

TIE-DYEING PATTERN FAST-GENERATION METHOD BASED ON DEEP-LEARNING AND DIGITAL-IMAGE-PROCESSING TECHNOLOGY

Suqiong Liu^{1,*}, Xiaogang Xing¹, Shanshan Wang¹, Jinxiong Zhou²

¹ Department of Clothing, School for the Art, Jinling Institute of Technology, Nanjing, Jiangsu, China

² Department of Mathematics, University of Liverpool, Liverpool, UK

*Corresponding author. E-mail: lsq@jit.edu.cn

Abstract:

Contingency and uniqueness are regarded as typical artistic characteristics. To accomplish the realistic effect of each tie-dyeing pattern artwork, we propose a digital tie-dyeing pattern fast-generation algorithm based on auxiliary-classifier deep-convolution generative adversarial network (AC-DCGAN) and image-processing technology. To apply this algorithm, the designer first draws the planar layout diagram of the tie-dyeing patterns. The diagram consists of a white background and polychrome circles, and the regional-connectivity algorithm is used to extract information on all the circle positions as well as the pattern categories in the diagram. Then the AC-DCGAN-generated background image is color-corrected to stitch and complete its construction. The AC-DCGAN-generated tie-dyeing pattern image is also color-corrected and is then segmented and copied to the circle area. Mean filtering creates the final digital tie-dyeing patterns. Results show no obvious color difference in generated patterns, splicing edges show uniform transition, and unique patterns exhibit tie-dye characteristics, achieving realistic artistic effects.

Keywords:

Generative adversarial-network, deep-learning, digital-image-processing technology, tie-dyeing pattern

1. Introduction

In the production of tie-dye-pattern art products, usually one of two conventional methods is used. One is the traditional tie-dyeing process, by which tying workers perform the physical dyeing by use of the fabric-defending treatment, completing the production through dip dyeing and the corresponding post-treatment processing. The other method is to use a digital scan or analog image that captures the tie-dye characteristics, generate the pattern by digital-synthesis conversion and other technologies, and then print the pattern by printing equipment (Gu, 2004; Liu et al., 2016). The benefit of using the first method is that each pattern is rendered by hand and is therefore very flexible and vivid, and all the patterns are different without repetition, each with its own special artistic characteristics of contingency and uniqueness. However, because the whole operation, including the dip-dyeing process, is manual, there are limitations in terms of production efficiency, large-scale batch operation, and fabric selection as well as environmental concerns. The second production method, which makes use of the advantages of digital technology, allows the patterns to be arbitrarily edited and repeatedly copied and printed, and thus this process has the benefits of minimal limitation for fabric selection, low pollution, low cost, and fast response, and it also can adapt well to personalized and mass production. However, a weak point is that the created patterns are rigid and inflexible, having lost the artistic soul's contingency and uniqueness that are characteristic of tie-dye art. Therefore, there was a need to work out a method that could generate complex tie-dyeing patterns by having the designer initially provide simple pattern outlines, thereby not only ensuring the artistic

characteristics of digital tie-dyeing patterns but also improving work efficiency.

Effective digital pattern-generation technologies, based on computer-aided methods, have used chaos or fractal theory and computer-graphics research to explore and prove mechanisms for the construction of intuitive and vivid visual images and have provided alternative approaches to discover new phenomena and study new patterns (Carter et al., 1998; Lu et al., 2005, 2017; Lv et al., 2014). Such study has also achieved abundant results in the design of textile patterns (mainly based on fractal models) and clothing. In pattern design, in general, the fundamental pattern unit should first be designed and then a limited number of iterations created, thus forming a larger unit-organization cycle. Tian et al. (2019) proposed a method for automatic generation of a batik floral pattern based on fractal geometry and achieved automatic simulation of traditional manually created batik patterns on a computer. Wang et al. (2019), by using the generation principle and graphic features of a complex dynamic system and L-system fractal graphics, developed two art graphics (floral and geometric), used Photoshop to create secondary designs from the generated art graphics, and applied them in the design of clothing patterns. Zhou (2004, 2007) put forward an automated batik-pattern-generation system based on fractal geometry, fulfilled the pattern design through the use of computer-aided design weaving software, and simulated weave patterns. Barnsley and Hurd (2000) used iteration of Julia set fractal-pattern functions to modify design methods for structural reorganization of fashion patterns. Together these applications reveal that most fractal graphics are individual patterns characterized by fine structure, pattern repetition, and brightness but



that lack naturalness in color. Normally, they cannot be used directly in pattern design, and thus it was apparent that a secondary manual design would be required. Because merely using a method based on complex mathematical models or graphics results in defects in the variety and effects of the generated tie-dye patterns, we aimed to derive an appropriate mathematical calculation for each pattern, set complex parameters, and finally configure appropriate textures. Therefore, mimicking the traditional digital-pattern-generation process, we set out to imitate the basic flower-shaped-pattern features obtained from the tie-dyeing process and then generate a natural and nonrepetitive tie-dye pattern based on deep-learning and digital-image technology, which we expected could solve the perceived problem of the loss of the soul and vividness in traditional tie-dyeing patterns, thereby developing a new direction for the design of textile patterns and derivative clothing.

