game industries. Existing pixelation tools and algorithms generate pixel images with artifacts, color clutter, blurring, and a lack of aesthetics. Generally, aesthetics are the dominant concern for pixel art. In this paper, an art-oriented pixelation (AOP) algorithm is proposed to effectively retain the main features of the original image content and the integrity of essential details with the artistic and aesthetic styles. At the same time, the AOP algorithm enables high-quality pixel image generation of arbitrary size without paired datasets and model training effort. The experimental results demonstrate that the pixel image generated by the AOP algorithm outperforms existing algorithms and tools in terms of aesthetics.

Keywords Pixelation · Pixel art · Image processing · AOP algorithms

1 Introduction

Pixel iconography is an image drawn in pixels, also known as pixel art. It first appeared in computer applications (icon) and early 8-bit video games. Since the 21st century, it has been widely used on the internet, for GUI (graphical user interface) design, and in games. Pixel art does not refer to a single image. It emphasizes a clear outline, a bright palette, and an unfettered style. A pixel contains two features: square shape and single color [1]. In addition, the artist proposed a primary criterion—that a pixel image is a creative image drawn from regular and non-side-by-side overlapping pixels (as shown in Fig. 1)—to regulate the scope of pixel art.

Due to the restriction of pixel image size, artists tend to highly summarize the painting content, which leads to

a strong personal style and subjective consciousness in the artwork. However, there are rules for the artist when creating pixel images (Fig. 2). Each line follows the geometric proportion and increases or decreases step by step. Therefore, we can use mathematic modeling to generate the pixel image. In addition, the artist generalizes the colors, mainly by preserving the dominant colors and the luminance relationships of the original image content, which leads to fewer colors in the final pixel image.

The downsampling methods and neural network-based methods generate pixel images with artifacts, blurring, color clutter, and irregular pixel arrangements. Neural network-based methods are highly dependent on paired training data and generate pixel images of a fixed size. The existing algorithms extract colors by applying clustering methods, such as *k*-means [2]. The clustering approaches narrow thousands of colors down to a dozen or fewer in an image without affecting its visual appearance. However, the clustering method has difficulty in accurately estimating the required number of colored regions. The variation in the cluster amount affects the final presentation.

This paper proposes an art-oriented pixelation algorithm (AOP), which can be considered a multistep iterative process. Each iteration involves operations for determining the content, clustering coefficient, and primary color. The whole process refers to the creation process of pixel artists. The main contributions of our work are as follows:

✉ Shuchang Xu
xusc@hznu.edu.cn

Peng Lei
leipeng@zju.edu.cn

Sanyuan Zhang
syzhang@cs.zju.edu.cn

¹ School of Computer Science and Technology, Zhejiang University, Hangzhou, China

² School of Information Science and Engineering, Hangzhou Normal University, Hangzhou, China

³ Institute of Big data and Information Technology, Wenzhou University, Wenzhou, China

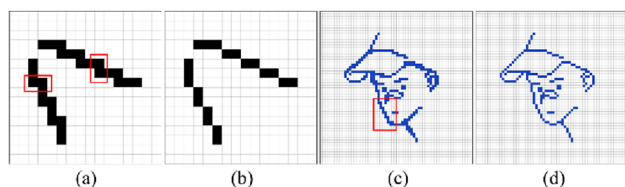


Fig. 1 **a** Pixels are side-by-side overlapping, **b** pixels are non-side-by-side overlapping, **c** pixels are irregular, and **d** pixels are regular and non-side-by-side overlapping

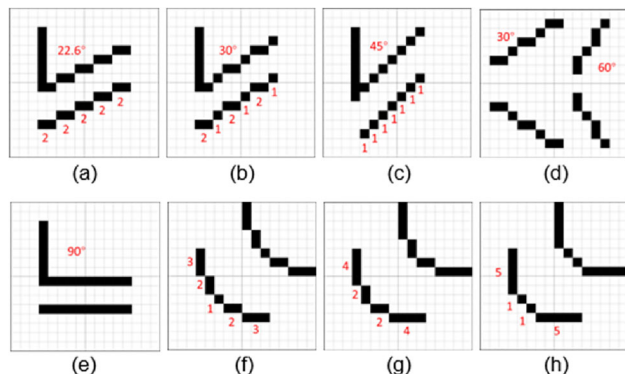


Fig. 2 Smooth pixel lines acceptable to human vision [1]: **a** 22.6° diagonal (double dotted line), **b** 30° oblique line, **c** 45° oblique line, **d** 60° oblique line, **e** straight line, **f** arc arranged in 3-2-1-2-3, **g** arc arranged in 4-2-2-4, and **h** arc arranged in 5-1-1-5

- (1) AOP can process arbitrary cartoon images and generate pixel images that conform to pixel art creation's logical rules and aesthetics.
- (2) AOP retains the original image's content intact, such as the basic shape, complete outline, and color, reducing the occurrence of image artifacts, color clutter, and other problems and ensuring that the image content is legible.
- (3) AOP can generate pixel images of arbitrary size and output high-quality pixel images without relying on paired datasets and spending much time in training models according to practical requirements.

Typical pixel art is mostly two-dimensional images, so our proposed AOP algorithm is mainly applied to process arbitrary two-dimensional cartoon images (as shown in Fig. 3a). The images used in the AOP algorithm test experiments are two-dimensional cartoon images of all sizes at 1080*1080 provided by Beijing Dailybread Co., Ltd. It contains simple cartoon images, cartoon images without background, and cartoon images with texture.

2 Related works

Pixel images are typically created by artists or specified software, such as the online generation tools: PixelMe [3] and Pixelator [4]. There are two different ways to generate pixel

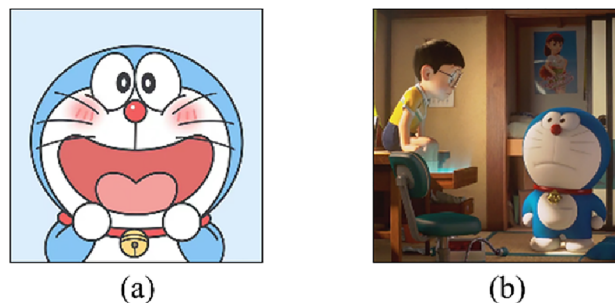


Fig. 3 **a** Two-dimensional cartoon image, **b** three-dimensional cartoon image. Images are public domain

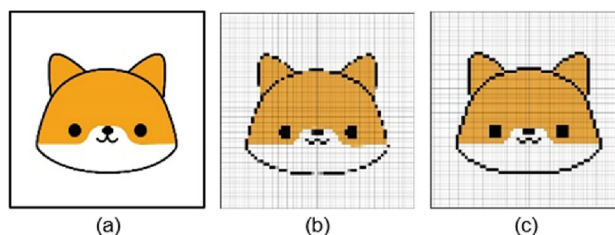


Fig. 4 Artist pixel image creation process: **a** original image, **b** image after processing by "pixelation" function, and **c** final pixel image

images: image downsampling and the neural network-based image-to-image approach. However, the pixel images generated by the existing methods usually have artifacts, color clutter, pixels arranged irregularly, and a lack of aesthetics.

