Dual Stage Augmented Colorful Texture Synthesis from Hand Sketch

Jinxuan Liu, Junyu Dong, Shu Zhang,
Tiange Zhang, Ying Gao
College of Information Science and Engineering
Ocean University of China
Qingdao, China
dongjunyu@ouc.edu.cn

Hui Yu
School of Creative Technologies
University of Portsmouth
Portsmouth, UK
hui.yu@port.ac.uk

***Abstract*—In this paper, we investigate the texture synthesis method generated from the hand-made sketches. In recent years, GANs have been vigorously studied in the field of image generation, yet the texture synthesis from the hand sketch has not been extensively studied. In order to enable the synthesized image not only possesses the texture features, but also the vibrant color, we propose a cascaded network model that generates a texture image through a dual-stage network. The proposed framework firstly generates a grayscale image with basic texture properties from hand sketch based on the conditional GANs. This grayscale texture is then colorized in the second stage. The network in the second stage is pre-trained using our constructed dataset to learn how to translate the grayscale image to a colorful image. This paper aims to design a new network to generate realistic texture from hand sketch. A series of experiments are conducted to validate the effectiveness of our method. Encouraging results are achieved. The experimental results demonstrate that our dual stage model outperforms the state-of-art generative models in the related areas.**

Keywords-texture synthesis; cGANs; image colorization

I. INTRODUCTION

Texture synthesis has always been an important field of computer vision and image processing. Every object has its own specific texture. These textures are universal and different. Some textures are regular, however, some textures are irregular and random. How do we synthesize texture? Synthetic texture refers to the artificially synthesized realistic texture images. They have three key characteristics: (1) The synthesized texture should be visually similar to the existing texture and can successfully fool humans; (2) The generated texture should be random, so-called creativity of model. (3) It should be in line with human aesthetic needs. Existing synthesizing methods can be achieved by texture sample mapping, method based on Markov random field or deep learning.

Most of the earlier researches on texture synthesis were achieved by an instance-based approach that synthesized new textures similar to existing textures. Kwatra et al. introduced a Graph-Cut method to minimize inconsistencies among the synthetic images. Encouraging results were obtained [1]. However, the computational time was high. As deep learning becomes popular, more researches try to adopt the great advantages from deep network into texture synthesis tasks. Specifically, the texture synthesis based on Convolutional Neural Networks (CNN) has been widely explored. Gatys et al. stated that besides the elegant texture production, the deep networks can also stylize an image with a single texture example [2].

Recently, the Generative Adversarial Network (GAN) presented by Goodfellow et al. allows the research of image manipulations to enter a deeper and wider domain [3]. GAN utilizes a loss to determine the true and false of the output image in term of real of fake. It uses this loss to train a generated model. In the training process, the goal of generator G in this model is to generate a realistic picture as much as possible to deceive the discriminator D, which tries to accurately distinguish the fake image generated by G from the real image. Accordingly, G and D constitute a dynamic "gaming process". Based on this concept, the introductions of various GAN networks inspire more new ideas for texture synthesis. A common method is the one based on the conditional GANs (cGANs). It adds additional condition information y to both the generator and the discriminator.

How to make the computer to generate a texture that we need? To address this problem, Phillip Isola et al. presented an approach to achieve image to image translation in pixel-level [4]. Yongli Lu et al. proposed a contextual GAN that used hand sketches to generate realistic images [5]. In our network, the shape and contour of textures are limited by hand sketches.

