Fast textile pattern generation combining MRF-based and Gram-based methods

DOI: 10.35530/IT.074.04.202254

YUEXIN SUN YU CHEN

ABSTRACT – REZUMAT

Fast textile pattern generation combining MRF-based and Gram-based methods

Textile pattern design is a tedious and challenging task for designers. This paper proposes a fast textile pattern generation algorithm that combines MRF-based and Gram-based methods. First, the reconstruction method based on image optimisation is determined after analysing the specific requirements of textile pattern design. The pre-trained VGG19 is selected as the style feature extraction neural network. Then, we compare the generation results of various combinations of style loss functions and propose a multi-resolution image optimisation method. Finally, the smoothing loss and colour histogram matching are added to improve the generation quality further, thus constructing an image generation algorithm for textile pattern design. Experimental results demonstrate that our algorithm can effectively generate complex textile patterns with global style and local detail features. The average image generation time is 575s, over 84.3% faster than traditional algorithms. At the same time, this algorithm is convenient for switching styles and requires lower computational capability. It can improve pattern design efficiency and promote the application of image generation technology in textile design.

Keywords: *artificial intelligence, computer-aided design, image generation, neural networks, textile pattern design*

Generare rapidă de modele textile, combinând metode bazate pe MRF și pe Gram

Designul modelelor textile este o sarcină minuţioasă și provocatoare pentru designeri. Această lucrare propune un algoritm de generare rapidă a modelelor textile, care combină metode bazate pe MRF și pe Gram. În primul rând, metoda de reconstrucție bazată pe optimizarea imaginii este determinată după analiza cerințelor specifice ale designului modelului textil. VGG19 pre-antrenat este selectat ca rețea neuronală de extracție a caracteristicilor de stil. Apoi, se compară rezultatele generării diferitelor combinații de funcții de modficare a stilului și propunem o metodă de optimizare a imaginii cu rezoluții multiple. În cele din urmă, diminuarea netezirii și histograma potrivirii culorii sunt adăugate pentru a îmbunătăți și mai mult calitatea generării, construind astfel un algoritm de generare a imaginii pentru designul modelelor textile. Rezultatele experimentale demonstrează că algoritmul nostru poate genera în mod eficient modele textile complexe cu stil global și caracteristici de detalii locale. Timpul mediu de generare a imaginii este de 575 de secunde, ceea ce este cu peste 84,3% mai rapid decât algoritmii tradiționali. În același timp, acest algoritm este convenabil pentru schimbarea stilurilor și necesită o capacitate de calcul mai mică. Se poate îmbunătăți eficiența designului modelului și se poate promova aplicarea tehnologiei de generare a imaginii în designul textil.

Cuvinte-cheie*: inteligență artificială, proiectare asistată de calculator, generare de imagini, rețele neuronale, design de modele textile*

INTRODUCTION

Textile patterns are extensively used in digital textile printing. Traditional textile pattern design requires designers to collect inspirational images to extract motifs or textures, then repeat and mend them to create a continuous textural design without seams appearing [1]. This work is time-consuming and requires designers with inventiveness, information extraction and expression skills, which need years of professional training [2]. In recent years, artificial intelligence has provided the textile and apparel industry with various methods and solutions [3–4], and neural style transfer has been steadily used in textile design [5]. This technology can automatically extract style features from a target style image and shows them on a content image. Visual style modelling and generation techniques of style transfer can be used in the design process of textile patterns. The pattern features in an inspirational image, which is the style image in style transfer, can be modelled and generated into a pattern image by reconstruction. With the help of pattern generation techniques, design efficiency can be significantly improved.

However, the current research object of style modelling and transfer focuses mostly on artistic painting, which cannot sufficiently meet the specific requirements for textile pattern design. To liberate human designers from laborious and time-consuming design work and to encourage the implementation of image-generating technology in textile design, it is crucial to research the generation algorithms for textile pattern design.

