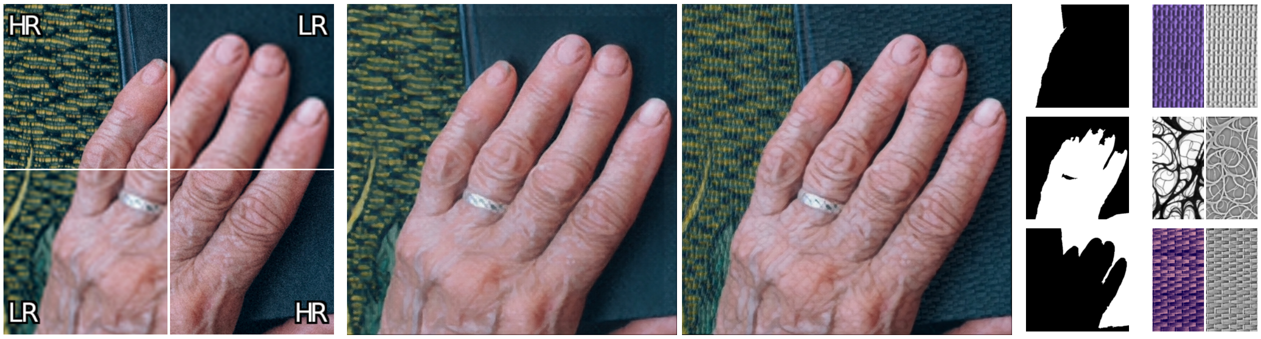
2021 IEEE International Conference on Image Processing (ICIP) | 978-1-6654-4115-5/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICIP42928.2021.9506261

SHALLOW MULTI-SCALE NETWORK FOR STYLIZED SUPER-RESOLUTION

Thibault Durand[*], *Julien Rabin and David Tschumperlé*

Normandie Univ, UNICAEN, ENSICAEN, CNRS, GREYC, 14000 Caen, France
{Thibault.Durand, Julien.Rabin, David.Tschumperle}@unicaen.fr



HR & LR (bicubic interp.) Result of SR-step ($\times 4$) Result of ST-step Masks β_i Styles Y_i

Fig. 1: *The proposed method performs super-resolution (SR) in two steps. The low resolution (LR) input x (here, the test image* 0787 *from the DIV2K dataset [1], upsampled with bicubic interpolation) is first processed with a shallow multi-scale convolutional network (SR-Step). Within the same network, the resulting image can then be locally stylized (ST-step, here with user-defined "Masks") to add lost details, such as textures or grain, from pretrained "Styles". On the right, DoG_5 filters are applied to the style images, showing the high frequency patterns used to train the ST-Network. Masks are defined using GMIC [2] "interactive extract foreground". More examples can be found at [3].*

ABSTRACT

Image Super Resolution (SR) has come a long way since the early age of image processing. Deep learning methods nowadays give outstanding results, yet very few are actually used in digital illustration and photo retouching software due to large memory storage and GPU computational requirements, but also due to the actual lack of control provided to the user over the final result. This paper introduces a two-step framework for stylized SR using a multi-scale network built with independent parallel branches. The approach aims at: *i.* designing a *shallow* network based on image processing techniques making it usable on light hardware architecture (low memory cost, no GPU); *ii.* providing a versatile, controllable and *customizable* network to *stylize* SR results in a plug-and-play manner. We show that the proposed method offers significant advantages over state-of-the-art reference-based approaches regarding these aspects.

Index Terms— Image Super Resolution; Style Transfer; Shallow Neural Network; Texture Synthesis; Interactive Computation

***Acknowledgments**— This work has been supported by Région Normandie under PhD grant RIN.

1. INTRODUCTION

The goal of Super-Resolution (SR) is to recover the geometry and textures of an unknown High Resolution image (HR) given a degraded Low Resolution image (LR). The degradation process consists in a possible blurring followed by subsampling, which mainly cuts out high frequencies and deteriorates medium frequencies as well.

Single Image Super Resolution (SI-SR) consists in performing SR from the low resolution image alone. Over the last years, deep-learning based methods have achieved significant PSNR improvements. SRCNN [4], the first convolutional neural network designed for SR, learns an end-to-end image mapping function between LR and HR images. Since then, numerous methods have been proposed mainly focusing in improving PSNR, using deeper and wider networks, either processing directly the low resolution input [5, 6] or its bicubic / bilinear interpolation upsampling [7, 8, 9].