In this study, the texture synthesis is considered as an image-to-image translation process, which means the style transfer from sketch to texture. Considering that the color of texture images has a strong randomness. For example, some texture are colorful, such as flowers and clothes. And some texture are monotonous, such as trees and soil. By experiments, we found that the color has a great influence on the authenticity of synthetic texture. For example, a blue trunk cannot be real, and it is likely that the pink soil is fake. Although these strange objects may actually exist, they are extremely rare in practice. It is believed that this is because the color of texture is not regular. If there are more images of trees and leaves in the dataset, green and brown will account for a great proportion in the dataset. This problem causes the network to lose a lot of color information during the training process. Inspired by the idea of image-to-image translation, we propose a new method of dual stage augmented colorful texture synthesis from sketch. We divide the texture synthesis process into two parts: The

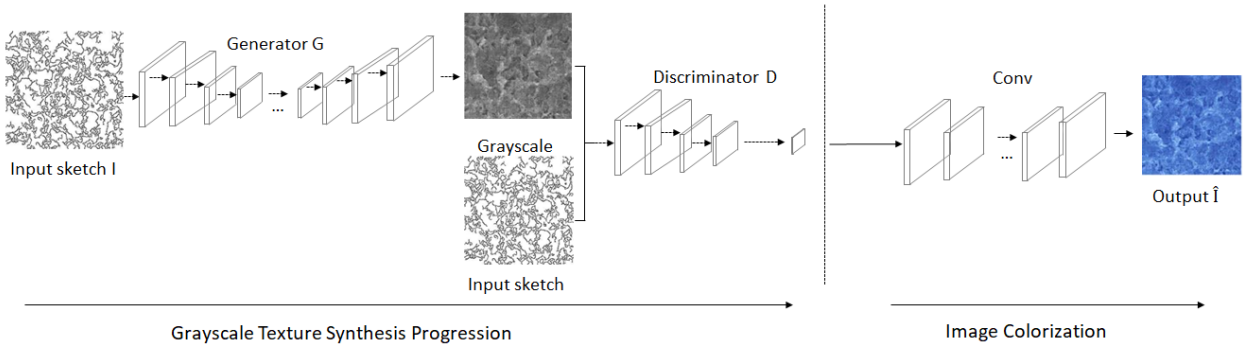

Fig. 1. Architecture of our approach. Our model designed to synthesize texture by dual stage. Grayscale texture synthesis progression tries to generate realistic grayscale, which showing texture pattern, roughness, etc. Image colorization tries to color the image credibly.

first part generates a grayscale texture image according to the given hand sketch. Rather than considering the color information, this part focuses on texture structure and detail of the synthesized image, and enforces the authenticity of the texture. In the second part, the proposed model colorizes the image generated by the first part. The goal of this step is not to produce an accurate color in the image same as the ground truth, but a realistic color as much as possible. It tries achieving not only creativity but also aesthetics. The two networks are trained independently, and then merged together. Experiments demonstrate the effectiveness and superiority of the proposed method

II. Related Work

Texture synthesis has been an active research topic since mid-1990s. In the field of computer graphics, any representations of a graphic surface can be treated as texture, including geometric information, material information, color information, etc. In general, the approaches of texture synthesis can be mainly divided into two categories as follows:

Nonparametric methods. There are two approaches to synthesize textures based on nonparametric strategy.

(1) Pixel-based models synthesize a new texture by resampling pixel of the original texture. A. Efros and T. K. Leung proposed a pixel-based method for texture synthesis with non-parametric sampling [6]. It only synthesized one texture pixel at a time, and gradually covered the entire texture image point by point. It is the earliest type of the texture synthesis algorithms. Additionally, Li Wei et al. also used a nonparametric method based on a tree-structured vector quantization strategy to synthesize texture [7].

(2) Patch-based models compare the similarities between patches. Compared to the pixel-based method, the patch-based methods synthesize the full image at once instead of one pixel a time [8]. For example, Kwatra et al. proposed to generate image and video using graph cuts [1]. It calculated the size and shape of the posted patch according to the graph-cut method.

However, those nonparametric methods are not able to establish an actual model to describe the natural textures. They only provide a mechanical process without changing its perceptual characteristics.

Parametric methods. These methods demonstrate better performance in synthesizing a wide range of textures. They are statistical measurements based on the filter responses rather than the image pixels [9]. For example, Gan et al. introduced a perception driven texture generation approach [10].