2.1 Artist's way [5]

Artists can manually create pixel art. Alternatively, they can be created by exclusive software, for example Photoshop. They set the pixel block sizes (pixel block consisting of $n*n$ pixels) using the "pixelate" function to initially pixelate the image (Fig. 4b) and optimize the pixel image manually. Finally, the artists can obtain the corresponding pixel image (Fig. 4c). The other method is that the artist uses Fig. 4a as a reference to create a new canvas and draw a pixel image (Fig. 4c).

2.2 Autonomous pixelation tools

PixelME [3] is an online website (shown in Fig. 5) where users can generate pixel images by uploading original images. Users can choose four output sizes from PixelMe. Pixelator [4] is a pixelation tool that requires the user to use the functions to process the original image. These functions include "Pixelation," "Colors and Palette," and "Smoothing Filter." Therefore, only users who have sufficient knowledge about pixel art can use Pixelator well.

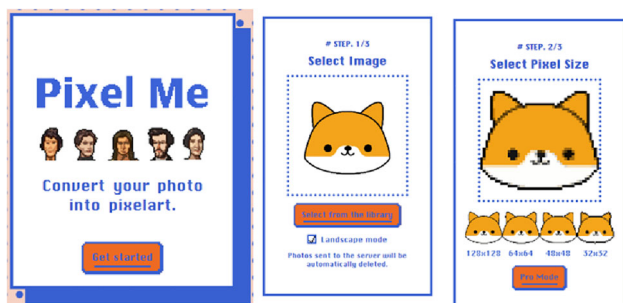


Fig. 5 PixelMe pixelation process

PixelMe and Pixelator can pixelate images quickly, but the processing brings many artifacts, color clutter, irregular pixel alignment, and a lack of aesthetics.

2.3 Automatic generation algorithm

The image downsampling methods are similar to the art creation methods. The image is first divided into multiple pixel blocks of the same size, and then, each pixel block is sampled and filled. Popular downsampling methods include bilinear [6] and bicubic [7] methods. The image downsampling methods actually cannot generate high-quality pixel images, considering that the generated pixel images usually have image artifacts, image blurring, and color clutter. Shang [8] proposed an automatic pixelation algorithm based on simple linear iterative clustering (SLIC) and fuzzy iterative self-organizing data analysis (FISODATA) algorithms to solve the above problems. However, the generated results still cannot be considered a pixel image.

In recent years, neural networks have become popular methods to perform image-to-image transformation. These include CycleGAN [9], DualGAN [10], DiscoGAN [11], and deep unsupervised pixelization [12]. The neural network approaches are highly dependent on paired training data. However, paired data are challenging to collect and prepare pixel art data. Meanwhile, because the GAN-based approach is uncontrolled, the pixel images generated by neural networks do not meet the basic aesthetic requirements of pixel art, such as a single-pixel width and regular arrangement.

3 Method

The AOP algorithm is an iterative process, as shown in Fig. 6. The processing is as follows: the original image is first gridded (Fig. 6a), followed by image content extraction (Fig. 6b). Then, the AOP algorithm starts iterating. The iteration processing includes pixelating the contour part (Fig. 6c) and the non-contour part (Fig. 6d) of the original image. In each iteration, operations such as extraction and filtering primary

colors, pixel block filling, and matching output positions are successively performed. Finally, the AOP algorithm generates the pixel image.

The variables used in the following section are illustrated in Fig. 7. As shown in Fig. 7a, the variables involved are the original image's dimensions, the width is W_{input} , and the height is H_{input} . As shown in Fig. 7b, the variables involved are the pixel image dimensions, the width is W_{output} , and the height is H_{output} . Figure 7c shows the image after preprocessing (Fig. 6a), and the variables involved are the preprocessed image dimensions. The preprocessed image width is $W_{reimage}$, and the height is $H_{reimage}$. In addition, the main outline of the image content is determined by the line width W_{line} . Figure 7d shows the image after preprocessing (Fig. 6b). The variable involved is the image content in judgment parameter N_{pixel} . The pixel block's (as shown in Fig. 7e) width and height are G_{pixel} , and the diagonal is $G_{diagonal}$.

The AOP algorithm is summarized as shown in Algorithm 1.

Algorithm 1 AOP algorithm

Input original image (W_{input}, H_{input}), pixel image sizes (W_{output}, H_{output})

Output pixel image

```

1:  $G_{pixel} = W_{input}/W_{output} = H_{input}/H_{output}$  according to Eq. (1)
2: if  $G_{pixel}$  is not integer then
3:    $G_{pixel} = \text{round}(G_{pixel})$ 
4:    $G_{diagonal} \leftarrow G_{pixel}$ 
5:   reimage (pixel blocks),  $W_{reimage}, H_{reimage} \leftarrow$  original image,  $G_{pixel}$ 
6:    $W_{line} \leftarrow$  the reimage is calculated according to Eq. (4)
7:   reimage is used to generate pixel image
8: else then
9:    $W_{line} \leftarrow$  the original image is calculated according to Eq. (4)
10:  original image (pixel blocks) is used to generate pixel image
11: end if
12: for pixel blocks do
13:  if  $W_{line} < G_{pixel}$  then
14:    Use pixelation scheme (Table 1(1)) to generate pixel image
15:  elseif  $W_{line} > G_{diagonal}$  then
16:    Use pixelation scheme (Table 1(2)) to generate pixel image
17:  else  $G_{pixel} \cong W_{line} \cong G_{diagonal}$  then
18:    Use pixelation scheme (Table 1(3)) to generate pixel image
19:  end if
20:  pixel art aesthetic optimization
21: end for
22: return pixel image