Texture generation is the foundation of pattern generation. Markov Random Field (MRF) [6] can model and extract simple texture features from images. However, generation with pixel-by-pixel matching uses a great deal of time. With the significant advance in deep learning, more powerful neural style transfer became popular. Gatys et al. [7] found that colour and texture can describe the style features of an image. The style features were extracted by convolutional neural networks (CNN) and modelled with Gram matrices, which can be used to compute style loss and generate images. Li et al. [8] combined MRF with CNN for style transfer. Patches are obtained by CNN and then matched by MRF in the feature spaces to generate images with local detail features. Fayyaz et al. [9] used adversarial generative networks to generate complex textile patterns automatically but required a professional graphics processing unit (GPU) to train a neural model for an extended period. Jiang et al. [10] argued that compared to painting style transfer, fashion style transfer needs to generate not only the global features of style image but also the local details features of patterns.

Therefore, textile pattern generation should address the balance between global style and local pattern details. Secondly, designers visualise various inspirational images frequently. In addition to maintaining generation quality, increasing generation speed and facilitating style switching is required. Lastly, traditional deep learning relies heavily on the computational performance of GPU, and designers do not have specialised scientific computers to deploy large neural network models. The algorithm's model must consider the generation feasibility in the central processing unit (CPU), reducing GPU consumption.

Based on the specific requirements of textile pattern design, we propose a multi-resolution optimisation strategy to combine the MRF-based and Gram-based methods. Low-resolution image generation uses the MRF-based method for local pattern features. After upsampling to a high-resolution image, the Gram-based method is used to refine the global image style. This strategy significantly decreases time while preserving the quality of image generation, building a fast textile pattern generation algorithm with the smoothing function and histogram matching.

TEXTILE PATTERN GENERATION ALGORITHM

Style feature extraction neural network

Style extraction network is the basis of image style modelling. Wang et al. [11] discovered that new network architectures, such as residual neural network (Res-Net), are inappropriate for style feature extraction. The residual connection will reduce the entropy value of the feature map, which needs to add softmax layers to smooth the feature maps. However, the pre-trained Visual Geometry Group network (VGG) performs well in style feature extraction without additional layers. Therefore, pre-trained VGG19 is selected as the style feature extraction neural network, where '19' represents the number of convolutional layers. The original VGG19 network consists of 16 convolutional layers and 3 fully connected layers [12]. During the style extraction process, network layers for classification tasks after conv5_1 in the VGG19 are eliminated to save computational capability.

In VGG19, shallower feature maps capture more texture features, and deeper feature maps capture more sophisticated semantic features [13]. To fully cover textile pattern features, it is important to combine features from different layers in the VGG19.

Image reconstruction methods

After the style features are extracted, they need to be reconstructed into an image. Currently, image reconstruction approaches are divided into descriptive methods based on image optimisation and generative method based on model optimisation [14]. The first method reconstructs by iteratively optimising the initial image pixel by pixel. This method focuses on a target-style image, requires less computational capability and can be generated by CPU only. Moreover, it offers better image quality and is convenient to switch pattern styles. However, as this method involves multiple backpropagations, image generation takes a longer time. The second method can generate images using forward propagation after generative models are trained, resulting in significant time savings. However, model training needs large-scale datasets and professional GPUs. Each style demands hours or even days of training, making style changes difficult. In addition, many model-based methods employ the same setups as image-based methods [15–16], such as the feature extraction model and loss function. Due to the utilisation of images in the dataset rather than a specific style map during training, the image quality generated by the second method is inferior to the first one and lacks pattern details.

In conclusion, our research selected the reconstruction method based on image optimisation. It uses VGG19 to extract and model the style features of the generative and target style images. The style loss between these two images is computed using a specific loss function based on the extracted style features. The generative image will be optimised based on the style loss value. The initial image to generate uses a random noise image. This method does not need datasets or a significant amount of time to train generative models. Simultaneously, it can generate high-quality pictures, and it is convenient to switch styles, making it more suitable for textile pattern generation.