In recent years, deep convolutional neural networks have achieved great success in texture synthesis due to their powerful learning capabilities. Gatys et al. proposed a new parametric texture model, which was based on Convolutional Neural Network [2]. However, this method was instability, and its brightness and contrast could change drastically. Reference [11] presented a multi-scale synthesis pipeline based on Convolutional Neural Networks to address these issues. Based on the work of [2], Ulyanov et al. introduced a feed-forward convolution network to generate multiple samples of the same texture of any size. It transferred the given images into some images of artistic style [12].

After Generative Adversarial Networks were presented, many studies apply GANs to image synthesis [13-16]. For example, GANs were extended as cGANs for conditional image synthesis by Mirza and Osindero [17]. The image-conditional GANs have tackled the problem in product photo generation [18], image generation from sparse annotations [19], [20], digit and face image generation and many others. Wang and Gupta factorized the image generation process using a Style and Structure Generative Adversarial Network [21]. In [4], Isola et al. improved cGANs and demonstrated their wide applicability in image-to-image translation. However, the way of generating images from sketches still requires a huge database, from which images are retrieved [22], [23]. To address this problem, Lu et al. introduced contextual GAN for image generation from hand sketch [5]. Inspired by these works, we proposed a dual stage augmented colorful texture synthesis from sketch model.

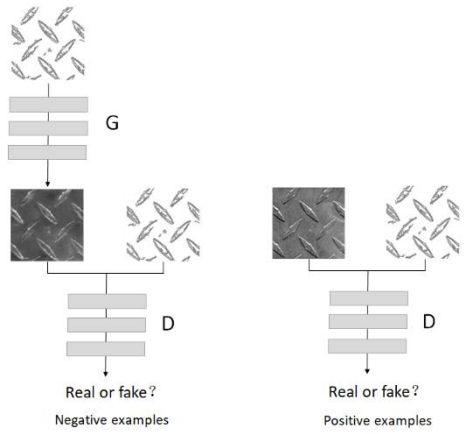

Fig. 2. Training a conditional GAN to map sketch→grayscale texture. The discriminator, D, learns to classify between fake (synthesized by the generator G) and real {sketch, grayscale} tuples. The generator, G, tries to synthesize fake images that fool D. Based on cGANs, both the generator and discriminator observe the input image.

III. METHODOLOGY

In this paper, we design a sketch-to-texture translation model as a solution to establish a texture synthesis problem model. The goal of our model is that, given a texture sketch I, to synthesize a new image $\hat{I}$, where $\hat{I}$ preserves the texture features described in I, such as shape, regularity and roughness, meanwhile has plausible color. The pipeline of the proposed model is shown in Fig. 1. The model consists of two stages: a grayscale texture synthesis progression and an image colorization step.

A. Grayscale Texture Synthesis Stage

In this stage, our goal is to generate a corresponding grayscale texture given a hand sketch. At this stage, we only focus on the preservation of the texture features ignoring other color factors. We train the network using the method proposed by Isola et al. The training process is demonstrated in Fig. 2. Traditional GANs are generative models that learn a mapping from a random noise vector z to an output image y, G: z→y [2]. cGANs are also generative models that learn a mapping from a observed image x and random a noise vector z to y, G: {x, z}→y. The objective function of a cGAN is as shown in (1).

$$L_{cGAN}(\mathrm{G},\mathrm{D}) = E_{x,y}[logD(x,y)] + E_{x,z}\left[\log\left(1 - D\left(x, G(x,z)\right)\right)\right] \quad (1)$$

For image translation tasks, the input and output actually share a lot of information. Therefore, in order to ensure the similarity between the input image and the output image, we train the network using an L1 loss:

$$L_{L1}(G) = E_{x,y,z}[\|y - G(x,z)\|_1] \quad (2)$$

The loss function of grayscale texture synthesis stage can now be written as:

$$L_G = \underset{G}{argmin}\underset{D}{max} L_{cGAN}(G,D) + \lambda L_{L1}(G) \quad (3)$$

Generator architecture. The traditional convolutional neural network will cause all the layers to hold all the information. An encoder-decoder, more specifically a "U-net" structure [24] is used to reduce errors. The network does not have any fully connected layers and only uses the valid part of each convolution [24]. The architecture is illustrated in Fig. 3.

