A Deep Learning Based Interactive Sketching System for Fashion Images Design

Y. Li¹, X. G. Yu², X. G. Han¹, N. J. Jiang³, K. Jia⁴ and J. B. Lu³

¹DexForce Technology Co. Ltd.
²Chinese University of Hong Kong (Shenzhen)
³SmartMore Corporation
⁴South China University of Technology

Abstract
In this work, we propose an interactive system to design diverse high-quality garment images from fashion sketches and the texture information. The major challenge behind this system is to generate high-quality and detailed texture according to the user-provided texture information. Prior works mainly use the texture patch representation and try to map a small texture patch to a whole garment image, hence unable to generate high-quality details. In contrast, inspired by intrinsic image decomposition, we decompose this task into texture synthesis and shading enhancement. In particular, we propose a novel bi-colored edge texture representation to synthesize textured garment images and a shading enhancer to render shading based on the grayscale edges. The bi-colored edge representation provides simple but effective texture cues and color constraints, so that the details can be better reconstructed. Moreover, with the rendered shading, the synthesized garment image becomes more vivid.

CCS Concepts
• Networks → Network reliability; • Computing methodologies → Computer vision;

Figure 1: The interactive pipeline to use our sketching system for fashion images design. The user can firstly draw contours to determine the shape of the desired garment. Three different modes are then provided for specifying different textures on the garment, i.e., pure colored, sparse textured and dense textured. A generative adversarial network is adopted here to convert these users’ inputs to realistic photos. Thirdly, some shading edges are allowed to be drawn which will also be converted into shading information with deep learning methods, making the garment image enhanced to be more stereoscopic. As post-processing, all the colors can be further modified straightforwardly.

† Corresponding author: Xiaoguang Han, hanxiaoguang@cuhk.edu.cn
‡ Kui Jia is with the School of Electronic and Information Engineering, South China University of Technology, Guangzhou, China, the Pazhou Lab, Guangzhou, 510335, China, and the Peng Cheng Lab, Shenzhen 518005, China

arXiv:2010.04413v1 [cs.CV] 9 Oct 2020.

© 2020 The Author(s)
Eurographics Proceedings © 2020 The Eurographics Association.
1. Introduction

In recent years, a growing number of customers are looking for a more exclusive personalized customizing, especially garment customization. The key technique of garment customization is to synthesize a corresponding cloth image based on user-provided design elements like contours, fabric patterns and so on. Existing garment-drawing software often requires professional experiences and the drawing process is also time-consuming. It is arduous for a novice user to design a garment using common software. Therefore, it is meaningful to design a user-friendly interactive tool to meet users’ demands while alleviating their workloads. With the invention of conditional generative adversarial networks (cGANs) in [IZZE17], we can formulate garment-drawing as an image-to-image translation problem, which can be solved by the Pix2pix [IZZE17, WLZ*18] framework.

Based on Pix2pix [IZZE17, WLZ*18] framework, a few methods have been proposed for garment image generation. Xian et al. [XSA*18] introduced a textureGAN framework to generate garment images from an input contour map and a user-provided texture patch. They propose a new loss function to guide the network to propagate the texture patch to the corresponding regions of the given contour map, but its results require the correct position and proper size of the texture patch. To generate a variety of texture pattern outputs, similar to BicycleGAN [ZPIE17], FashionGAN [CLGS18] encodes the color and material information into a latent vector through an encoder network, rather than directly taking texture patch as the input. However, prior methods suffer from two drawbacks: 1) only a small texture patch is propagated to a whole garment image under the constraint of the contour map. Although this input setting requires less efforts for users, it is difficult to generate a high-quality garment image containing wrinkles and shading solely from a small-size texture pattern; 2) the contour map with the texture patch representation are not sufficient to generate fashion garment with more designing details, e.g. garment decoration and logos. An intuitive solution for this is to represent the detailed texture patterns with binary edges, and combined with several color points to synthesize a garment image. But as Dekel et al. [DGK*18] demonstrated it is insufficient to generate a high quality image from a binary edge map. Moreover, we empirically find that this method often leads to blurry results at the edge areas.