Generative adversarial networks (GANs), which could generate absolutely realistic-looking samples by automatically learning the distribution of the real data, have become one of the favorite research directions in the deep-learning field in recent years (Goodfellow et al., 2014; Creswell et al., 2017). GANs have the most profound and diverse application in the image field and have been widely used, for example, in the generation of English art fonts (Azadi et al., 2018), human faces (Huang et al., 2017; Li et al., 2018; Yang et al., 2021), automatic-driving scenarios (Santana and Hotz, 2016), and synthetic-painting sketches (Chen and Hays, 2018). In terms of the evolution of the algorithm, to solve the problem of no limitation for the input of the GAN generation model, which could cause model corruption, when modeling the generation model and the discriminant model, Mirza and Osindevo (2014) added conditional variables to the models to limit their input, thereby proposing conditional GANs. Radford et al. (2015) introduced the convolutional neural-network structure into the GAN, made some detailed improvements to the original GAN structure and the training process, and proposed deep-convolutional generative adversarial network (DCGAN). Owing to its training stability and industry-earned value, the DCGAN model has been widely used. To achieve the controllable generation of images, Odena et al. (2017) proposed that for the input of the auxiliary-classifier generative adversarial networks (AC-GANs), in addition to ordinary noise z in the GAN structure, the category label should also be added to it, that discrimination would no longer be limited to determining the authenticity of the input data, and that the discriminator would analyze the data's category label.

Based on the above analysis, we have studied the construction of auxiliary-classifier deep-convolution generative adversarial-network (AC-DCGAN) to generate nonrepetitive tie-dye flower-pattern primitives and have synthesized the digital tie-dye patterns combined with color correction, stitching, and segmentation algorithms in image-processing technology.

This article is organized as follows: Section 2 is a detailed introduction to the digital tie-dyeing pattern-generation algorithm flow, including the extraction of the data from the planar layout of the tie-dye patterns, the construction of the AC-DCGAN model, the stitching of the background, and the flower graphs. Section 3 provides and analyzes the results of image-generation.

2. Digital tie-dyeing pattern-generation algorithm

In the traditional tie-dye-pattern production process, the tie-dyeing technicians first need to transform and mark the pattern using the language of technology, make the tie-dyeing process production chart indicating the center point where the fabric needs to be stitched by means of the dot-matrix layout, and then hand it over to the workers for stitching, dyeing, and other post-treatment processing. The dot-matrix layout of the tie-dyeing production process is a key step to obtain the tie-dye patterns.

The tie-dyeing pattern-generation algorithm creates the digital tie-dye patterns by combining digital-image-processing technology and AC-DCGAN (Figure 1). First, it simulates the tie-dyeing process production chart with the dot-matrix layout. Then, using the drawing software, the designer draws a tie-dye-pattern planar layout, which is composed of non-overlapping multicolor circles against a white background. The layout becomes the input to the algorithm. Then, the regional-connectivity algorithm is used to extract information on all the circles' center positions and pattern categories, after which AC-DCGAN is applied to generate the background image to fill the layout and copy key areas of the AC-DCGAN-generated tie-dye pattern to the areas of the circles. Finally, mean filtering is applied to the entire image to obtain the final digital tie-dye pattern.

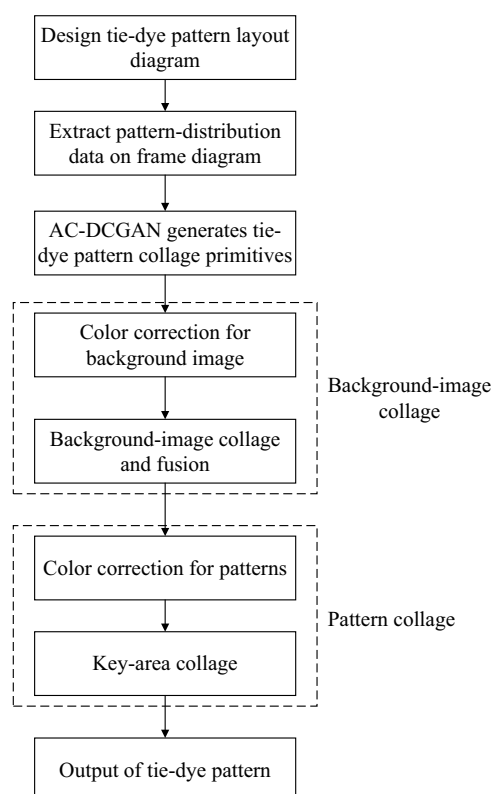


Figure 1. Digital tie-dyeing pattern-generation process.

2.1. Extraction of data from planar layout of tie-dyeing patterns

To make the creation of tie-dyeing patterns more flexible, the designer creates the planar layout of the floral graphs including circles in different colors representing different types of floral patterns that, when grouped together, form geometric patterns used to achieve the overall model of the pattern and to improve the efficiency of pattern design. A collaged tie-dye pattern spliced by a number of primitives within a rectangular area located on a rectangular-coordinate XY plane, which is described as a canvas, is shown in Figure 2. The colored circles represent the five types of floral patterns, and the planar layout contains information on the size of the canvas and on the types and position coordinates of the floral patterns. The primitives collaged on the canvas are divided into two types: floral patterns and background image. Additionally, the key areas of the floral patterns are filled with round areas, and the background image is filled with white areas in a zigzag pattern.

We defined a digital planar layout red–green–blue (RGB) three-dimensional image array P converted to a gray matrix P_g and specified the image size as $m \times n$. Then we extracted all the circles in the layout by using a four-neighborhood labeling algorithm and scanned the matrix P_g line by line. When center point C was not white, we judged C 's four neighboring positions (0,1,2,3), and if those four positions were not white, we considered C to have been connected with those four positions. Otherwise, we regarded C as an isolated point. The four-neighborhood labeling algorithm, used to label the layout (Figure 3), ensured that we obtained w regions and the coordinates $r(x_i, y_i)$ of the circle centers, for $i = 1, 2, \dots, w$. We set all points of the matrix P_g as (255,255,255).