```

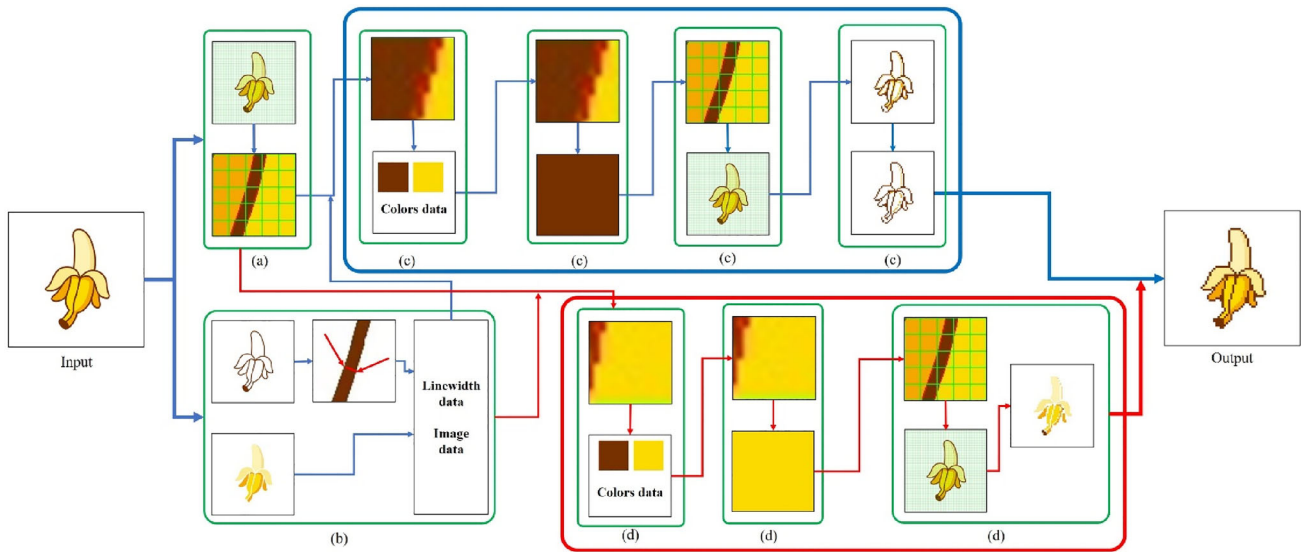


Fig. 6 AOP algorithm process. The original image is first gridded (a), followed by image content extraction (b). Then, algorithm starts iterating. Iteration is divided into two parts: c pixelation of the contour part

of the cartoon image content and d pixelation of the non-contour part. Finally, the AOP algorithm generates the pixel image

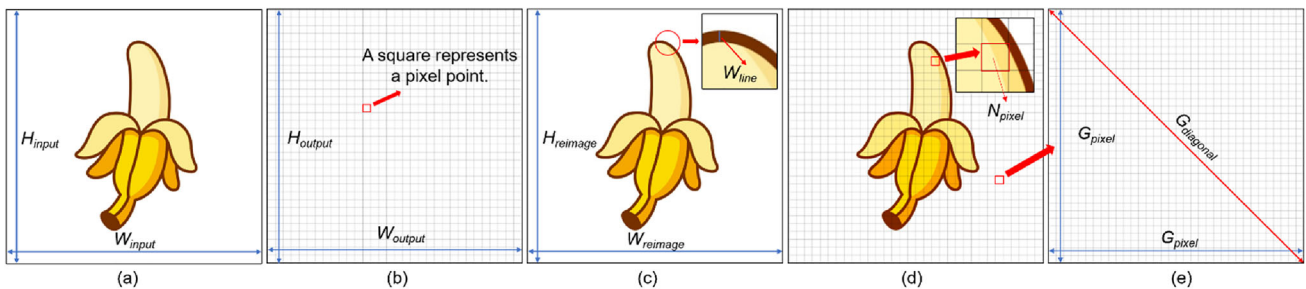


Fig. 7 Variable parameter diagram in the AOP algorithm

3.1 Preprocessing

We divide the original image to obtain multiple pixel blocks based on the size of the input and output. Moreover, we use line characteristics to divide the image’s content and use the luminance relationships to determine the primary color.

3.1.1 Dividing pixel blocks

The width and height of the original image are represented by W_{input} and H_{input} , respectively. The pixel image width is W_{output} , and the height is H_{output} . The pixel block is square, and its width and height are G_{pixel} . Their relationship can be represented by Eq. (1):

$$G_{pixel} = W_{input}/W_{output} = H_{input}/H_{output} \tag{1}$$

To reduce errors in the subsequent pixelation process, it is essential to avoid substantial changes to the dimensions of

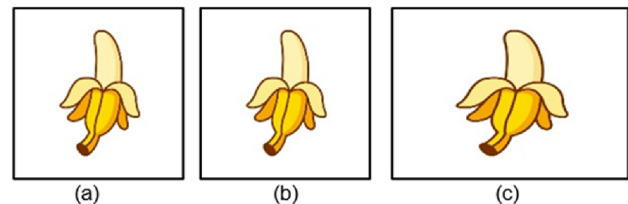


Fig. 8 a Original image of square, b pixel image of square, and c pixel image of rectangle

the original image as much as possible. As shown in Fig. 8, the original image (Fig. 8a) is square. The original image will not be distorted if the output is a square image (Fig. 8b), but the original image will be severely distorted if the output is a rectangular image (Fig. 8c). Therefore, it is required that the value of W_{output}/H_{output} is equal to or approximates the value of W_{input}/H_{input} . This can be represented by Eqs. (2) and (3):

$$W_{output}/H_{output} \approx W_{input}/H_{input} \tag{2}$$

$$W_{output}/H_{output} = W_{input}/H_{input} \tag{3}$$

Since the value of G_{pixel} is not necessarily an integer, G_{pixel} is rounded off using the rounding method. That is, the original image is scaled up or down. The processed image's width is $W_{reimage}$, the height is $H_{reimage}$, and the relationship with G_{pixel} can be represented by Eq. (4):

$$W_{reimage} = G_{pixel} * W_{output}, \quad H_{reimage} = G_{pixel} * H_{output} \tag{4}$$

3.1.2 Marking of the main image content

With the complex and changeable nature of the image subject content, there is no objective law to determine the dominant content of the image directly. In art creation, painting techniques, such as color contrast and outlining the subject matter, are often used to highlight the differences between the subject matter and the content, i.e., the outline constitutes the image content. Therefore, we can mark the image's outline to provide data for subsequent processing.

We use the "findContours" function in OpenCV to find image contours from the binary image. Due to complex image content and potential image noise, the results from the primary method have a large gap with the actual edge of the image. Therefore, the primary method applies the Canny [13] edge detector. The main evaluation criteria of the Canny edge detection operator are a low error rate, high locality, and minimum response. This is a reasonable solution to the problems caused by the essential functions.

- (1) Low error rate: Identify as many actual edges as possible while minimizing noise-generated false alarms.
- (2) High locality: The marked edges should be as close as possible to the actual edges in the image.
- (3) Minimum response: The edge in the image can only be identified once, and the possible presence of noise in the image is not identified as an edge.

3.1.3 Division of image content

Overall, as shown in Fig. 9, the image can be decomposed into contour lines (Fig. 9b) and colored regions (Fig. 9c). In contrast, Fig. 9d shows composed of colored regions only.