Comparison of style loss functions

Style loss is mainly divided into two types: Gram-based loss (L_G) and MRF-based loss (L_M) [17]. Gram matrix consists of inner products of specific vectors. The angular and directional relationships between two vectors can be expressed by their inner product. When computing style loss, a specific convolution kernel encodes the feature maps to Gram matrices. Its diagonal components represent the number of different features. The Gram matrix can therefore demonstrate the number relationship of each feature and the interrelationships between features, thus modelling the image's global style features. After obtaining the respective Gram matrices for the style and generation images using VGG19, the style loss of each layer is calculated using mean square error. The sum of all layer losses is L_G between the generative and style images. The process details are shown in step 2 in figures 1 and 2 and the L_G can be expressed as follows:

$$L_G(\hat{y},y_s) = \sum_l \left\| G^{[l]}(\hat{y}) - G^{[l]}(y_s) \right\|_2^2 \tag{1}$$

where $G^{[l]}(\hat{y})$ and $G^{[l]}(y_s)$ are the Gram matrices at layer [l] in the VGG19 of the generative image ($\hat{y}$) and target style image (y_s) respectively.

MRF is a classical texture modelling method. The generative and style images are extracted separately as patches. Each patch in the generative image is matched with the most similar patch from the style image patch library for filling or as a generation reference. Currently, MRF often combines CNN to calculate L_M. First is data augmentation for the style image, such as scaling and rotation. Then, style patches of the augmented style images and the generative image are obtained using VGG19. Utilising the feature maps of {conv3_1 and conv4_1} layers to extract patches for enhanced patch matching [8]. Patches of the generative and augmented style images are matched by computing the cosine similarity. It means encoding style features using MRF. Finally, the style loss is obtained by calculating the difference between the patches of the generative image and the matched patches of augmented style images. The process details are shown on step1 in figure 2 and L_M can be expressed as follows:

$$L_M(\hat{y},y_s) = \sum_l \sum_k \left\| P_k\left(F^{[l]}(\hat{y})\right) - P_{MRF(k)}\left(F^{[l]}(Aug(y_s))\right) \right\|^2 \tag{2}$$

where $P_k\left(F^{[l]}(\hat{y})\right)$ is the k-th patch extracted by feature maps of $\hat{y}$ at layer [l]; $Aug(y_s)$ is a series of images obtained by y_s after image augmentation; $P_{MRF(k)}\left(F^{[l]}(Aug(y_s))\right)$ is the k-th MRF-matched patch extracted by feature maps of augmented y_s at layer [l].

Image generation using L_G and L_M, respectively. Target style image: Chinese pattern picture shown in figure 1, a; feature maps for L_G: {conv1_1, conv2_1, conv3_1, conv4_1, conv5_1} layers; rotation angles for L_M: 0°, 90°, –90° and 180°; optimiser: Adam; learning rate: 1e-2; number of iterations: 1000; deep learning framework: PaddlePaddle 2.2; programming language: Python 3.7; development platform: Baidu AI Studio; operation system: Ubuntu 16.04; GPU: NVIDIA Tesla V100; GPU memory: 16GB; CPU: Intel(R) Xeon(R) Gold 6148 @ 2.40 GHz; random access memory: 16 GB; hard disk drive: 100GB.

The results are shown in figure 1. It took 34 seconds with L_G and 1125 seconds with L_M for the size of 800*800 pixels. Generation using L_G is faster and more sensitive to global styles such as colour and texture. However, as shown in figure 1, *b*, L_G cannot generate local pattern details. It is challenging to generate intricate and specific pattern features, such as petals and stems. L_M is a patch-based loss that is more sensitive to local pattern features, but the portrayal of global features is inadequate, as shown in figure 1, *c*. Because substantial patches need to be extracted and matched for each round of iterative optimisation, image generation using L_M is time-consuming. Therefore, a combination of these two losses may speed up the generation process and satisfy the requirements for global style and local pattern features in textile pattern generation.

Combination of style loss functions

Previous research [18–19] has often added L_G and L_M to generate patterns with both global style and local features. However, direct summation can lead to mutual interference and reduce generation quality, since L_G and L_M have an adverse effect on the generation process. By referring to the image pyramids [20] and generative adversarial networks [21], we suggest five combination strategies to assess the quality and duration of image generation individually, as shown in table 1.