Therefore, based on the above discussions, we decompose this task into texture synthesis and shading enhancement. Particularly, we propose a bi-colored edge texture representation, that is, extract the color points at both sides of texture edges as our texture information to synthesize textured garment images, as shown in Figure 2. The bi-colored edge representation provides simple but effective texture cues and color constraints, so that the details can be better reconstructed. We also introduce a shading enhancer to render shading on a garment image by simply drawing some sparse binary edges. In this way, the user can achieve more control on the shading information of a garment image. Even though the bi-colored edge representation will increase the demand of users’ interactions compared with the patch representation, our experiments demonstrate this representation has sufficient capability to generate diverse high-quality garment images. We also design a handy interactive system based on this representation. It allows free-form design such as garment decoration, logo and etc.

Figure 2: An illustration of the bi-colored edge representation. We extract the color point at both sides of the texture edges as our texture information.

Our major contributions in this work can be summarized as follows:

- We propose an elaborate interactive-system for fashion images design. It is convenient for users to customize personalized fashion garments.
- We decompose the task of garment images synthesis into texture synthesis and shading enhancement, and propose the bi-colored edge representation which reaches a good balance between the quality of the synthesized images and the workload of users.
- We propose a shading enhancer for interactively drawing grayscale edges and separately inferring the shading information. This greatly facilitates the photo-realism of the results.

2. Methodology

Our framework includes two generators, namely garment generator and shading generator. The former generates color images based on the fashion contours and the proposed bi-colored edge representation. The latter one renders shading based on the shading edges. And then an interactive system is proposed for generating high-quality fashion garment images based on the aforementioned two generators as shown in Figure 1.

2.1. Image synthesis from sketches

In this subsection, we first introduce how to train an image generator to synthesize high-resolution, colored clothing image, and train a shading generator to improve the stereoscopic shape of the synthesized garment image. The framework is shown in Figure 3.

2.1.1. Fashion image generator

Our fashion image generator network is based on the architecture proposed by Johnson et al. [JAFF16], which is a deep residual convolutional neural network. It consists of three components: a convolutional layer to downsample the input, a set of residual blocks [HZRS16, GW16] and a convolutional layer to upsample. The network downsampling and then upsampling allow the network to have an effective receptive field size on the input, making the generated image more realistic. The input of our fashion image generator is a tensor of size $512 \times 512 \times 4$: a binary sketch that describes the structure of a garment image, a RGB bi-colored edges map.
To enhance the performance of the generator, we adopt the framework of cGAN [IZZE17] which has been used intensively for image-to-image translation tasks. Thus both the contour map and the bi-colored edges are inputted into the discriminator as a condition to guide the generator to synthesize more realistic images. In addition, to synthesize high-resolution garment images, we adopt a multi-scale discriminator framework similar to Wang et al. [WLZ18]. Those discriminators are trained to distinguish real and synthesized image at different scales, respectively. At each scale, the discriminator has different receptive fields on the input.

In our experiment, we apply lsGAN [MLX17] to train the generator and the discriminator. The adversarial loss of the discriminator is written as:

$$L_{adv}(D) = \sum_{i=1}^{S} \frac{1}{2} \mathbb{E}_{(x,y) \sim p_{data}(x,y)} \left[D_i(x,y) - 1 \right]^2$$

(1)

$$+ \frac{1}{2} \mathbb{E}_{x \sim p_{data}(x)} \left[D_i(x,G(x))^2 \right],$$

where the $S$ is the number of discriminators in the network architecture. Also, the adversarial loss of generator is:

$$L_{adv}(G) = \mathbb{E}_{x \sim p_{data}(x)} \left[D_i(x,G(x)) - 1 \right]^2.$$  

(2)

$L1$ loss is considered to ensure the generated image $G(x)$ from the input $x$ to be close to its ground truth image $y$. It is given by:

$$L_{L1}(G) = \mathbb{E}_{(x,y) \sim p_{data}(x,y)} \left[\|y - G(x)\|_1 \right].$$

(3)

We also adopt a perception loss to increase the quality of the generated image. The perceptual loss measures high-level perceptual and semantic differences between two images by projecting images into feature spaces using VGG-19 [SZ14] which is pre-trained on the ImageNet.

$$L_p(G) = \sum_{i=1}^{N} \frac{1}{M_i} \left[\|\Theta^{(i)}(y) - \Theta^{(i)}(G(x))\|_1 \right].$$

(4)

Here, $\Theta^{(i)}(\cdot)$ is the feature map with $M_i$ elements, as the output from the $i$-th layer of VGG network.