2.2. AC-DCGAN generates collage primitives

GANs were originally intended for generation of data that do not exist in the real world, which is similar to enabling artificial intelligence to be creative or imaginative. Tie-dye images have profound

	0	
1	C	3
	2	

Figure 3. Four-neighborhood labeling.

features and contain complex texture information. GANs possess powerful image-generation ability to generate realistic tie-dye primitives, and, combined with color correction, image splicing, segmentation, filtering, and other digital-image-processing technologies, they can generate large-scale tie-dye patterns for use by designers.

2.2.1. GANs

GANs comprise two models – the generating model G and the discriminant model D . Through model G , random noise z generates the sample $G(z)$, which is subject to the real sample data distribution P_{data} to the maximum degree, and model D determines whether the input sample is real data x or generated data $G(z)$. To discern the distribution of data x , the generator first makes the prior noisy distribution $P_z(z)$ construct a mapping space $G(z; \mu_g)$, for which the corresponding discriminator mapping function is $D(x; \mu_d)$, and then it yields scalar output representing the probability that x is the real data. The optimization function of the generated model is as follows:

$$\min_G \max_D V(D, G) = E_{x \sim P_{\text{data}}(x)} [\log D(x)] + E_{z \sim P_z(z)} [\log (1 - D(G(z)))], \quad (1)$$

where x represents the real sample, $D(x)$ represents the probability that x is judged to be the real sample according to the discriminant network, z represents the noise of the input generated samples, $G(z)$ represents the samples generated by the noise z in the generation network, and $D(G(z))$ represents the probability that the generated samples are judged to be real

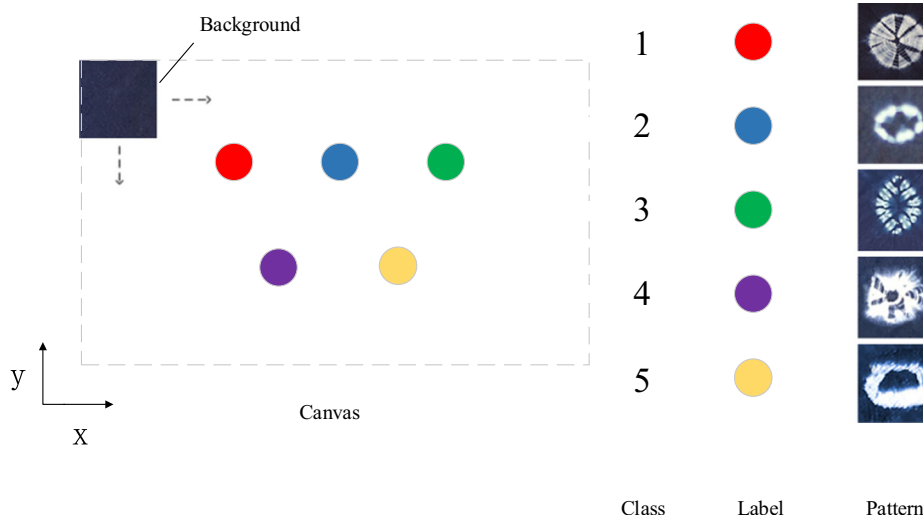


Figure 2. Example of planar layout of tie-dye pattern.

samples after passing through the discrimination network. The generation network is designed to make the generated samples as close to the real samples as possible, that is, it would be preferable when $D(G(z))$ is close to 1, as $V(D, G)$ would become smaller. The discrimination network is aimed at making $D(x)$ close to 1, and when $D(G(z))$ approaches 0, $V(D, G)$ will increase.

2.2.2. AC-DCGAN model

AC-DCGAN combines the advantages of both AC-GAN and DCGAN and introduces the *a priori* condition class c (on the basis of DCGAN) to guide the training. The discriminator D not only judges the authenticity of the input samples but also determines their type. As a result, it is possible to generate a specified category of patterns on demand, thus saving the time of manual classification. Different from GANs, the input of generator G in GANs only contains the random noise z , while in AC-GAN every generated sample has its corresponding label information, and the input of its generator G is from the label information c of the generated sample and the random noise z (i.e., the generated image $X_{\text{fake}} = G(c, z)$). The original GAN discriminator D is a dual-classifier that determines whether the input comes from real data or data generated by a generator, while the discriminator D of AC-GAN consists of both a dual-classifier (able to determine whether input comes from real or generated data) and a multiple-classifier (able to accurately classify the labels of both the generated and the real data). AC-DCGAN introduces the convolutional network into the structure of AC-GAN to replace the fully connected layer of the main network, and it directly uses the convolutional layer to connect the input and output layers of the generator and discriminator, while also improving the effectiveness of the network by using the powerful feature-extraction capability of the convolutional layer. The structure diagram of AC-DCGAN is shown in Figure 4. Thus, the objective function of the discriminator D is composed of a dual-classifier loss L_S and a multiple-classifier cross-entropy loss L_C :

$$L_S = E[\log P(S = \text{real} | X_{\text{real}})] + E[\log P(S = \text{fake} | X_{\text{fake}})], \quad (2)$$

$$L_C = E[\log P(C = c | X_{\text{real}})] + E[\log P(C = c | X_{\text{fake}})], \quad (3)$$

where L_S represents the loss of the accurately classified real samples and generated samples, and L_C represents the loss of the category of accurately classified samples. Accordingly, the optimization goal of the discriminator D of the AC-DCGAN is to maximize $L_S + L_C$, while the optimization goal of the generator G is to maximize $L_C - L_S$.

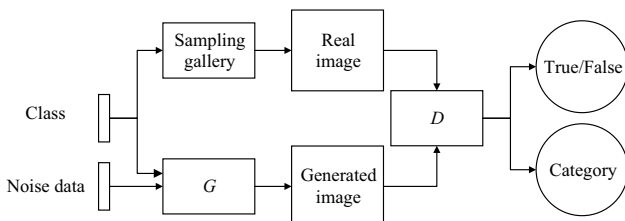


Figure 4. AC-DCGAN model structure diagram.