Discriminator architecture. Since L1 loss can accurately capture the low frequencies, in this paper, we use PatchGAN [4], a discriminator architecture to enforce the high-frequency structure. The network cuts an image into different sizes of patches. The discriminator tries to distinguish whether each N*N patch in an image from real to fake. The average of all responses in an image is taken as the final output of discriminator.

B. Colorization Stage

In the second stage, the network is trained based on [25]. The reason we utilize this method to colorize the grayscale image is that instead of restoring the true color of the texture image, this method uses the texture, semantics, and other features of the input to predict possible colorization results. The final image is plausible and realistic. This not only reduces the difficulty of colorization, but also satisfies people's aesthetic standards. To address the problem of automatic colorization, [25] designed an appropriate objective function to handle the multi-model uncertainty of the colorization problem and captured multiple colors. Unlike other CNN architectures, the proposed method uses a classification loss, with rebalanced rare classes, and a VGG-styled network with added depth and dilated convolutions [13], [26]. The network architecture is shown in Fig. 4.

C. Training details

In the first stage, to optimize the proposed networks, we follow the training procedure proposed in [4]. We divide the objective by 2 while optimizing D, which slows down the rate at which D learns relative to G. In our experiments, we use minibatch SGD and apply the Adam solver [27] to the training process with a learning rate of 0.0002. The batch size is set to 1. The size of input image is 256*256. In the second stage, we pre-trained our networks with our dataset. The entire training procession takes approximately 8 - 10 hours.

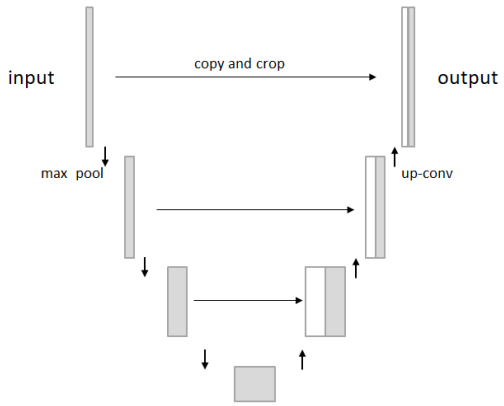

Fig. 3. U-net architecture. The "U-net" is an encoder-decoder with skip connection between mirrored layers in the encoder and decoder stacks.

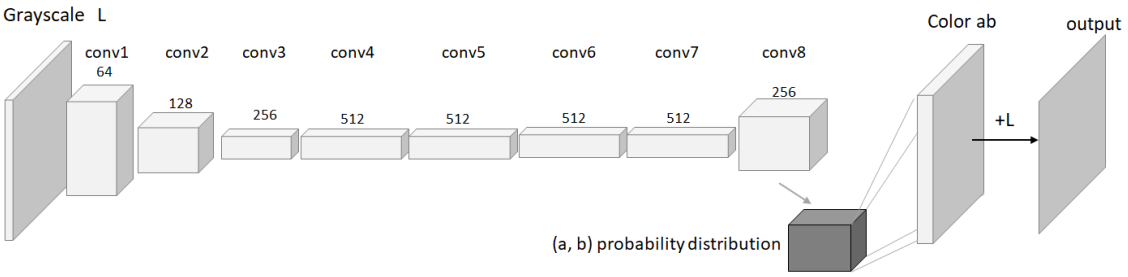

Fig. 4. The network of image colorization.