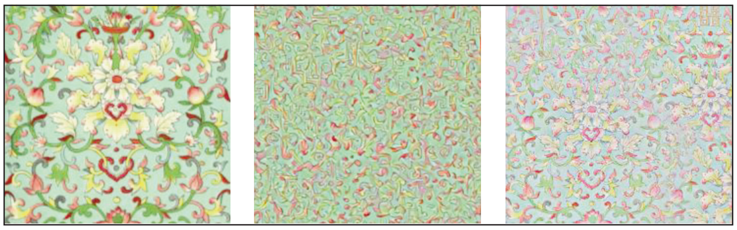
Figure 1, *a* is the target style image, 1000 times for each step, and the final image size is 500*500 pixels. The generated images are scaled to the same size, as shown in table 2. The generation time is shown in table 3 for every option.

In option 1, as the values of L_G and L_M have distinct orders of magnitude, it is simple for one loss to suppress another, resulting in a generated image representing just a single kind of feature. As shown in table 2, only global style features are shown after step 1. Local pattern features like petals and stems are absent from the generated image. It is difficult to

Table 1

STYLE LOSS FUNCTION COMBINATION STRATEGIES

№	Option descriptions	
	Step1	**Step2**
1	$L_G + L_M$ (high-resolution generation)	/
2	down-sampling, L_G (low-resolution generation)	up-sampling, L_M (high-resolution generation)
3	L_G (high resolution-generation)	L_M (high-resolution generation)
4	down-sampling, L_M (low-resolution generation)	up-sampling, L_G (high-resolution generation)
5	L_M (high-resolution generation)	L_G (high-resolution generation)



a *b* *c*

Fig. 1. Generation effects of different loss functions: *a* – target style image; *b* – generated image by L_G; *c* – generated image by L_M

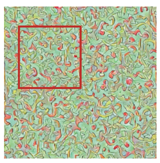
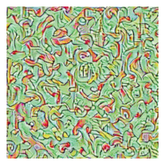
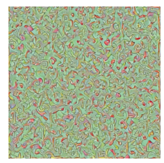
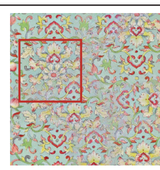
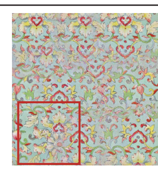
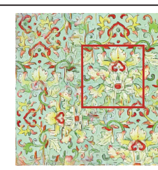
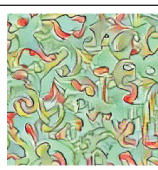
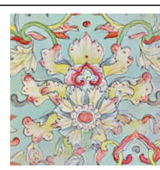
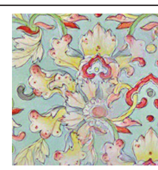
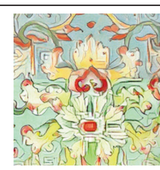
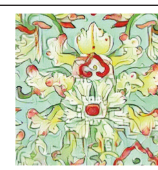
generation time is reduced by 75.5% compared to option 5.

In option 5, the generated image is similar to option 4, and the second step using L_G keeps the pattern basic structures generated in step 1. The generation process takes longer due to the use of L_M in the first step to generate a high-resolution image.

It can be found from the above experiments that using L_G is faster and easier to maintain

Table 2

IMAGES GENERATED BY DIFFERENT COMBINATION STRATEGIES, INCLUDING INTERMEDIATE RESULTS

Option №	1	2	3	4	5
Step1					
Step2	/				
Image details					

Note: The regions in red frames are magnified to provide more details.

optimise the balance between these two losses and to generate images with global style and local pattern features.

In option 2, the generated image of the step 1 is rough because of global features generation in low resolution. Pattern position information is missing in the subsequent step. The local pattern features in the generated image of step 2 are random and cannot follow the structures and shapes in the resulting image of step 1. The final effect tends to use L_M alone, and the image background appears colour cast. In addition, the patch exaction and matching process consume a long time due to using L_M to generate high-resolution image. Too many patches may easily cause memory overflow when an image reaches 500*500.

In option 3, the generated image is similar to option 2. The colour and other style features from step 1 are overwritten in step 2. Since both steps generate high-resolution image and L_G is insensitive to resolution increase, the total time savings is insignificant compared to option 2. Option 3 also has the problem of memory overflow.