The main contour line is the primary reference for dividing the content. When the line width value of the main contour line is immense or the pixel block value is small, the main contour line is treated as a colored region during the sampling process and pixelated according to the edge pixelation rules of the colored region. Therefore, we need to calculate

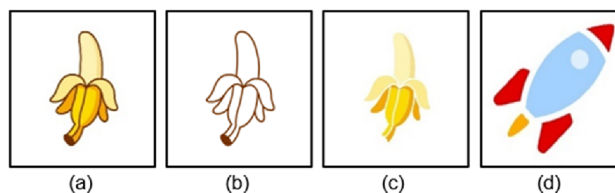


Fig. 9 **a** Original image, **b** contour lines of original image, **c** color regions of the original image, and **d** colored region-only image

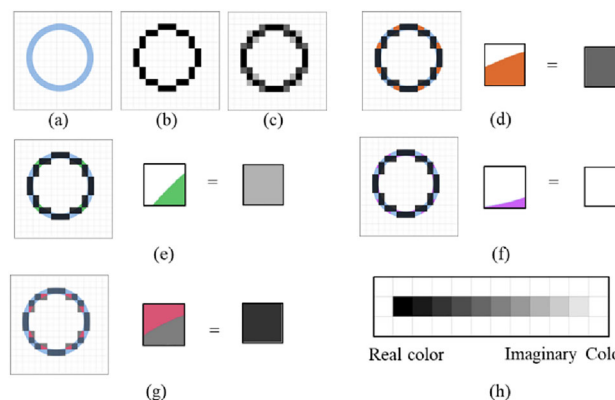


Fig. 10 **a** Original image, **b** pixelated circle, **c** manual anti-aliased circle, **d** pixel blocks with a large percentage of pixels, **e** pixel blocks with a small percentage of pixels, **f** pixel blocks with a tiny percentage of pixels, **g** pixel blocks in which the percentage of pixels does not reach 100%, and **h** Real color and Imaginary color

the main contour line's line width (W_{line}) as a judgment reference.

A dataset of the line-to-line distance of the detected edge lines is created, and the highest repeated value is taken as the line width of the main contour line. When the line width is equal to zero, it is determined that there is no main contour line; when the line width is not equal to zero, it is judged that there is a main contour line.

Calculation method: According to the coordinates of pixels (x_1, y_1) and (x_2, y_2) of different marker lines, the distance between marker lines can be calculated by applying the formula as follows:

$$W_{line} = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \tag{5}$$

3.2 Iterative process

3.2.1 Idea of sampling

The artist performs manual anti-aliasing [14] of the edges during the pixel image process to smooth the color transition of the edges of the graphic. As shown in Fig. 10, the circle (Fig. 10a) is pixelated to obtain the pixel circle (Fig. 10b). Strictly speaking, the pixel circle is not standard, so manual

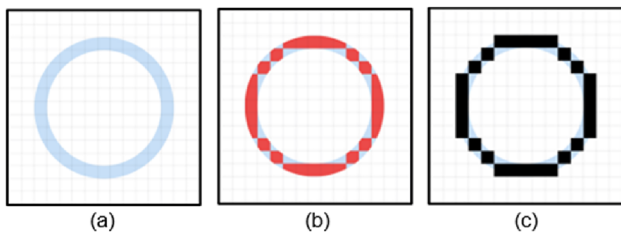


Fig. 11 **a** Original image, **b** part with high percentage of marked pixels, and **c** pixel image

anti-aliasing is needed to make it closer to a standard circle to obtain an artistic pixel circle (Fig. 10c).

Manual anti-aliasing is mainly performed on the unprocessed pixel blocks. The primary color is chosen according to the pixel percentage. The pixel blocks with more pixels are selected to be filled with a real color (Fig. 10d). Pixel blocks with a smaller percentage of pixels are filled with an imaginary color (Fig. 10e). Pixel blocks with a tiny percentage of pixels are ignored or filled with the background color (Fig. 10f).

In addition, the artist needs to refer to the percentage of pixels to reprocess the processed pixel blocks and selects the color based on the luminance relationships to make smooth color transitions. As shown in Fig. 10h, the color with strong contrasts compared to the background color (or the reference color) is the real color, while the imaginary color is the opposite.

Based on the manual anti-aliasing principle, we fill the part of the pixel block with more pixels (greater than 50%) with real color and ignore the part with fewer pixels (fewer than 50%) (as shown in Fig. 11). The visual appearance is closer to a standard circle.

3.2.2 Contour line handling scheme

We decide whether the image has a main contour line based on the line width (W_{line}) and the pixel block size (G_{pixel}). There are two schemes for handling the main contour line: if $W_{\text{line}} < G_{\text{pixel}}$, then, the main contour line is processed as a line scheme; if $W_{\text{line}} > G_{\text{pixel}}$, then, the main contour line is processed as a colored region scheme.

However, in the colored region scheme, it is possible that although $W_{\text{line}} > G_{\text{pixel}}$ (Fig. 12a), and the actual pixelation (Fig. 12b) is not according to the colored region scheme. As shown in Fig. 12c, the main contour line width is equal to the diagonal of the pixel block ($W_{\text{line}} = G_{\text{diagonal}}$). In the pixel block not completely covered by the contour line, the fill percentage of the contour line is precisely 50%, which is consistent with the sampled scheme. Therefore, the diagonal parameter of the pixel block (G_{diagonal}) is introduced to solve the above problems. The final processing scheme of the

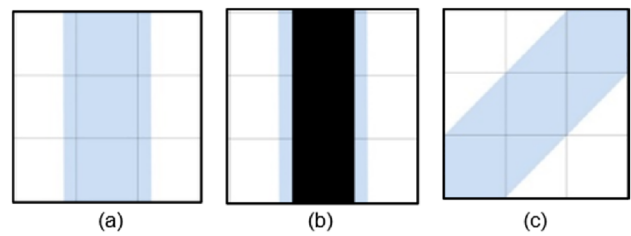


Fig. 12 **a** $W_{\text{line}} > G_{\text{pixel}}$, **b** pixelated processing **a**, and **c** $W_{\text{line}} = G_{\text{diagonal}}$

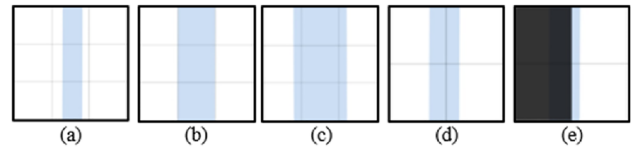


Fig. 13 **a** $W_{\text{line}} < G_{\text{pixel}}$, **b** $W_{\text{line}} = G_{\text{pixel}}$, **c** $W_{\text{line}} = G_{\text{diagonal}}$, **d** the main contour line is divided into two parts, and **e** pixelated **d**

main contour line is as follows: $W_{\text{line}} < G_{\text{pixel}}$, the main contour line is processed according to the line scheme; $G_{\text{pixel}} \leq W_{\text{line}} \leq G_{\text{diagonal}}$, the main contour line needs to be processed considering both line and colored region schemes; $W_{\text{line}} > G_{\text{diagonal}}$, the main contour line is processed according to the colored region scheme.

3.2.3 Pixelation processing parameter setting

Different processing schemes are proposed according to the three division cases mentioned above.

$W_{\text{line}} < G_{\text{pixel}}$ (Fig. 13a), where only the line scheme is considered, and the number of pixels in the pixel block (N_{pixel}) is calculated for the line width and the pixel block size; $W_{\text{line}} > G_{\text{diagonal}}$ (Fig. 13c), where only the colored region scheme is considered, and N_{pixel} is calculated with reference to the pixel block size only; $G_{\text{pixel}} \leq W_{\text{line}} \leq G_{\text{diagonal}}$ (Fig. 13b), where both the line and the colored region scheme are considered.