2.2.3. AC-DCGAN model structure

The generation network has six layers. The input z is a 100-dimensional random noise vector that is subject to normal distribution and connects z with the category label c vector (6 dimensions). After successively passing two linear full-connection layers, it transforms into vectors of 1,024 and $4 \times 4 \times 1,024$ in size, and then the vector of $4 \times 4 \times 1,024$ in size converts into a tensor of size $[1,024, 4, 4]$, thereby resulting in nonlinear rectified linear unit (ReLU) function transformation. After that, it would go through four deconvolution layers with a convolution kernel of size 5×5 and a step size of (2,2), and all the outputs would be activated by the nonlinear ReLU function to obtain a tensor of size $[3, 64, 64]$, and finally it would be activated with the use of the tanh function to output the generator image.

The input of discriminator D , an image with dimensions of $[3, 64, 64]$, would go through five convolution layers of size 5×5 and a step size of 2×2 , and leaky ReLU function transformation must be performed for the output of every layer to obtain a tensor of size $[1,024, 4, 4]$ and to finally reshape into a feature vector of size $4 \times 4 \times 1,024$; after that it will successively pass through two full-connection layers and yield output as a vector of size $4 \times 4 \times 1,024$, and then it would respectively connect with two full-connection layers of size 256 to finally produce two network outputs: the true/false probability of the image and the class label of the image.

2.3. Background image collage

The images generated by AC-DCGAN are of a fixed and limited size, but the size of the designed layout diagrams is larger and can change flexibly, so the background of the canvas must be collaged by a large number of small background images. In combination with color correction and collage-fusion treatment, a complete background image can be generated with uniform color and without edge segmentation.

2.3.1. Color correction for background image

The images generated by AC-DCGAN are inconsistent in their brightness and color, but the color correction algorithm could handle such inconsistency successfully during image stitching (Li et al., 2016; Tian et al., 2016). This algorithm starts locally by performing color correction for a single image that is to be spliced, and it finally obtains a panoramic image with consistent brightness and color. The key to color correction is to calculate the color-adjustment factor between the target image and the reference image. The reference image is determined by the following procedure: First the three-channel RGB mean of all background images combined is calculated, and then the RGB mean of each background image is separately calculated to determine the reference image by choosing an image whose single-image RGB mean is the closest to the three-channel RGB mean of all the background images. In this way, the selected reference image color can be very representative. The method for calculation of the color-adjustment factor is as follows:

$$P_{\text{Diff}} = P_{r.\text{mean}} - P_{d.\text{mean}}, \quad (4)$$

$$\rho = 1 + \frac{\text{sign}(P_{\text{Diff}}) \times (P_{d,\text{mean}} - P_d)}{255}, \quad (5)$$

$$P' = P_d + P_{\text{Diff}} \times \rho, \quad (6)$$

where $P_{r,\text{mean}}$ represents the mean brightness of the reference image, $P_{d,\text{mean}}$ represents the mean brightness of the target image, and P_{Diff} represents the difference between the mean brightness of the target image and the reference image; $\text{sign}(P_{\text{Diff}})$ is a symbolic function; P_d represents the current brightness of a pixel in the target image, ρ represents the brightness adjustment factor of the target image, and P' represents the corrected brightness of a pixel in the target image. P in the formula represents three-component RGB.

2.3.2. Collage and fusion

The background image collage is produced by image stitching technology, which splices multiple background images into a seamless panoramic image on the canvas. Suppose that the number of background images to be collaged on the canvas is $I_x \times I_y$ based on I_x rows and I_y columns. The process to create the background image collage is as follows: The first step is to move through the I_{yi} columns from left to right on the canvas while making sure there is an appropriate overlap between adjacent images so that the images can transition smoothly; the next step is to move downward, return to the starting point of the next row, and fill that second row of images similarly from left to right, while making sure that each image not only overlaps with its adjacent images in the same row but also with the adjacent image above it; this acquisition process is to be repeated until all the I_x rows are filled.

To eliminate the splice marks, a weighted-average fusion algorithm is used for the overlapping parts of adjacent images. During data fusion, the fade-in and fade-out idea is applied to the image data in the overlapping area so that the transition area of the spliced images can be smooth and natural. For different coordinate points, the weighting coefficient β must be changed. Taking the horizontal direction as an example, assuming the leftmost position point of the overlapping area is the origin, the horizontal distance between other overlapping points and the origin is α , and the β value is set to be the ratio of α to the overlapping width o with its range within $(0, 1)$, revealing a linear change from left to right within the overlapping area. The changes in weighting coefficient β are the same in the vertical direction. The weighted-fusion formula is as follows:

$$P'_{i,j} = (1 - \beta) \times P_{i,j} + \beta \times P_{i,j-1}. \quad (7)$$

2.4. Collage of patterns

As soon as the collage of the canvas background image is completed, the pattern image collage can be created. In the pattern images generated by AC-DCGAN, there are also inconsistencies in brightness and color, so it is necessary to conduct color correction for each image, and the algorithm for this is the same as that for the background image. In addition, the colors of the background image and the flower image are not the same.

Unlike the background images, the pattern images contain more color information and are composed of a white pattern area on a blue background; the white pattern area could affect the color correction between the background image and the pattern image. If directly fused with the background image color correction and overlain, the blue background color of the pattern image may present a significant contrast with the surrounding background color. Therefore, an algorithm specifically for the key area collage is proposed to solve the problem of fusion between the pattern and the background images. Based on the image-segmentation algorithm, this algorithm extracts the white pattern area that reflects the characteristics of the pattern, then directly copies this area into the pattern position in the canvas, and finally uses a 3×3 filter window to perform mean filtering of three-channel-RGB canvas image to eliminate the edge effect.