In option 4, Local pattern features are generated in step 1, and global style is refined in step 2. As a result of reducing image resolution in step 1, the generation time of L_M is significantly shorter. The total

Table 3

GENERATION TIME FOR DIFFERENT COMBINATION STRATEGIES

Items	Option №				
	1	2	3	4	5
Step 1 time (s)	1354	18	34	240	1093
Step 2 time (s)	/	1121	1020	35	31
Total time (s)	1354	1139	1054	275	1124

pattern structural features in the initial image. Using L_M for generation is time-consuming and overwrites the original structural features. Therefore, based on the advantages and disadvantages of L_G and L_M, we proposed a multi-resolution image optimisation method. L_M for low-resolution generation, upsampling and L_G for high-resolution generation. The resulting images are similar to those generated at high resolution in both steps. This method can quickly generate textile patterns with global style and local detail features.

Algorithm framework

After the basic procedure is determined, the generated image contains noises. Therefore, a smoothing function (equation 3) is needed to reduce noise by minimising the image's total variation [22].

$$L_{TV}(\hat{y}) = \sum_{i=1}^{h-1}\sum_{j=1}^{w}\sum_{c=1}^{3}(x_{i+1,j,c} - x_{i,j,c})^2 + \\ + \sum_{i=1}^{h}\sum_{j=1}^{w-1}\sum_{c=1}^{3}(x_{i,j+1,c} - x_{i,j,c})^2 \quad (3)$$

where h, w and c denote the generated image's height, width and channel.

Therefore, it is determined that L_M is used for low-resolution textile pattern generation in the first step, which is called the MRF-based method. L_G is is used for high-resolution generation in the second step, which is called the Gram-based method. Their loss functions are shown below, with equation 4 for the MRF-based method and equation 5 for the Gram-based method:

$$L'(\hat{y}', DS(y_s)) = L_M(\hat{y}', DS(y_s)) + \lambda_1 L_{TV}(\hat{y}') \quad (4)$$

$$L(\hat{y}, y_s) = L_G(\hat{y}, y_s) + \lambda_2 L_{TV}(\hat{y}) \quad (5)$$

where $\hat{y}'$ is the generative image of the first step; $DS(y_s)$ is a down-sampled image of y_s; the initial of $\hat{y}$ is an up-sampled image of $\hat{y}'$; λ_1 and λ_2 are the weights of the smoothing term.

Risser et al. [23] pointed out that Gram matrices do not store statistical information such as mean and variance, leading to unstable image generation. The statistical information of the style features is included after a histogram matching process, which can increase generation quality and stability. Therefore, adding histogram matching after the second step can bring the colours of the textile pattern image closer to the colours of the style image.

In summary, the workflow of the textile pattern generation algorithm is as follows:

(a) Down-sampling a target style image to reduce the size.

(b) Generating a low-resolution image with local pattern features by the MRF-based method.

(c) Up-sampling the resulting image of the MRF-based method to the final size by interpolation.

(d) Refining the global style of the up-sampled image by the Gram-based method.

(e) Histogram matching between the resulting image of the Gram-based method and the target style image.

The framework is shown in figure 2. Additional upsampling and optimisation processes can be added for higher resolution and more detailed portrayals.

DISCUSSION

Comparing the pattern generation quality and duration of Johnson's [15], Li's [8] and our algorithms. The generated image size is 500*500 pixels; 2000 times for each step; COCO2017 [24] for the model training set; and the critical hyper parameter settings are maintained. The generated images are shown in table 4. Johnson's algorithm takes longer to train the model, with an average total time of 21134 s for training and generation. The global style of the patterns in