In addition, the pixel block division of an arbitrary image has a random nature. When the line is cut into two parts (Fig. 13d) or more, we process the pixel blocks according to the percentage of pixels (Fig. 13e), and the criticality of the percentage is 50%. Therefore, the final processing scheme is shown in Table 1.

3.2.4 Selection of pixel block fill color

Visually, the same color in an image may have different RGB values, and it is impossible to extract the color directly by calculating the RGB values. Therefore, we use the k-means method to extract the color for each pixel block to be processed. The value of k is determined by the situations in which the pixel block is divided. As shown in Fig. 14a, if the pixel

Table 1 Processing scheme

Conditions	Scheme of parameter setting
(1) $W_{line} < G_{pixel}$	$N_{pixel} > W_{line} * G_{pixel} * 1/2$
(2) $W_{line} > G_{diagonal}$	$N_{pixel} > G_{pixel} * G_{pixel} * 1/2$
(3) $G_{pixel} \leq W_{line} \leq G_{diagonal}$	$N_{pixel} > W_{line} * G_{pixel} * 1/2,$ and $N_{pixel} > G_{pixel} * G_{pixel} * 1/2$

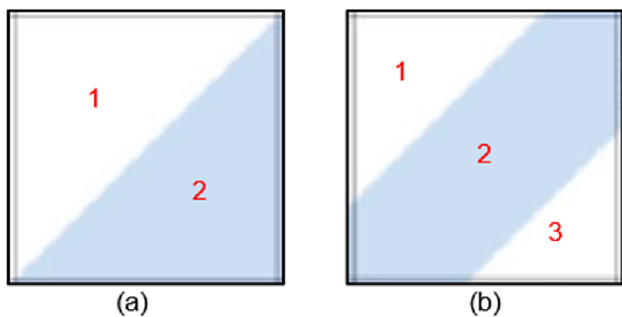


Fig. 14 Pixel block divided into **a** two parts and **b** three parts

block is divided into two parts, then $k = 2$; in Fig. 14b, if the pixel block is divided into three parts, then $k = 3$. Since the pixel block is small, to reduce interference, when the pixel block is divided into more than three parts, the value of k is taken as 3 in all cases.

Real color is used as the judgment color among the extracted colors. The V value of the HSV color space corresponding to the extracted color is used to determine the real color. The smaller the V value is, the darker the color and the closer to the accurate color.

3.2.5 Pixelated art aesthetic optimization processing

The complex randomness of the image content leads to deviations in the results regardless of the processing method used. To ensure the quality of the final pixelated artistic aesthetics, the pixelated lines need to be beautified in advance, as shown in Fig. 15a. The processing rule [15] is that the boundary of each line or colored region obeys the geometric scale, increasing or decreasing step by step, as shown in Fig. 15b.

3.2.6 Non-contour regions sampling processing

Non-contour regions sampling processing is mainly for the unprocessed pixel block. Most of the pixel blocks in this part are not lines or edges of colored regions in the image content, so the sampling and filling rule is that the color with the most pixels in the block is the filling color. The color extraction is still done by the k -means method. To reduce interference,

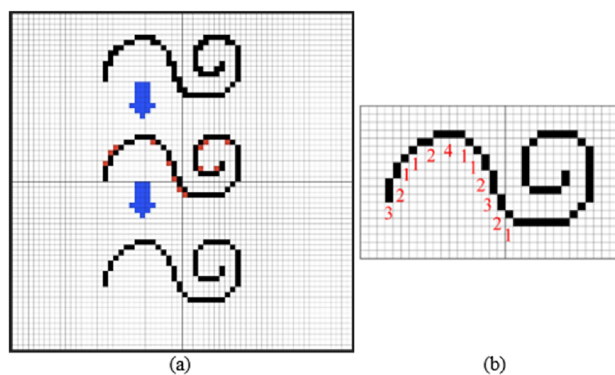


Fig. 15 Linear rule of pixel painting



Fig. 16 Experimental results. **a** Original image (from top to bottom, names are A, B, C, D, and E), and pixel images of size, **b** 32*32, **c** 48*48, and **d** 64*64

the smaller the k -value is, the better. However, the k -value cannot be taken as 1, which will lead to a considerable error in the extracted color; therefore, $k = 2$.

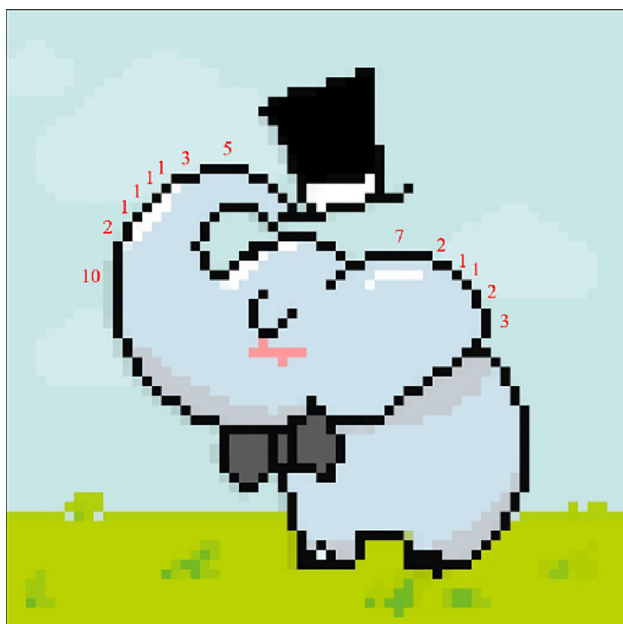


Fig. 17 Pixel image with output size of 48*48. Pixel arrangement of two random lines is 10-2-1-1-1-3-5 and 7-2-1-1-2-3

4 Experiment

4.1 Experimental results

Five cartoon images (two with background and three without background) are randomly selected and pixelated using the AOP algorithm. The images are named A, B, C, D, and E. The sizes of the generated pixel images are 32*32, 48*48, and 64*64, respectively (as shown in Fig. 16).

The experimental results show that the pixel image generated by the AOP algorithm can ensure the integrity of the original images regardless of the content, features, and colors. Meanwhile, the pixel images are clear and high-quality with few artifacts. We take the pixel image with an output size of 48*48 in image D as an example (as shown in Fig. 17); the pixel arrangements of the two random lines in Fig. 17 are 10-2-1-1-1-3-5 and 7-2-1-1-2-3. The pixel arrangement conforms to the law of exponential rise or fall, which is in line with the aesthetics of pixel art and has good pixel art performance.