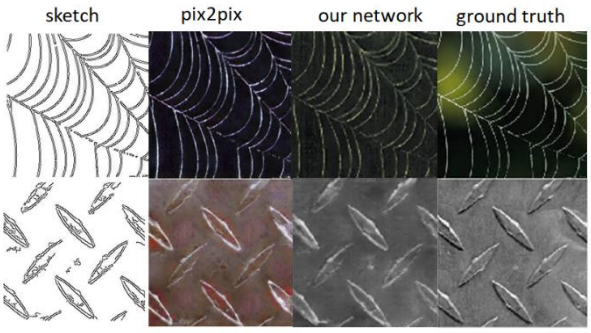

Fig. 5. Comparing experiments on "HED" and "Canny". The first and third columns are the result of Canny, and the second and fourth columns are the result of the HED. The second row is the generated images from texture synthesis progressing.

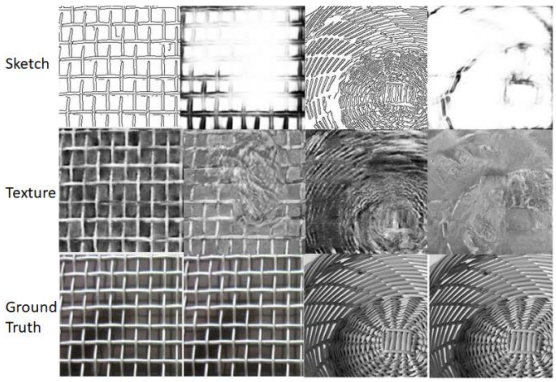

Fig. 6. Example results of our method compared to ground truth.

IV. EXPERIMENTS

A. DataSets

The natural texture images we used in our experiments are based on Describable Textures Dataset (DTD). On DTD, there are 47 groups containing 5640 texture images. In order to further eliminate the interference of color factors in the first stage of training, we convert these 5640 pictures into grayscale images with a resolution of 256*256. The key contribution of our network's training is that it has paired data of sketches to textures maps. However, the workload of collecting thousands of hand-drawn images is too much. Therefore, we use HED edge detector and Canny edge detector to extract hand-drawn sketches instead of manual annotations. Through experimental comparisons, the latter results can fit our experimental needs better. We thus finally used the results of Canny edge detector as training data. For each run of experiment, nearly 3900 images are utilized for training and the rest for testing.

To verify the applicability and effectiveness of our network, we collected two sets of real hand drawings. Each set contains around 470 hand drawings, corresponding to 470 texture maps. These images are also used for training and testing.

B. Creation of training data

In order to implement our sketch to texture synthesis, we use two mainstream algorithms to perform edge extraction on the original texture images. The comparisons of the training results of these two data sets as shown in Fig. 5. Differing to the training process used in pix2pix, we utilize sketches extracted by Canny edge detector in the training.

C. Comparison with state-of-the-art methods

We compare our approach against the methods for image-to-image translation [4] on our dataset. The comparative experiments results are shown in Fig. 6. The Structural Similarity Metric (SSIM) is used to measure the similarity between generated image and ground truth. Results are shown in TABLE Ⅰ.

We randomly select one hundred pairs of ground truth and synthesized textures to evaluate. The participants were asked to rate the images with a score of 10 points. The higher the score is, the more closer to ground truth it is. We evaluated these images in color, structure, and authenticity. The scores are shown in TABLE Ⅱ. The experimental results proved that our network can generate greatly realistic texture image. It is superior to pix2pix. More importantly, our approach has made great strides in enhancing the color of textures compared to pix2pix. More experimental results on our network are shown in Fig. 7.