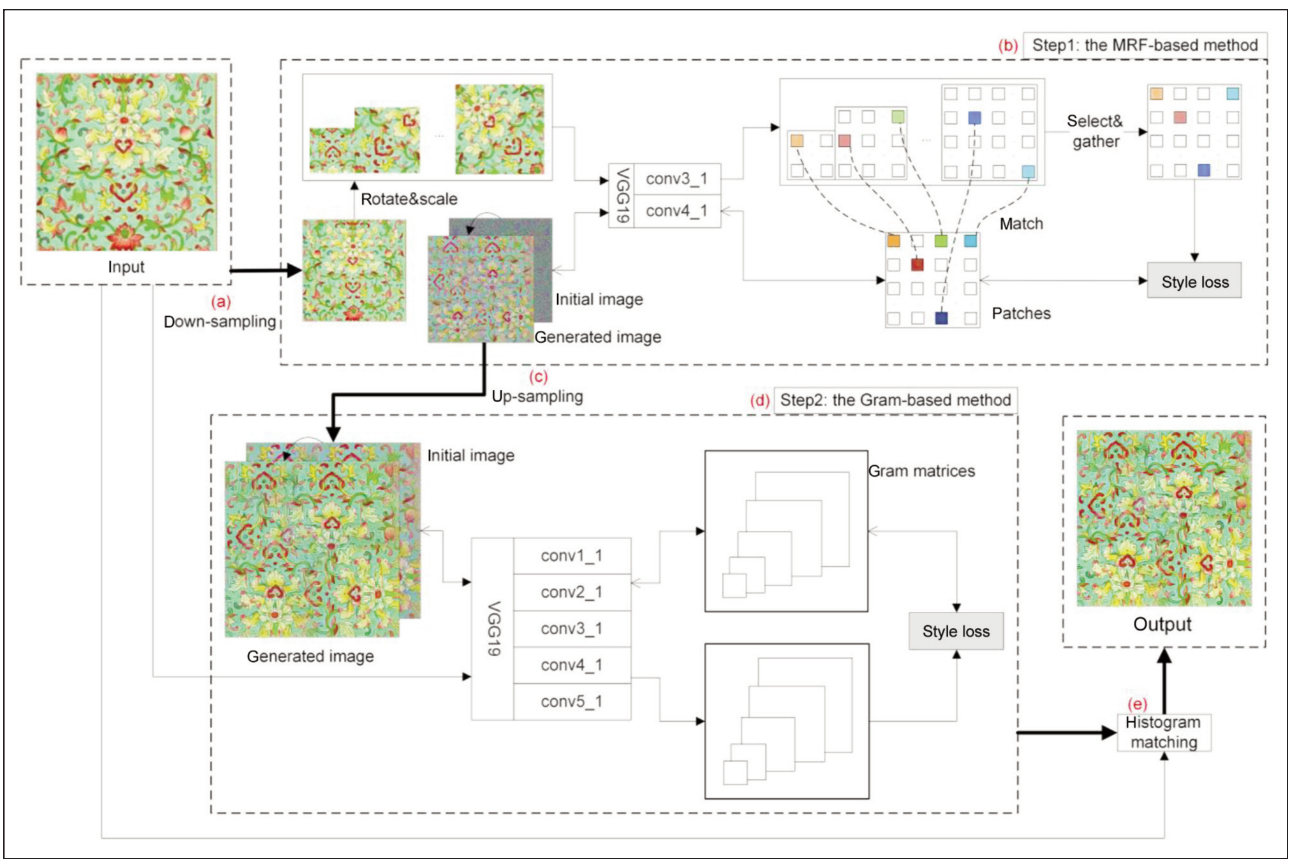
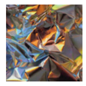
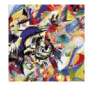
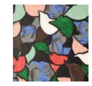
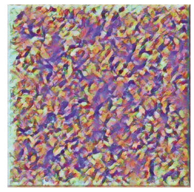
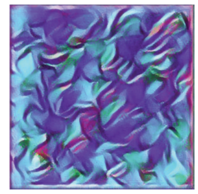
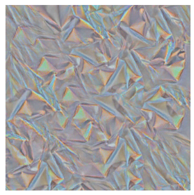
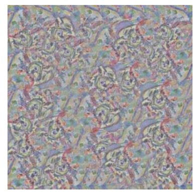
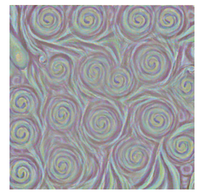
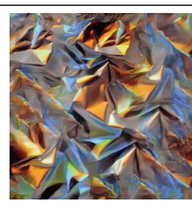
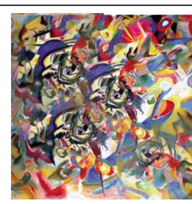
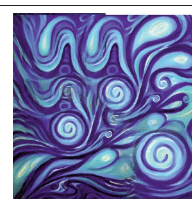
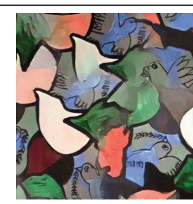