In addition, the experimental results of image E show that the adaptive inductive processing of images by the AOP algorithm is not flexible enough, and there is a loss of image detail content (as shown in Fig. 18a). The image detail content processing is unreasonable and produces artifacts (as shown in Fig. 18b). The image processing is not stable enough, and unreasonable parts appear (as shown in Fig. 18c). These problems need to be solved in future work.

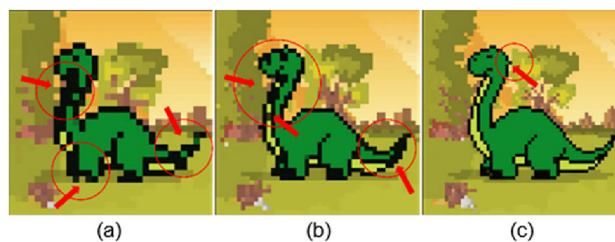


Fig. 18 Experimental results for E with output sizes of a 32*32, b 48*48, and c 64*64

4.2 Comparisons with existing algorithms

We compare the experimental results with existing pixelation algorithms in terms of image quality, contour integrity, and artistic aesthetics. The existing pixelation algorithms include deep unsupervised [12] and automatic portrait images [8]. In addition, the pixel images created by the artist are used as a reference for comparison. As shown in Fig. 19, the results of these methods are obtained from [8], and the pixel image size is 64*64. As shown in Fig. 20, we use the AOP algorithm to generate pixel images with sizes of 32*32 and 48*48.

The AOP algorithm well preserves the original image's content features in the case of different output sizes. The pixels arrangement conforms to the rules of exponential rise and fall, which have a better aesthetic perception, as shown in Fig. 21. Meanwhile, our algorithm can generate high-quality pixel images of arbitrary sizes without relying on any paired datasets and model training time.

We also compare with the latest CycleGAN method [20]. Since there is no publicly available code, we chose the original images from the paper to compare the results. As shown in Fig. 22, the AOP algorithm can stably generate pixel images with different sizes and good artistic quality. Compared to [20], AOP shows better results in detail and a more powerful ability to produce multiscale output.

4.3 Comparison with artistic creation and PixelMe

To verify that our algorithm has better performance, we compare our results with the pixelated images created by the artist and the online tools. PixelMe was chosen in the experiment since Pixelator requires many user interactions, which usually leads to entirely subjective results.

We selected five cartoon images (as shown in Fig. 16a) in the experiment. The generated pixel images have three sizes (32*32, 48*48, and 64*64). The pixel images are scored in the experiment.

As shown in Fig. 23, the cartoon image (a) with a size of 32*32 is used as the reference image. Fig. 23b–d shows the result created by the AOP algorithm, PixelMe, and artists. The participants are required to score each image out of 10.



Fig. 19 Comparisons of our results with those obtained by existing methods: **a** input, **b** subsampling method, **c** content-adaptive method [16], **d** perceptually based method [17], **e** image abstraction method [18, 19], **f** deep unsupervised pixelization [12], **g** automatic portrait

image pixelization [8], **h** our algorithm, and **i** artistic creation. The results of existing methods were obtained from Shang and Wong [8]. Images are public domain

The participants were randomly selected, and the whole scoring experiment was conducted without any instructions or intentional guidance. The final score was given by the average of each pixel image.

In addition, participants' knowledge of pixel art needed to be assessed in the experiment. A score of 10 means that the participant has enough knowledge about pixel art. A total of 25 participants were involved in this experiment. Fig. 24 shows that 60% of the participants' grades are lower than 4, and only 8% have grades higher than 8. This means that the participants generally knew nothing about pixel art and tended to rate the overall quality of the images, such as the clarity of the pixel image, integrity of the content, and aesthetics.

As shown in Table 2 and Fig. 25, the pixel image size is 32×32 , and the average scores of the AOP algorithm, PixelMe, and artists' creations are 5.784, 5.064, and 6.888, respectively. In the case of a larger size of 48×48 , the average scores of the AOP algorithm, PixelMe, and artists creation are 6.704, 5.56, and 7.36, respectively. In the case of a much larger size of 64×64 , the average scores of the AOP algorithm, PixelMe, and artists creation are 7.464, 6.04, and 7.712, respectively. The experimental results show that the AOP algorithm outperforms PixelMe.

In addition, the pixel images of image C (without background) and image E (with background) are taken as examples where the pixel image sizes are both 64×64 . As shown in Fig. 26, the pixel image generated by the AOP algorithm has a pixel arrangement of 5-3-2-1-1-1-1-2-9-3-1-1 (Fig. 26a), the pixel image generated by PixelMe has a pixel arrangement of 4-4-2-3-1-2-1-1-1-2-4-9-3-2-1 (Fig. 26b), and the pixel image generated by an artistic creation has a pixel arrangement of 5-3-2-1-1-1-1-1-2-10-2-1-1 (Fig. 26c). The pixel image generated by the AOP algorithm has a pixel arrangement of 1-1-2-1-1-1-2-4-3-1-1 (Fig. 26d), the pixel image generated by PixelMe has a pixel arrangement of 2-2-1-1-1-1-8-2-2-1-2-1 (Fig. 26e), and the pixel image generated by an artistic creation has a pixel arrangement of 2-2-1-1-2-5-2-1-1-1 (Fig. 26f). The pixel images generated by PixelMe are randomly arranged, and they do not conform to the step-by-step increasing or decreasing pixel arrangement pattern. The main content processing appears partially missing, and the background cannot be preserved. The pixel image generated by the AOP algorithm does not show many artifacts. The arrangement of pixels conforms to the arrangement law of pixel art. The main content and background can be handled and remain intact.

The AOP algorithm is currently superior to commonly used pixel image processing programs. The AOP score is



Fig. 20 Results of different output sizes generated by AOP algorithm: **a** original image, **b** pixel image of size 32*32, and **c** pixel image of size 48*48



Fig. 21 Pixel image of output size 48*48. Pixels arranged as 8-1-1-1-1-1-1-1-2-6-1-1-1-2-5

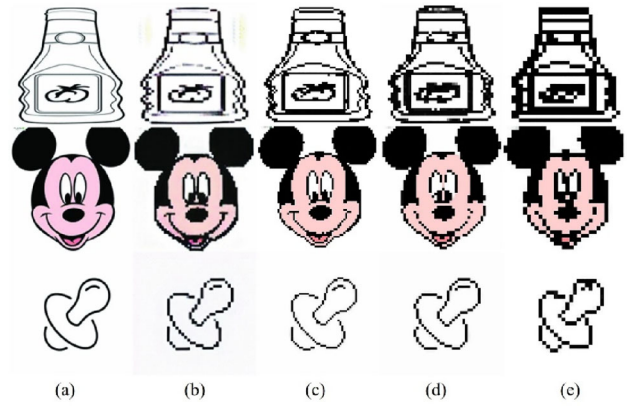


Fig. 22 Comparisons of our results with those obtained by the latest CycleGAN method: **a** input, **b** result of CycleGAN, **c** result of AOP, size 64*64, **d** result of AOP, size 48*48, and **e** result of AOP, size 32*32