The pattern image (Figure 1) is composed of a bright background and a foreground. The foreground pattern was extracted by using the Otsu algorithm for determining the binarization segmentation threshold of an image (Otsu, 1979; Satapathy et al., 2018). For the threshold obtained by this method, the class variance between the foreground and background image is largest after binarization segmentation of the image. Because the variance is a measurement of the uniformity of the gray distribution, the large variance between the background and the foreground suggests a significant difference between the two parts that make up the image, and the misclassification of either a partial foreground into the background or of a partial background into the foreground will result in a smaller difference between the two parts. Accordingly, segmentation maximizing such variance between classes minimizes the probability of misclassification. The specific algorithm method is to convert the image P from RGB to gray space, assume that the gray level of the image-segmentation threshold is T , and divide the image into two areas with gray levels of $[0, T]$ and $[T, 255]$ to correspond to the background and foreground, respectively. The proportions of background and foreground pixels in the image are θ_1 and θ_2 respectively, and their average gray levels in the area are u_1 and u_2 , respectively; the average gray level is set to u , which is obtained by

$$u = \theta_1 \times u_1 + \theta_2 \times u_2, \quad (8)$$

and the between-class variance is obtained by

$$\sigma^2 = \theta_1(u_1 - u)^2 + \theta_2(u_2 - u)^2, \quad (9)$$

where the effect threshold of the segmentation is better when σ^2 is at a maximum, and the T that corresponds to the maximum σ^2 is calculated as

$$T = \underset{0 \leq T \leq 255}{\text{argmax}} \sigma^2. \quad (10)$$

3. Experimental results and analysis

To verify the effectiveness of the algorithms used in this study and to achieve the generation of tie-dyeing patterns, we

constructed six small-category datasets consisting of the tie-dyeing patterns and background images. We performed AC-DCGAN training and primitive generation verification based on these datasets and generated digital tie-dye patterns almost as vivid as the real images.

3.1. Experiment

Point, line, and plane are the basic elements that make up a pattern. This experiment started from the basic point-like model shape; a total of five types and 10,000 pieces of the most-common point and circular model shapes were collected in the tie-dyeing bundling and sewing process, all of which had white patterns on blue backgrounds; 2,000 images with blue backgrounds were also collected for this experiment. The size of each image was set to 96×96 pixels, and the images were in RGB color mode. The model used the TensorFlow 1.1.4 deep-learning framework and the Compute Unified Device Architecture 10.0. Batch size for each training was 100, and the experiment involved a total of 10,000 epochs (Figure 5). Optimizing the generator and discriminator, β_1 was set to 0.5, and learning rate to 0.0005. The experiment was performed by using the Windows 10 operating system on the

Anaconda–Spyder 4.6.14 development platform. Hardware configuration for the experiment included the Intel Core E5-2650 v4 CPU (main frequency 2.2 GHz), 2 GTX 1080Ti graphics cards, a 1T hard disk, and a 16 GB memory.

3.2. Results and analysis

The duration of the image training was 30 h. To make a more-intuitive observation and comparison of the quality of the images generated by using different numbers of training generations, we randomly stacked the 64 generated images of different generations in the training process to form a large image. The 0 generation is a pattern generated by random noise without any effective information. The 500th-generation pattern has learned the approximate shape of the pattern, but its texture is indistinct. When the network training attains the 2,000th generation, the outline of the pattern is basically formed, with its texture very close to the real one, but the details of the pattern are not clear enough, as shown in the first, third, and fourth class diagram. Too, there is also obvious noise in the blue background area. In the 10,000th generation, however, the images generated by AC-DCGAN show clear details, are