Fig. 2. Framework overview of the fast textile pattern generation algorithm
(the bidirectional arrows are paths of backpropagation)

Table 4

EXAMPLES OF TEXTILE PATTERN GENERATION BY DIFFERENT ALGORITHMS					
Option №	**1**	**2**	**3**	**4**	**5**
Style image					
Johnson's					
Li's					
Ours					

the generated images closely resembles that of the style images, but local detail features are absent. Li's algorithm takes an average of 3655 s to generate and does not need model pre-training. The generated images show local details but do not adequately refine the global style for various style images. Our algorithm also does not need to pre-train the model and the average generation time is 575 s. The resulting images with our algorithm generate global and local features of different target styles, which can be used as textile printings.

Limitations and future work

Three limitations in this paper could be addressed in future works:

1. Our algorithm does not solve the issue of the slow MRF-based method fundamentally. To increase efficiency, we will attempt to save patch-matching results for reuse in the following processes.
2. The patterns generated by the current algorithm are relatively random, and the algorithm's functionality needs to be expanded to generate more pattern effects, such as symmetry and gradients. It will enable the algorithm to be used in a broader range of design scenarios.
3. This algorithm can only generate patterns. In the future, we will include content loss to build a fashion style transfer algorithm for showing patterns directly on textiles or apparel.

CONCLUSIONS

This paper proposes a fast textile pattern generation algorithm combining MRF-based and Gram-based methods. With a multi-resolution optimisation strategy, this algorithm can generate complicated textile pattern images with global style and local details. The average generation time for a 500*500 pixels image is 575 seconds, more than 84.3% faster than traditional algorithms. In addition, it is simple to switch target styles and can generate images by CPU only. Therefore, it can increase the productivity of designers and encourage the application of artificial intelligence in textile design.

ACKNOWLEDGEMENTS

Research supported by The Program for Professor of Special Appointment (Eastern Scholar) at Shanghai Institutions of Higher Learning (TP2017074).

REFERENCES

[1] Studd, R., *The Textile Design Process*, In: The Design Journal, 2002, 5, 1, 35–49, https://doi.org/10.2752/146069202790718567

[2] Arslan, P., *Creative Process in Textile Pattern Design: Wallas' Four-Stage Model*, In: Journal of Academic Social Science Studies, 2021, 14, 85, 199–218, https://doi.org/10.29228/JASSS.49901