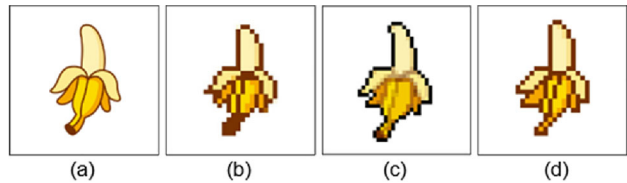


Fig. 23 Questionnaire for scoring experiments: **a** image A, **b** result of AOP, **c** result of PixelMe, and **d** result of artistic creation

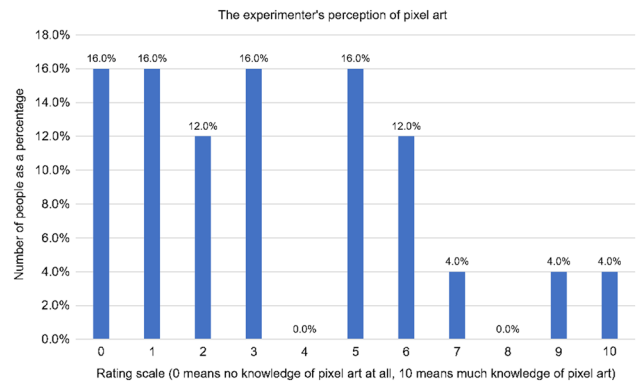


Fig. 24 Experimenter's perception of pixel art

lower than that of the artistic creation in the scoring experiments. However, in the pixel images processed by AOP, the arrangement of pixels step-by-step increases or decreases, which is consistent with the basic requirements of the artist drawing pixel lines and meets the aesthetic requirements of pixel image art.

5 Conclusion

Inspired by the whole process of pixel art creation, we propose an art-oriented pixelation algorithm in this paper to

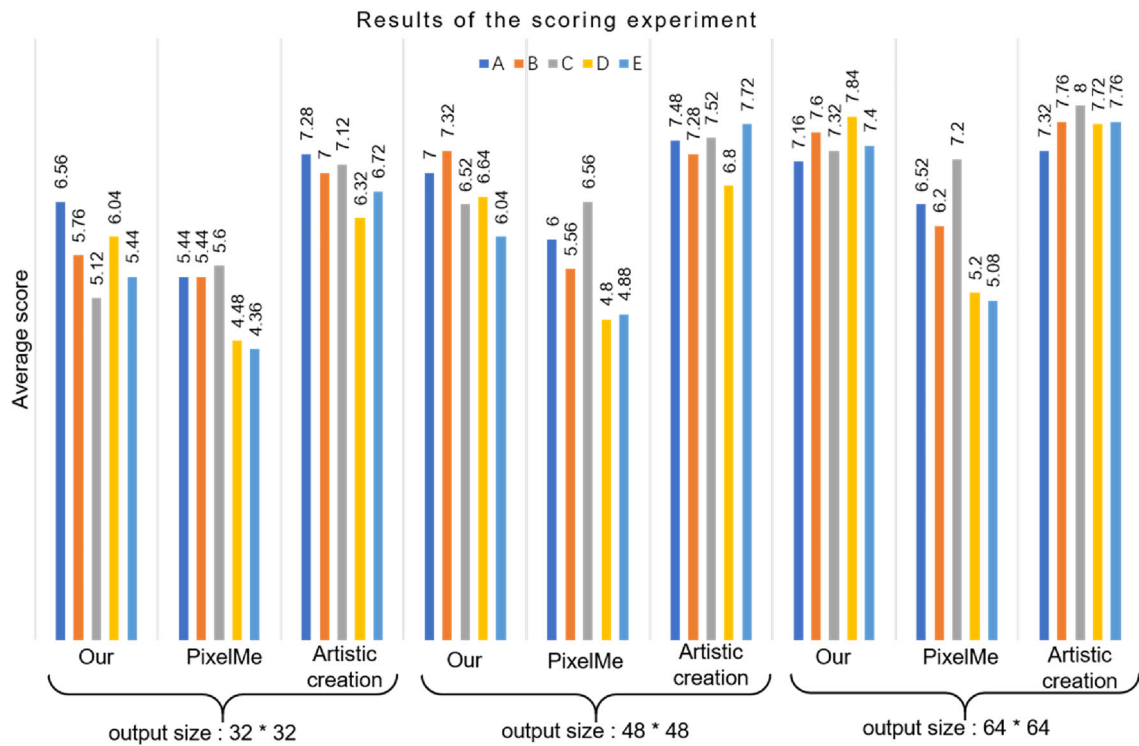


Fig. 25 Visual presentation of Table 2

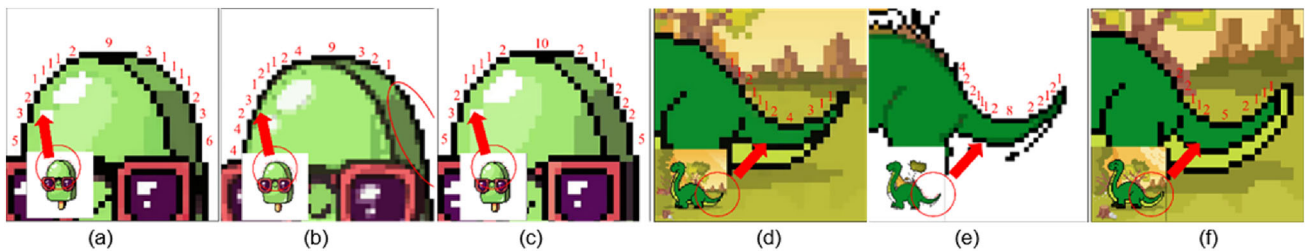


Fig. 26 Presentation of pixel arrangement of processing results. a AOP algorithm, b PixelMe, c artistic creation, d AOP algorithm, e PixelMe, and f artistic creation

Table 2 Results of scoring experiment

	Grade								
	Our	PixelMe	Artistic creation	Our	PixelMe	Artistic creation	Our	PixelMe	Artistic creation
Size of output	32*32			48*48			64*64		
A	6.56	5.44	7.28	7	6	7.48	7.16	6.52	7.32
B	5.76	5.44	7	7.32	5.56	7.28	7.6	6.2	7.76
C	5.12	5.6	7.12	6.52	6.56	7.52	7.32	7.2	8
D	6.04	4.48	6.32	6.64	4.8	6.8	7.84	5.2	7.72
E	5.44	4.36	6.72	6.04	4.88	7.72	7.4	5.08	7.76
Average score	5.784	5.064	6.888	6.704	5.56	7.36	7.464	6.04	7.712

convert cartoon images into pixel images with arbitrary sizes by multistep iterations. The pixel images generated by the AOP algorithm preserve the original image’s basic shape,

outline, and color. Moreover, the pixel images’ lines and colored regions conform to the logical rules and aesthetic requirements of pixel art creation. In addition, the generated