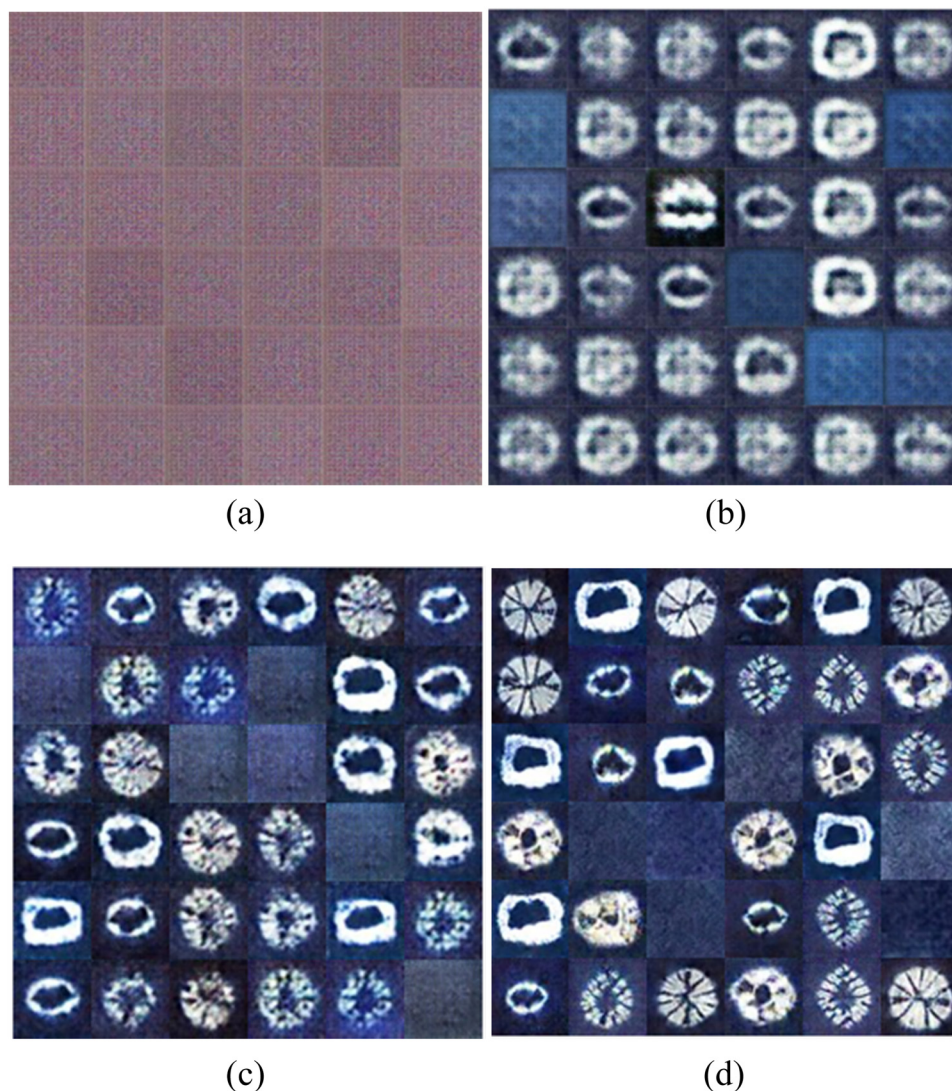


Figure 5. Diagram of samples generated in training process. (a) 0 Generation, (b) 500th generation, (c) 2,000th generation, and (d) 10,000th generation.

very similar to the original training images, and have only minor noise, thereby closely reflecting core artistic characteristics of the tie-dyeing process. Accordingly, the network parameters governing the 10,000 generations should be saved for generating large digital tie-dyeing patterns.

To quantify the pattern effect, we calculate the Peak Signal to Noise Ratio (PSNR) and Structural Similarity Index Metric (SSIM) values of GAN and AC-DCGAN, respectively (Zhou and Gu, 2004). PSNR and SSIM are universal evaluation indicators of image generation quality. PSNR is an objective evaluation index used to evaluate the noise level or image distortion. The larger the PSNR, the less the distortion, and the better the quality of the generated image. SSIM is used to evaluate the similarity level between two images. The higher the similarity, the more similar. The lowest PSNR value of this research method is 22.39, which indicates that this research method has the lowest image distortion. The lowest SSIM value of AC-DCGAN is 0.81, which indicates that AC-DCGAN-generated image has the highest structural similarity with the original image. The values of PSNR and SSIM are consistent with the subjective evaluation of patterns, reflecting that the image generated by AC-DCGAN has the lowest distortion and the highest structural similarity (Figure 6).

Thus, we can summarize the digital tie-dyeing pattern-generation process as follows. The first steps are to design a layout diagram and use the regional-connectivity algorithm to extract the positions of all the circles in the diagram and mark them with red squares so that all circles can be correctly identified (Figure 7a). Then combined color correction and fusion operations are applied to process the background image generated by AC-DCGAN and to form the layout diagram (Figure 7b); there should be no obvious color difference between the small images in the diagram and the transition in the edge area, where the transition should be uniform. The next steps are to perform the segmentation of the white pattern area of the tie-dyeing pattern generated by AC-DCGAN by applying the Ostu algorithm and to directly copy this area to the pattern position in the canvas. Finally, mean filtering is to be applied to the RGB three-channel space of the canvas image. The pattern area should be well preserved, with no edge effect in the area connected to the background (Figure 7c).

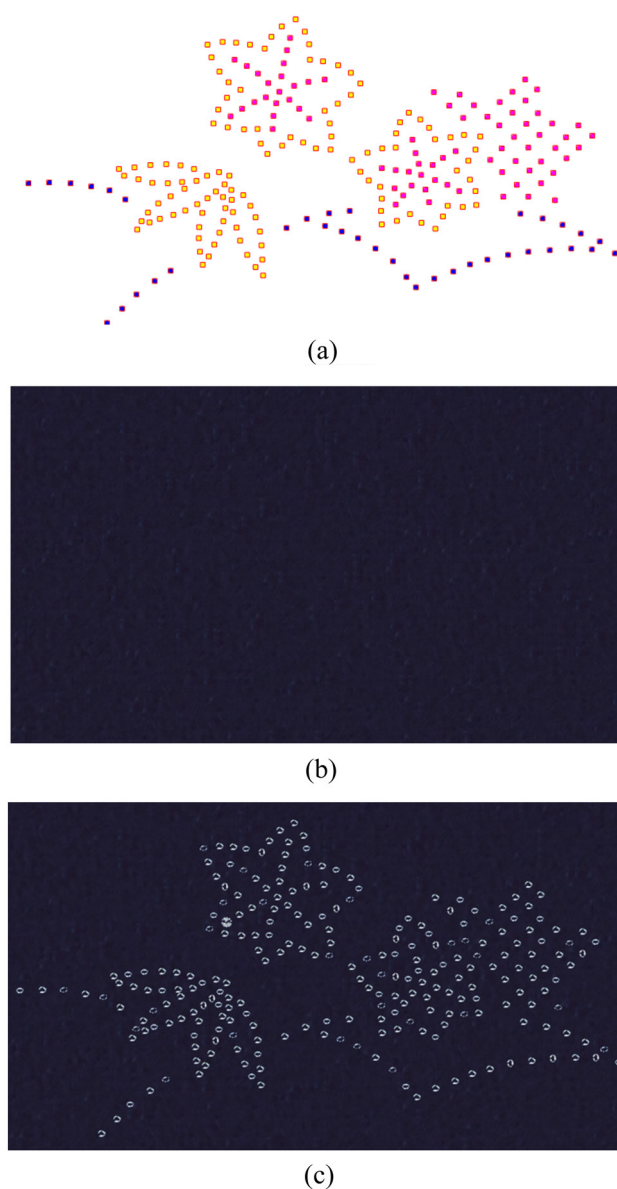


Figure 7. Tie-dyeing pattern-generation process diagram. (a) Extraction of pattern position, (b) background collage and fusion image, and (c) collage image of key areas of pattern.