Inspired by the color unmixing algorithm [AAPS16], we introduced a KL loss function to restrain the color distribution of the output to be consistent with the input’s. The color constraint loss function is defined as:

$$L_{kl}(G) = \sum_{i=1}^{K} \frac{1}{2} \left( \log \frac{\sum_{t}^{(i)} S_t}{\sum_{t}^{(i)} S_t} - n + tr(S_t^{-1} S_t) \right)$$

$$+ (\mu_y - \mu_x)^T S_t^{-1} (\mu_y - \mu_x).$$

(5)

Here, we denote the output of the generator $G(x)$ as $y$. We use the hierarchical clustering algorithm to obtain a set of color clusters and $K$ is the number of color cluster. $\mu^{(i)}$ and $\Sigma^{(i)}$ are the mean vector and the covariance matrix of the $i$-th color cluster calculated by using the cluster label.

The total generator loss function is written as:

$$L = \lambda_{adv} L_{adv} + \lambda_{L1} L_{L1} + \lambda_p L_p + \lambda_{kl} L_{kl}.$$  

(6)

2.1.2. Shading enhancement

The image generator can synthesize detailed texture patterns, but the results are likely to lack shading information. Thus, we introduce a shading generator to synthesize shading information from a contour map and sparse binary edges. Particularly, we are not directly to synthesize the shading-corrected image, but to synthesize a shading image which is a single channel image. We use the notion of intrinsic image decomposition [BHY15, LS18] to obtain the shading image. That is, a RGB image $I \in \mathbb{R}^{H \times W \times 3}$ can be factorized into a product of a reflectance image $R \in \mathbb{R}^{H \times W \times 3}$ and a shading image $S \in \mathbb{R}^{H \times W}$. $I = R \times S$. With that, the synthesized garment image by fashion image generator is regarded as $\tilde{R}$, the result of the shading generator is $\tilde{S}$, and the final result of our system is $\tilde{S} \times \tilde{R}$. The network architecture is the same as the garment generator, except that we change the last activation layer from Tanh function to ReLU. Since the pixel value range of the shading image is relatively discrete, it is unwise to conduct zero-centered operation on it.

The loss function to train the shading generator is shown as below:

$$L = \lambda_{rec} L_{rec} + \lambda_{dense} L_{dense}.$$  

(7)

The reconstruction loss $L_{rec}$ computes the distance between the ground truth image $I = S \times R$ and the synthesized image $\tilde{S} \times \tilde{R}$ in RGB domain.

$$L_{rec} = E_{(S,I) \sim p_{data}(S,I)} \left[\|I - S \times R\|_1 \right].$$

(8)

As the shading information of a garment image is sparse, and the final result $\tilde{S} \times \tilde{R}$ should be similar with $R$, that is most of the pixels value in synthesized shading image $\tilde{S}$ should close to 1. Thus we conduct the dense loss:

$$L_{dense} = E_{(S,I) \sim p_{data}(S,I)} \left[\|\tilde{S} - 1\|_1 \right].$$

(9)

2.2. Dataset collection

2.2.1. Clothing images collection

In this work, we focus on high-resolution garment fashion design. The garment image should have pure background colours as well as diverse fashion styles. Although the public large-scale garment dataset like DeepFashion [LLQ16] contains a large number of garment images including various textures, most of the clothing styles...
are T-shirt and lack rich fashion styles. We therefore collect a fashion garment dataset which have clear background, high-resolution images as well as diverse styles. The dataset contains top clothing images, skirts and coats around 4,300, of which 3900 are used as the training set and the remaining are used as the validation set. We set aside the pants category as their styles are usually relatively simple. The resolution of image in our dataset is 512×512.

2.2.2. Training samples preparing

To train the garment generator and the shading generator, the training samples include a contour map, texture information and shading edges. The contour map is applied in image generator as well as shading generator. We apply the commonly used edge detector, HED edge detector [XT15], to obtain edges as our contour map and then simplify those edges with an additional post-processing method provided by pix2pix [IZZE17].