TABLE I. SSIM ON TEST SET

Method	***Pix2pix***	***Ours***
SSIM	0.6494	0.8584

TABLE II. EVALUATION RESULTS

	Color	**Structure**	**Authenticity**
pix2pix	8.03	9.40	8.89
Ours	9.21	9.58	9.36

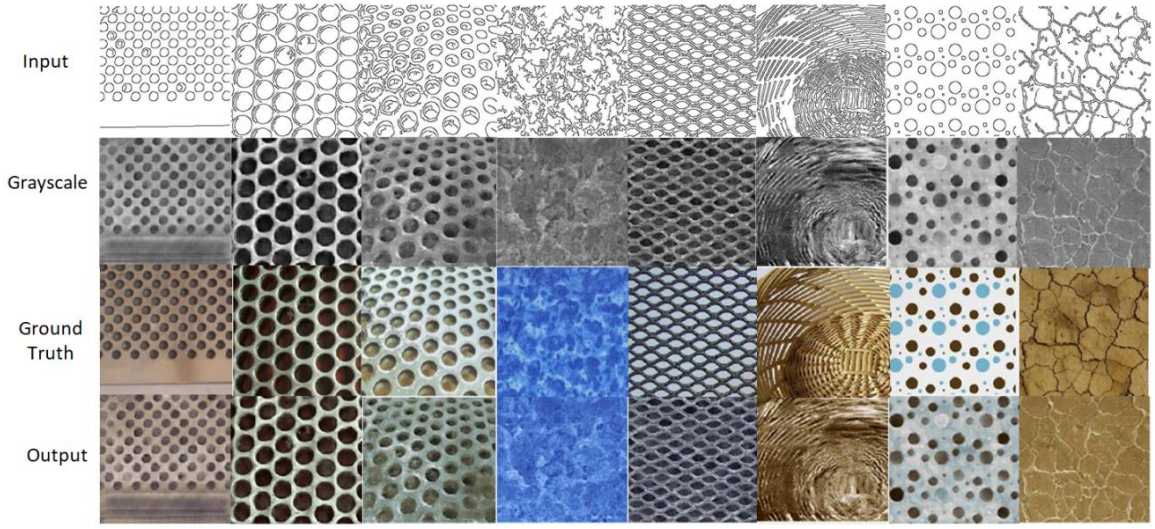

Fig. 7. From top to bottom: Input sketch, image of texture synthesis progression, ground truth, Output of our network.

D. Results on real hand sketch

In order to verify the practicality of our method, we conduct the experiments on the real hand sketch. Results are shown in Fig. 8. The experiments demonstrate that our method can successfully synthesize textures based on the real hand sketch. From the results, it can be noted that our method not only captures the important details from the input sketch, but also exhibits some degree of freedom in the appearance.

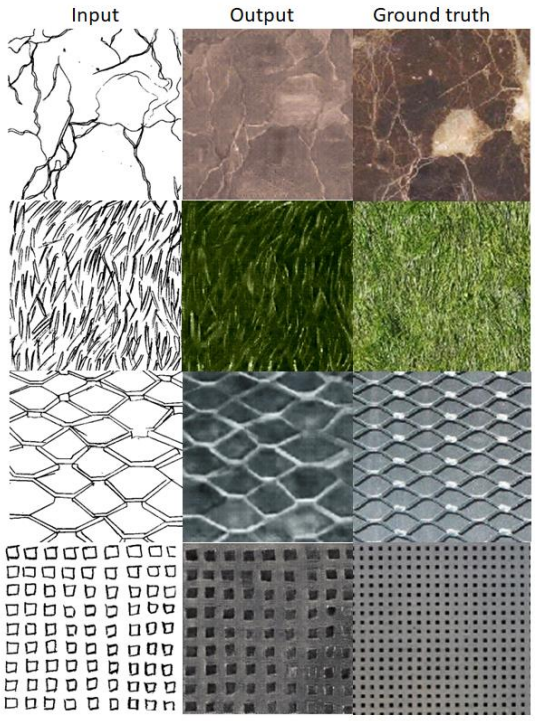

Fig. 8. Results on real hand sketch dataset.

CONCLUSION

In this paper, we propose a dual stage deep learning model for hand sketch to colorful texture synthesis problem. Compared to the previous methods, our approach is more robust. The experimental results demonstrate that compared to one-stage model, the synthesized texture images of dual stage are more realistic and closer to ground truth.