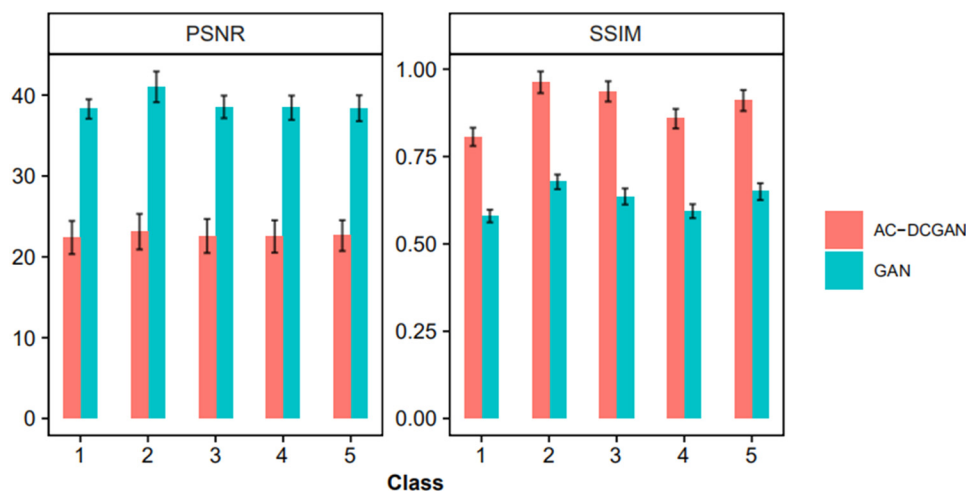


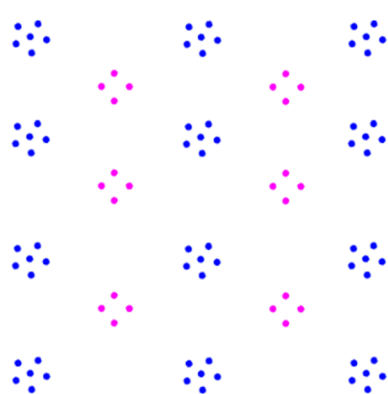
Figure 6. Performance comparison among GAN and AC-DCGAN on five tie-dye patterns in Figure 2.

According to the requirements of the digital tie-dyeing pattern-generation process, after the layout diagram is designed on a computer, it is possible to freely design and adjust the process variations and combinations according to the characteristics of the AC-DCGAN pattern source, thereby quickly generating a variety of tie-dyeing patterns. Various tie-dyeing pattern layouts and pattern-generation diagrams are shown in Figure 8. As seen from the perspective of the artistic effect, the patterns contain abundant and contrasting blue and white color layers. Each single small point reflects the characteristics of the tie-dyeing process without repetition, and the proportion and spacing between the points are appropriate and coordinated, enabling the whole pattern to present a natural

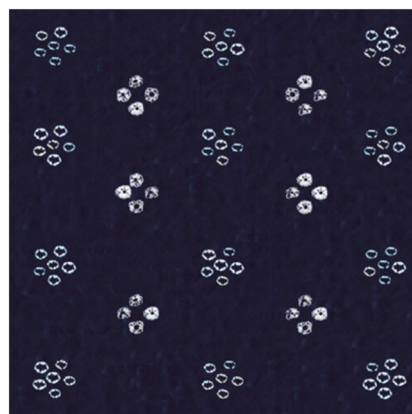
and dynamic impression, thus achieving a highly realistic artistic effect.

4. Conclusions

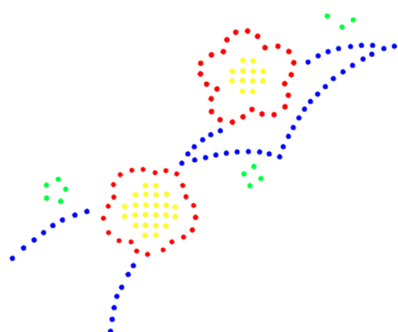
In this study, the traditional tie-dyeing process is simulated and a digital tie-dyeing pattern-generation algorithm is proposed that is based on deep-learning and image-processing technologies. The algorithm constructs an AC-DCGAN that can generate six types of graphs, which can be used for generating tie-dyeing patterns and background images. In the process of pattern-generation, the designer first needs to draw the



(a)



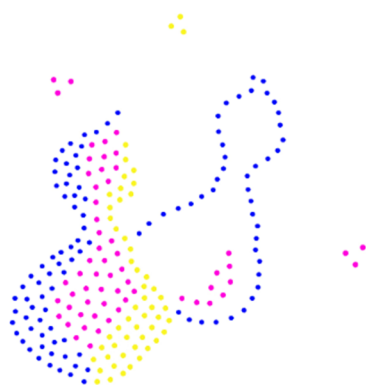
(b)



(c)



(d)



(e)



(f)

Figure 8. Tie-dyeing pattern layout and image-generation diagrams. (a) Scattered-point flower-pattern layout diagram, (b) generated image, (c) plum-blossom pattern layout diagram, (d) generated image, (e) gourd-shaped pattern layout diagram, and (f) generated image.

tie-dyeing pattern layout diagram, consisting of a white background and multicolored circles, and then use the regional-connectivity algorithm to extract the information about the center positions of the circles as well as the information on the categories of the patterns. The next step is color correction of a number of background images to splice together a complete background image. Then color correction for tie-dyeing patterns is to be performed, and the pattern area is segmented and copied to the area where the circles are located. Finally, mean filtering is applied to the entire image to achieve the final digital tie-dye pattern. The patterns generated contain many contrasting color layers, and each pattern could be said to reflect the characteristics of the tie-dye process without repetition, and the whole pattern image presents a natural and dynamic impression, thus achieving a highly realistic artistic effect. Combined with the flexible and changeable combination of process variation represented by the dot-matrix layout diagram, various tie-dye patterns can be quickly generated.