For image generator, the texture edge information is obtained by extracting the color information at both sides of the texture edges. Specifically, we first extract the texture edges by Canny edge detector [Can87] and remove the outermost edges to make sure the shape of the generated image is constrained by the contour map. Taking the convenience of interaction into account, we also remove the corner points. Then the texture edge information is extracted by computing the orientation of the texture edges and sample the color points at both sides of a texture edge for each edge pixel. An observation is that the extracted texture edge information is in the form of double-lines, thus we can draw the information by using various types of brush during the test phase as shown in Figure 1.

For shading generator, we need a pair of training sample (shading edge, shading image). Based on the notion of intrinsic image decomposition, We select about 2000 garment images which include large areas of pure colors, and compute the mean RGB value of the largest area and extend it to corresponding areas as the reflectance image R. The shading image is obtained by $S = I / R$. The texture edges extracted by Canny detector is regarded as our shading edges, and we only retain the largest area of a garment image by means of color clusters.

2.3. User interface

As illustrated in Figure 1, the UI items include a contour map, texture information and an optional item, shading edges.

**Drawing contours** The first step is to determine the overall shape of a fashion garment. This can be achieved by obtaining a contour map either from user’s interaction or automatically extracted from a garment image by our system. For the former input setting, a simple outline sketch is sufficient. Moreover, since the contour map is grayscale, it is easy for user to draw or manipulate them in our interactive system.

**Specifying texture** For allowing users to better provide texture information in our interactive system, we design three kinds of interactive tools to meet diverse demands. We also classify texture pattern into three categories based on how difficult to draw them, which include pure color texture pattern, sparse texture pattern and dense texture pattern. For pure color texture pattern, the user only need to provide a few color points to control the color of the synthesized garment image. For sparse texture pattern, such as stripes and garment decoration, our system supports using a variety of brushes to offer diverse styles of design. Those brushes are used to draw the bi-colored edges, and the colors of both sides are selected by a user. We illustrate two types of sparse texture pattern (2 strings and 4 strings) in Figure 1. The first type is used to draw a variety of texture shapes and the second type is able to draw the pinstripe texture. For dense texture pattern, such as floral and tartan which often include recurrent patterns, the user is required to provide a small texture patch, and our interactive system leverages the patch match algorithm to obtain a complete texture edges by enlarging this patch to any size we want, and then extract the bi-colored edges on it. This is different from TextureGAN [XSA'18] and FashionGAN [CLGS18] whose intention are to implicitly learn the Patch-Match algorithm by using the neural network.

**Adding shading** The user can also augment shading on the synthesized garment image by drawing grayscale shading edges. The shading information is garment-dependent, as different shapes have different shading. Thus it should be used cooperatively with the contour map to synthesize factual shading information, and it should be occurred in the proper area, such as the armhole of a T-shirt or the hem of a dress. The shading edges may overlap with contour edges, but it is not contradictory since the contour edges is used to synthesize the shape of a garment, while the shading edges is applied to enhance the shading.

**Changing color** Given that the abundant color of the garments, our system also supports to change the color of the synthesized garment image. Specially, for pure color texture pattern, it is easy to directly change the color of the previous given color points. For sparse texture pattern, to avoid redraw the bi-colored edges, we can select a color used in previous stage and then replace this color with another color. For a dense texture pattern, we can correctly classify the pattern’s color by using the K-means algorithm, in which the number of color clusters is decided on line, and then to change the color of each cluster. Finally, we can re-extract the bi-colored edges on it and to synthesize garment images.

Our system consists of three interfaces, namely drawing inter-
face, color regulator and output interface. The drawing interface includes three layers, the garment contour layer, the bi-colored texture edge layer and the shading edge layer. The garment contour edge and shading edge are both grayscale, it is easy for user to draw or manipulate them in our interactive system. With the designed interactive tools, it is also convenient to draw bi-colored edge. To change the color of the synthesized garment image, we collect the occur color on the bi-colored edge layer and show those color in the right-top corner. We only need to change the color of this corner, and the corresponding color in bi-colored texture edge layer will be changed.