REFERENCES

[1] V. Kwatra, A. Schödl, and I. Essa, "Graphcut textures: image and video synthesis using graph cuts," ACM Trans. , 2003.

[2] L. A. Gatys, A. S. Ecker, and M. Bethge, "Texture Synthesis Using Convolutional Neural Networks," pp. 110, 2015.

[3] I. Goodfellow et al., "Generative Adversarial Nets (NIPS version)," Adv. Neural Inf. Process. Syst. 27, 2014.

[4] P. Isola, J. Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, 2017.

[5] Y. Lu, S. Wu, Y. W. Tai, and C. K. Tang, "Image Generation from Sketch Constraint Using Contextual GAN," in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2018.

[6] A. A. Efros and T. K. Leung, "Texture synthesis by non-parametric sampling," in Proceedings of the Seventh IEEE International Conference on Computer Vision, 1999.

[7] L. Y. Wei and M. Levoy, "Fast texture synthesis using tree-structured vector quantization," 2005.

[8] A. A. Efros and W. T. Freeman, "Image quilting for texture synthesis and transfer," 2005.

[9] J. Portilla and E. P. Simoncelli, "Parametric texture model based on joint statistics of complex wavelet coefficients," Int. J. Comput. Vis., 2000.

[10] Y. Gan, "Perception Driven Texture Generation," pp. 889–894, 2017.

[11] E. Risser, P. Wilmot, and C. Barnes, "Stable and Controllable Neural Texture Synthesis and Style Transfer Using Histogram Losses," 2017.

[12] D. Ulyanov, V. Lebedev, A. Vedaldi, and V. Lempitsky, "Texture Networks: Feed-forward Synthesis of Textures and Stylized Images,"2016.

[13] L. C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, “DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs,”IEEE Trans. Pattern Anal. Mach. Intell., vol. 40, no. 4, pp. 834–848, 2018.

[14] A. Radford, L. Metz, and S. Chintala, “Unsupervised Representation learning with Deep Convolutional GANs,” Int. Conf. Learn. Represent., 2016.

[15] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, “Improved Techniques for Training GANs,” pp. 1–10, 2016.

[16] J. Zhao, M. Mathieu, and Y. LeCun, “Energy-based Generative Adversarial Network,” no. 2014, pp. 1–17, 2016

[17] M. Mirza and S. Osindero, “Conditional Generative Adversarial Nets,” pp. 1–7, 2014.

[18] D. Yoo, N. Kim, S. Park, A. S. Paek, and I. S. Kweon, “Pixel-level domain transfer,” in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2016.

[19] S. Reed, Z. Akata, S. Mohan, S. Tenka, B. Schiele, and H. Lee, “Learning What and Where to Draw,” no. Nips, pp. 1–9, 2016.

[20] L. Karacan, Z. Akata, A. Erdem, and E. Erdem, “Learning to Generate Images of Outdoor Scenes from Attributes and Semantic Layouts,” 2016.

[21] X. Wang and A. Gupta, “Generative image modeling using style and structure adversarial networks,” in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2016.

[22] Q. Yu, F. Liu, Y. Z. Song, T. Xiang, T. M. Hospedales, and C. C. Loy, “Sketch Me That Shoe,” 2016.

[23] P. Sangkloy, N. Burnell, C. Ham, and J. Hays, “The Sketchy Database: Learning to Retrieve Badly Drawn Bunnies,” ACM Trans. Graph., 2016.

[24] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 9351, pp. 234–241, 2015.

[25] R. Zhang, P. Isola, and A. A. Efros, “Colorful image colorization,” in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2016.

[26] F. Yu and V. Koltun, “Multi-Scale Context Aggregation by Dilated Convolutions,” 2015.

[27] D. P. Kingma and J. Ba, “Adam: A Method for Stochastic Optimization,” pp. 1–15, 2014.