Acknowledgments: This work was supported by National first-class professional construction point support project, Jiangsu Province key discipline support project, Jiangsu University Philosophy Society Project (2020SJA0547).

Conflict of interest: Authors state no conflict of interest.

Code availability: All codes are publicly available and can be obtained at <https://github.com/LorMeBioAI/GAN>.

References

- [1] Gu, M. (2004). The general talk about technique of modern tie-dye. *Journal of Donghua University*, 3, 41–45.
- [2] Liu, S. Q., Gao, W. D., Xue, W., Gu, M., Liang, H. E. (2016). Tie-dye technique and pattern features. *Indian Journal of Fibre & Textile Research*, 41, 180–187.
- [3] Carter, N. C., Eagles, R. L., Grimes S. M., Hahn A. C., Reiter C. A. (1998). Chaotic attractors with discrete planar symmetries. *Chaos Solitons and Fractals*, 9, 2031–2054.
- [4] Lu, J., Ye, Z. X., Zou, Y. R. (2005). Orbit trap tendering methods for generating artistic images with crystallographic symmetries. *Computers & Graphics*, 29, 787–794.
- [5] Lv, J., Pan, W., Liu, Z. (2014). Method of batik simulation based on interpolation subdivisions. *Journal of Multimedia*, 9, 286–293.
- [6] Lu, S., Mok, P. Y., Jin, X. (2017). A new design concept: 3D to 2D textile pattern design for garments. *Computer-Aided Design*, 89, 35–49.
- [7] Tian, G. D., Yuan, Q. N., Hu, T., Shi, Y. (2019). Auto-generation system based on fractal geometry for batik pattern design. *Applied Sciences*, 9, 2383.
- [8] Wang, W. J., Zhang, G. P., Yang, L. M. (2019). Research on garment pattern design based on fractal graphics. *EURASIP Journal on Image and Video Processing*, 2019, 29.
- [9] Zhou, J. (2004). Digital jacquard fabric design in colorful mode. *Journal of Donghua University*, 21, 98–101.
- [10] Zhou, J. (2007). Innovative principle and method for digital jacquard fabric designing. *Journal of Donghua University*, 24, 341–346.
- [11] Barnsley, M., Hurd, A. J. (2000). *Fractals everywhere*. *American Journal of Physics*, 97, 1053.
- [12] Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Bing, X., David, W.-F., Sherjil, O., et al. (2014). Generative adversarial nets. *Advances in Neural Information Processing Systems*, 3, 2672–2680.
- [13] Creswell, A., White, T., Dumoulin, V., Arulkumaran K., Sengupta B., Bharath A. A. (2017). Generative adversarial networks: An overview. *IEEE Signal Processing Magazine*, 35, 53–65.
- [14] Azadi, S., Fisher, M., Kim, V. G., Wang, Z., Shechtman, E., Darrell, T. (2018). Multi-content gan for few-shot font style transfer. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pp. 7564–7573.
- [15] Li, P., Hu, Y., Li, Q., He, R., Sun, Z. (2018). Global and local consistent age generative adversarial networks. *Proceedings of the 24th International Conference on Pattern Recognition, Beijing, China*. pp. 1073–1078.
- [16] Yang, H., Zhu, K., Huang, D., Li, H., Wang, Y., Chen, L., et al. (2021). Intensity enhancement via GAN for multi-modal face expression recognition. *Neurocomputing*, 454(1), 124–134.
- [17] Huang, R., Zhang, S., Li, T., He, R. (2017). Beyond face rotation: Global and local perception gan for photorealistic and identity pre-serving frontal view synthesis. *Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy*. pp. 2458–2467.
- [18] Santana, E., Hotz, G. (2016). Learning a driving simulator. *arXiv preprint, arXiv: 1608.01230*.
- [19] Chen, W., Hays, J. (2018). Sketchygan: Towards diverse and realistic sketch to image synthesis. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pp. 9416–9425.
- [20] Mirza, M., Osindero, S. (2014). Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*.
- [21] Radford, A., Metz, L., Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint, arXiv:1511.06434*.
- [22] Odena, A., Olah, C., Shlens, J. (2017). Conditional image synthesis with auxiliary classifier GANs. *Proceedings of the 34th International Conference on Machine Learning, Sydney, Australia. PMLR* 70.
- [23] Tian, J., Li, X., Duan, F., Wang, J., Ou, Y. (2016). An efficient seam elimination method for UAV images based on Wallis dodging and Gaussian distance weight enhancement. *Sensors*, 16, 662.
- [24] Li, W. Z., Sun, K. M., Li, D. R., Bai, T. (2016). Algorithm for automatic image dodging of unmanned aerial vehicle images using two-dimensional radiometric spatial attributes. *Journal of Applied Remote Sensing*, 10, 036023.
- [25] Ostu, N. (1979). A threshold selection method from gray-histogram. *IEEE Transactions on Systems, Man, and Cybernetics*, 9, 62–66.
- [26] Satapathy, S. C., Sri Madhava Raja, N., Rajinikanth, V., Ashour, A. S., Dey, N. (2018). Multi-level image thresholding using Otsu and chaotic bat algorithm. *Neural Computing and Applications*, 29, 1285–1307.
- [27] Zhou, J., Gu, J. (2004). Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13, 600–612.