3. Experiments

3.1. Implementation details

To train the garment generator, we employ two discriminators to distinguish real and fake images at two image scales. Our algorithm is implemented in PyTorch and trained on two GTX 1080 GPUs for 150 epochs using the ADAM optimizer with $\text{lr} = 0.0002$ and $\text{beta1} = 0.5$, $\text{beta2} = 0.999$, consuming two days. For the garment generator, we set different weights for the losses, where $\lambda_{\text{adv}} = 1, \lambda_{\text{L1}} = 10, \lambda_{\text{p}} = 10, \lambda_{\text{KL}} = 0.01$ in Eq. (6). For the shading generator, we set $\lambda_{\text{rec}} = 100, \lambda_{\text{dense}} = 1$ in Eq. (7).

3.2. Results and analysis

In this subsection, we firstly show more qualitative results designed by our sketching system. Then we also conduct quantitative and qualitative evaluations on the proposed bi-colored edge representation with some other variants.

3.2.1. Results gallery

As shown in Figure 4, our interactive system can design diverse and colorful garment. With the flexibility of the bi-colored edge representation and the shading edge representation to increase the photo-realism of the results, we can draw various texture patterns and synthesize high-quality garment image. Although drawing is required, our interactive system provides very convenient drawing tools and is suitable for any synthesized resolution.

3.2.2. Comparisons on different user inputs

One of the key contributions in this work is the proposed new texture representation, that is the bi-colored edge. So we compare the results of different texture representation. Compared with our bi-colored edge representation, the two other variants can be used as the inputs: 1) use binary edges to determine texture information and sampled color points to specify colors; 2) directly use a texture patch without a PatchMatch method, say the method of texture-GAN [XSA18]. For training the first network, we sample a small area of size from $1 \times 1$ to $9 \times 9$ on the real image and compute the mean color of this small area to obtain the sparse color points in the training phase. The number of color points is from 50 to 100. For training the second network, a garment patch of a random size from $50 \times 50$ to $70 \times 70$ is sampled from a real garment image, and the patch location is mainly chosen in the center regions of a garment.

Figure 5 shows the reconstruction results of the three kinds user input. The groundtruth garment images are selected from our dataset which are not used during the training procedure. Our bi-colored edges representation can synthesize more unambiguous texture patterns compared with the input of the binary edges and several color points. Although a local texture loss is considered [XSA18], the texture patch representation based method is still hard to synthesize complex texture patterns. On the contrary, our bi-colored edges representation can synthesize high-quality garment image and accord with the texture pattern given by user.

We then use Inception Score (IS) and Frechet Inception Distance (FID) [HRU17] evaluation metrics which are widely used...
in generative models to evaluate the performance of the two different representations. We also conduct a user study to assess the realism of the generated images and compare the performance of our proposed bi-colored edge representation with two other texture representations. 10 participants are shown 50 synthesized image pairs, and then grade each pair images in authenticity. The highest score is 3 and the lowest is 1. Table 1 proves that our bi-colored edge representation is valid. And our proposed color constraint loss function is improved in both evaluation metrics, which indicates that it can increase the diversity and quality of the generated images.

3.2.3. Ablation study on shading enhancement

Figure 6 shows the qualitative evaluations about our shading enhancement. It is obviously that the shading enhancement garment images become more vivid. We also performed a user study for evaluating the visual quality of shading enhancement. 10 participants were shown 30 garment image pairs with and without shading enhancement, and then are asked to which image is more photorealism. It will be scored if the participant answers correctly within a second, otherwise it will not be scored. The final average score is 26/30. This result also demonstrates that our shading enhancer can effectively render the shading on a garment image, so as to improve the realism of results.

4. Conclusion

In this work, we decomposed the task of garment images synthesis into texture synthesis and shading enhancement and proposed an interactive sketching system for fashion image synthesis. The system allows users to draw contours, specifying diverse texture information and performing shading enhancement by sketching shading edges. Deep learning based methods are applied to convert these sparse information to realistic photos. To well balance the output quality and interaction workload, a novel bi-colored edge representation is proposed. Experiments demonstrate the superiority of such representation against all others.