[3] Sikka, M.P., Sarkar, A., Garg, S., *Artificial Intelligence (AI) in Textile Industry Operational Modernization*, In: Research Journal of Textile and Apparel, 2022, ahead-of-print, https://doi.org/10.1108/RJTA-04-2021-0046
[4] Zou, X., Wong, W., *FAshIon after Fashion: A Report of AI in Fashion*, In: ArXiv, 2021, https://doi.org/10.48550/arxiv.2105.03050
[5] Pang, Z., Wu, S., Zhang, D., Gao, Y., Chen, G., *NAD: Neural Network Aided Design for Textile Pattern Generation*. In: Proceedings of the 28th ACM International Conference on Information and Knowledge Management, 2019, 2081–2084, https://doi.org/10.1145/3357384.3358103
[6] Efros, A.A., Leung, T.K., *Texture Synthesis by Non-Parametric Sampling*. In: Proceedings of the Seventh IEEE International Conference on Computer Vision, 1999, 1033–1038, https://doi.org/10.1109/ICCV.1999.790383
[7] Gatys, L.A., Ecker, A.S., Bethge, M., *Image Style Transfer Using Convolutional Neural Networks*. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, 2414–2423, https://doi.org/10.1109/CVPR.2016.265
[8] Li, C., Wand, M., *Combining Markov Random Fields and Convolutional Neural Networks for Image Synthesis*. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, 2479–2486, https://doi.org/10.1109/ CVPR.2016.272
[9] Fayyaz, R.A., Maqbool, M., Hanif, M., *Textile Design Generation Using GANs*. In: 2020 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE), 2020, 1–5, https://doi.org/10.1109/CCECE47787.2020.9255674
[10] Jiang, S., Li, J., Fu, Y., *Deep Learning for Fashion Style Generation*, In: IEEE Transactions on Neural Networks and Learning Systems, 2021, 1–13, https://doi.org/10.1109/TNNLS.2021.3057892
[11] Wang, P., Li, Y., Vasconcelos, N., *Rethinking and Improving the Robustness of Image Style Transfer*. In: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2021, 124–133, https://doi.org/10.1109/ CVPR46437.2021.00019
[12] Simonyan, K., Zisserman, A., *Very Deep Convolutional Networks for Large-Scale Image Recognition*, In: The International Conference on Learning Representations (ICLR), 2015, abs/1409.1556, https://doi.org/10.48550/arXiv.1409.1556
[13] Zeiler, M.D., Fergus, R., *Visualizing and Understanding Convolutional Networks*. In: European Conference on Computer Vision (ECCV), 2014, 818–833, https://doi.org/10.1007/978-3-319-10590-1_53
[14] Jing, Y., Yang, Y., Feng, Z., Ye, J., Yu, Y., Song, M., *Neural Style Transfer: A Review*, In: IEEE Transactions on Visualization and Computer Graphics, 2020, 26, 11, 3365–3385, https://doi.org/10.1109/TVCG.2019.2921336
[15] Johnson, J., Alahi, A., Fei-Fei, L., *Perceptual Losses for Real-Time Style Transfer and Super-Resolution*. In: Computer Vision – ECCV 2016, 2016, 694–711, https://doi.org/10.1007/978-3-319-46475-6_43
[16] Deng, Y., Tang, F., Dong, W., Ma, C., Pan, X., Wang, L., Xu, C., *StyTr^2: Image Style Transfer with Transformers*. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2022, https://doi.org/10.48550/arXiv.2105.14576
[17] Champandard, A.J., *Semantic Style Transfer and Turning Two-Bit Doodles into Fine Artworks*, In: arXiv, 2016, https://doi.org/10.48550/arXiv.1603.01768
[18] Liu, S., Lin, T., He, D., Li, F., Wang, M., Li, X., Sun, Z., Li, Q., Ding, E., *AdaAttN: Revisit Attention Mechanism in Arbitrary Neural Style Transfer*. In: Proceedings of the IEEE International Conference on Computer Vision, 2021
[19] Zeng, X., Lu, Y., Tong, S., Xu, L., *Photorealism Style Transfer Combining MRFs-based and Gram-based Features*, In: 57, Journal of Nanjing University (Natural Science), 57, 1, 1–9, https://doi.org/10. 13232/j. cnki. jnju. 2021. 01. 001
[20] Denton, E., Chintala, S., Szlam, A., Fergus, R., *Deep Generative Image Models Using a Laplacian Pyramid of Adversarial Networks*, In: Advances in Conference on Neural Information Processing Systems(NIPS 2015), 2015, https://doi.org/10.48550/arXiv.1506.05751
[21] Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y., *Generative Adversarial Nets*. 2014, 2672–2680, https://doi.org/10.48550/arXiv.1406.2661
[22] Rudin, L I., Osher, S., Fatemi, E., *Nonlinear Total Variation Based Noise Removal Algorithms*, In: Physica D: Nonlinear Phenomena, 1992, 60, 1, 259–268, https://doi.org/10.1016/0167-2789(92)90242-F
[23] Risser, E., Wilmot, P., Barnes, C., *Stable and Controllable Neural Texture Synthesis and Style Transfer Using Histogram Losses*, In: arXiv, 2017, https://doi.org/10.48550/arXiv.1701.08893
[24] Lin, T., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C. L., *Microsoft COCO: Common Objects in Context*. In: Computer Vision – ECCV 2014, 2014, 740–755

Authors:

YUEXIN SUN, YU CHEN

Shanghai University of Engineering Science, School of Textiles and Fashion,
333 Longteng Road, Songjiang District, Shanghai, 201620, China
e-mail: m090120303@sues.edu.cn

Corresponding author:

YU CHEN
e-mail: ychen0918@sues.edu.cn