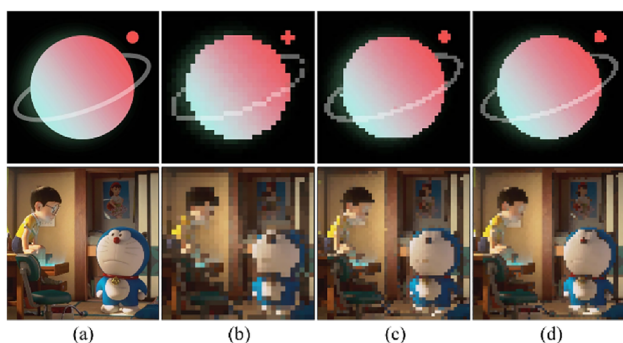


Fig. 27 Experimental results. **a** Original image, and pixel images of size, **b** 32*32, **c** 48*48, and **d** 64*64. Images are public domain

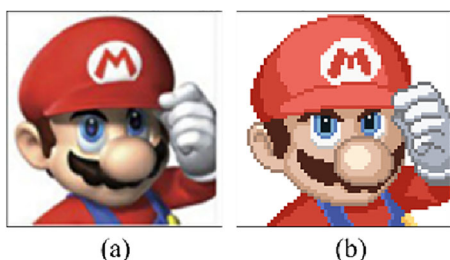


Fig. 28 **a** Original image, **b** artistic creation, size 64*64. Images are public domain

pixel images are clear and of high quality, without a large number of artifacts.

Although the AOP algorithm can generate exceptionally qualified pixel images, it fails in regard to the original images with more three-dimensionality, gradient colors, and rich details. As shown in Fig. 27, the experimental results in a mosaic image are not pixel art because it is challenging to accurately express the richness of the content in only one color. The artist simplified the original image by high generalization and luminance relations to create a three-dimensional picture into a two-dimensional pixel image, as shown in Fig. 28. Therefore, we are working on more complex cartoon images by simplifying images with various colors and textures.

Acknowledgments The work is supported by Beijing Dailybread Co., Ltd., and partly supported by the Soft Science Key Research Project of Zhejiang Province (No. 2022C25033).

Declarations

Conflict of interest The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Qian et al.: Pixel Design-Designers Talk about Pixel Art Creation. Electronic Industry Press (2004)

2. MacQueen, J., et al.: Some methods for classification and analysis of multivariate observations. In: Proceedings of the 5th Berkeley symposium on mathematical statistics and probability, pp. 281–297 (1967)
3. Kazuya Saito: PixelMe: Convert your photo into pixelart. <https://pixel-me.tokyo/en/> (2020). Accessed 10 Jan 2022
4. RonenNess: Pixelator. <http://pixelatorapp.com/index.html> (2021). Accessed 10 Jan 2022
5. Kabka007: Photoshop quickly converts illustrations to pixel-style. https://www.bilibili.com/video/BV1Tx411q7Wb?share_source=copy_web (2017). Accessed 10 Jan 2022
6. Allebach, J., Wong, P.W.: Edge-directed interpolation. In: Proceedings of 3rd IEEE International Conference on Image Processing, vol. 3. IEEE (1996)
7. Carlson, R.E., Fritsch, F.N.: Monotone piecewise bicubic interpolation. *SIAM J. Numer. Anal.* **22**(2), 386–400 (1985)
8. Shang, Y., Wong, H.-C.: Automatic portrait image pixelization. *Comput. Graph.* **95**, 47–59 (2021)
9. Zhu, J.-Y., et al.: Unpaired image-to-image translation using cycle-consistent adversarial networks. In: Proceedings of the IEEE International Conference on Computer Vision (2017)
10. Yi, Z., et al.: Dualgan: unsupervised dual learning for image-to-image translation. In: Proceedings of the IEEE International Conference on Computer Vision (2017)
11. Kim, T., et al.: Learning to discover cross-domain relations with generative adversarial networks. arXiv preprint [arXiv:1703.05192](https://arxiv.org/abs/1703.05192) (2017)
12. Han, C., et al.: Deep unsupervised pixelization. *ACM Trans. Graph. (TOG)* **37**(6), 1–11 (2018)
13. Mao, X.: Introduction to OpenCV3 Programming, pp. 248–249. Electronic Industry Press (2015)
14. One pixel: {Pixel painting teaching}- Modeling 03- manual anti-aliasing. https://www.bilibili.com/video/BV1CE411B7bD?share_source=copy_web (2019). Accessed 10 Jan 2022
15. Pedro Medeiros: Start drawing pixel paintings. <https://indienova.com/indie-game-development/saint11-making-pixel-art-1/> (2020). Accessed 10 Jan 2022
16. Kopf, J., Shamir, A., Peers, P.: Content-adaptive image downscaling. *ACM Trans. Graph.* **32**(6), 173:1–173:8 (2013)
17. Öztireli, A.C., Gross, M.: Perceptually based downscaling of images. *ACM Trans. Graph.* **34**(4), 1–10 (2015)
18. Gerstner, T., DeCarlo, D., Alexa, M., Finkelstein, A., Gingold, Y.I., Nealen, A.: Pixelated image abstraction. In: Expressive, pp. 29–36 (2012)
19. Gerstner, T., DeCarlo, D., Alexa, M., Finkelstein, A., Gingold, Y., Nealen, A.: Pixelated image abstraction with integrated user constraints. *Comput. Graph.* **37**(5), 333–347 (2013)
20. Kuang, H., Huang, N., Xu, S., Du, S.: A pixel image generation algorithm based on CycleGAN. In: 2021 IEEE 4th Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC), pp. 476–480 (2021). <https://doi.org/10.1109/IMCEC51613.2021.9482118>

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

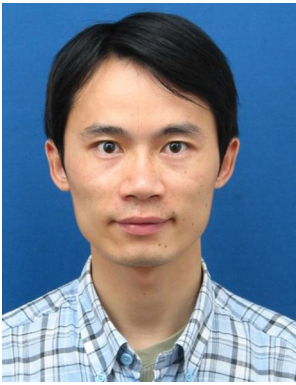
Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.



Peng Lei is a Master's student at Zhejiang University. He received his B.S degree in Chemical Engineering and Technology from Northwest Minzu University in 2010. His current research interests include image processing and computer vision.



Sanyuan Zhang is a professor at the College of Computer Science and Technology, Zhejiang University. He was a senior visiting researcher at the University of Hannover, Germany in 2004. His research interests include mathematical logic, computer mathematics, computer graphics, image and video analysis, and digital design methods.



Shuchang Xu Hangzhou, China. Birthday: September 1979. He received a Ph.D degree in computer science from the Department of Computer Science, Zhejiang University, China, in 2007. He is now an associate professor with the College of Information Science and Engineering, Hangzhou Normal University, China. From 2012 to 2013, he worked at Carnegie Mellon University as a visiting scholar. His current research

interests include image processing, artistic pattern generation, artificial intelligence, and image understanding.