5. Acknowledgements

The work was supported in part by the Key Area R&D Program of Guangdong Province with grant No. 2018B030338001, by the National Key R&D Program of China with grant No. 2018YFB1800800, by Natural Science Foundation of China with grant NSF-61902334 and NSF-61629101, by Guangdong Zhujiang Project No. 2017ZT07X152, and by Shenzhen Key Lab Fund No. ZDSYS201707251409055, by the Program for Guangdong Introducing Innovative and Entrepreneurial Teams (Grant No.: 2017ZT07X183), the National Natural Science Foundation of China (Grant No.: 61771201), and the Guangdong R&D key project of China (Grant No. 2019B010155001), by the Shenzhen Cloudream Technology Co. Ltd.†

References

[AAPS16] Aksoy Y., Aydin T. O., Pollefeys M., Smailić A.: Interactive high-quality green-screen keying via color unmixing. ACM Transactions on Graphics (TOG) 36, 4 (2016), 1–3

† This work was mainly done when Yao, Nianjuan and Jiangbo were interning and working in Cloudream Technology.

[BYHY15] Bi S., Han X., Yu Y.: An 11 image transform for edge-preserving smoothing and scene-level intrinsic decomposition. ACM Transactions on Graphics (TOG) 34, 4 (2015), 78–3

[BSFG09] Barnes C., Shechtman E., Finkelstein A., Goldman D. B.: Patchmatch: A randomized correspondence algorithm for structural image editing. In ACM Transactions on Graphics (TOG) (2009), vol. 28, ACM, p. 24.

[Canny J.]: A computational approach to edge detection. In Readings in computer vision. Elsevier, 1987, pp. 184–203. 4

[CLGS18] Cui Y. R., Liu Q., Gao C. Y., Su Z.: Fashiongan: Display your fashion design using conditional generative adversarial nets. In Computer Graphics Forum (2018), vol. 37, Wiley Online Library, pp. 109–119. 2, 4

[DGK*18] Dekel T., Gan C., Krishnan D., Liu C., Freeman W. T.: Sparse, smart contours to represent and edit images. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2018), pp. 3511–3520. 2

[GW16] Gross S., Wilber M.: Training and investigating residual nets. Facebook AI Research (2016). 2

[HRR*17] Heusel M., Ramsauer H., Unterthiner T., Nessler B., Hochreiter S.: Gans trained by a two time-scale update rule converge to a local Nash equilibrium. In Advances in Neural Information Processing Systems (2017), pp. 6626–6637. 5

[HRS*16] He K., Zhang X., Ren S., Sun J.: Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (2016), pp. 770–778. 2

[IZZ*17] Isola P., Zhu J.-Y., Zhou T., Efros A. A.: Image-to-image translation with conditional adversarial networks. In Proceedings of the IEEE conference on computer vision and pattern recognition (2017), pp. 1125–1134. 2, 3, 4

[JAFFF16] Johnson J., Alahi A., Fei-Fei L.: Perceptual losses for real-time style transfer and super-resolution. In European conference on computer vision (2016), Springer, pp. 694–711. 2

[LQ*16] Liu Z., Luo P., Qu S., Wang X., Tang X.: Deepfashion: Powering robust clothes recognition and retrieval with rich annotations. In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2016). 3

[LS18] Li Z., Snavely N.: Learning intrinsic image decomposition from watching the world. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2018), pp. 9039–9048. 3

[MLX*17] Mao X., Li Q., Xie H., Lau R. Y., Wang Z., Paul Smolley S.: Least squares generative adversarial networks. In Proceedings of the IEEE International Conference on Computer Vision (2017), pp. 2794–2802. 3

[SZ14] Simonyan K., Zisserman A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014). 3

[WLZ*18] Wang T.-C., Liu M.-Y., Zhu J.-Y., Tao A., Kautz J., Catanzaro B.: High-resolution image synthesis and semantic manipulation with conditional GANs. In Proceedings of the IEEE conference on computer vision and pattern recognition (2018), pp. 8798–8807. 2, 3

[XSA*18] Xian W., Sangkloy P., Agrawal V., Raj A., Lu J., Fang C., Yu F., Hays J.: Texturegan: Controlling deep image synthesis with texture patches. In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (2018), pp. 8456–8465. 2, 4. 5

[XT15] Xie S., Tu Z.: Holistically-nested edge detection. In Proceedings of the IEEE international conference on computer vision (2015), pp. 1395–1403. 4

[ZPE*17] Zhu J.-Y., Park T., Isola P., Efros A. A.: Unpaired image-to-image translation using cycle-consistent adversarial networks. In Proceedings of the IEEE international conference on computer vision (2017), pp. 2223–2232. 2

© 2020 The Author(s)

Eurographics Proceedings © 2020 The Eurographics Association.