The aforementioned methods are trained with pixel-wise losses, using mean square error (MSE) or mean absolute error. The PSNR metric, based on MSE, is somehow good to quantify evaluation, but ignores human perceptions and is limited for recovering textures, as exposed for instance in [10]. To circumvent this limitation, more sophisticated losses have been proposed, inspired from the literature in image synthesis [11, 12], providing more visually pleasing results, although with lower PNSR. While the former yields in texture-less image, the latter make use of deep semantic features to generate realistic details. For instance, EnhanceNet [13] builds upon the loss introduced by Gatys et al. [12, 14] for image synthesis, and is based on pre-trained features from the VGG classification network [15] to capture high-level/semantic information, later coined as the "perceptual loss" by [16]. SR-GAN [10] adopted the adversarial loss strategy introduced in [11] and exhibits interesting results.

Reference-based image SR (Ref-SR) aims to transfer the desired high resolution textures from a reference image to the low resolution image. Patch matching methods can recover textures from images [17] which is something SI-SR methods hardly do. Recently, convolutional neural networks such as [18, 19] aim at matching VGG features [15] from the reference image within the trained network, which yields noticeable improvements when making use of a relevant reference picture (*e.g.* same scene under a similar viewpoint).

Both SI-SR and Ref-SR methods suffer from practical limitations. To begin with, the number of parameters used in recent approaches is very large, typically millions of parameters, especially when encoders are used to extract perceptual features. This results in long inference times on CPU and requires large memory storage, penalizing near real-time time processing. Additionally, being deep and wide, such CNN are difficult to train and analyze. More importantly, the aforementioned end-to-end models do not leave much room for user control, as for instance proposed in [20] for colorization where the network lets one choose among the automatically generated color palettes. Even Ref-SR methods such as [18] or TTSR [19] are fully automatic after choosing a reference image and do not offer fine and local control over the outcome, as demonstrated in experimental section.

Contributions and outline. Our contributions are two folds. First, we propose in § 2.1 a shallow architecture to perform SR based on a *Multi-Scale Neural Network*. As demonstrated already by [21] for texture synthesis using perceptual loss, a very shallow CNN can be used to achieve high quality synthesis by making use of a multi-scale architecture, as opposed to adversarial methods. Unlike previous SR approaches such as LapSRN [22] or MDSR [9] using sequential upsampling, we propose a simple parallel processing based on linear scale-space analysis, yet efficient when compared to state-of-the-art. Secondly, we extend in § 2.2 the network with stylization branches which enable the user to control the synthesis of fine and textured details. By simply specifying locally pre-trained style, it allows the user to amend the SR result, as illustrated in Fig. 1.

2. MULTI-SCALE STYLIZING NETWORK

In this section, we introduce a **S**hallow **M**ulti-**S**cale **S**uper **R**esolution convolutional neural network **(SMS-SR)**, combining parallel SR branches (SR-step, presented in § 2.1) with stylization branches (ST-step, described in § 2.2). An overview of the proposed architecture is shown in Fig. 2.

From now on, $\mathbf{X}$ (resp. $\mathbf{x}$) $\in \mathbb{R}^{K\times N\times N\times 3}$ refers to a collection of K HR (resp. LR) color images of size $N \times N$, both used during training and evaluation. The k-th input LR image from the collection, noted $x_k \in \mathbb{R}^{N\times N\times 3}$, is encoded using YCbCr color system. Note that the LR image x_k is first upsampled to the size of the desired HR, *e.g.* with bicubic interpolation, before being fed to the network.

2.1. Multi-Scale Convolutional Neural Network (SR-step)

Multi-scale decomposition. The SR network is composed of $n = 6$ parallel branches, which outputs are linearly combined. Each branch i is filtered using a **D**ifference **of G**aussian filter $\mathbf{DoG}_i$ to specialize on a specific frequency bandwidth. This ensures that each branch output is independent. Indeed, *DoG* corresponds to first order approximation of the Laplacian filter in linear scale space, as shown in SIFT [23] where *DoG* are used to achieve multi-scale features detection. As in [23], Gaussian filters are parametrized by standard deviation with geometric progression.

Denoting θ the trainable parameters of the model, $R_{\theta_{i,k}}$ corresponds to the k-th convolution module $(1 \leq k \leq 4)$ parametrized by $\theta_{i,k}$ for the branch indexed by i. Each of such module begins with a 3×3 convolution, followed by a batch normalization, and ends up with a ReLU module. Finally, the 1-channel residual output of branch $1 \leq i \leq n$ can be written $(\text{SR}_\theta)_i(x_k) = [tanh \circ DoG_i \circ R_{\theta_{i,4}} \circ R_{\theta_{i,3}} \circ R_{\theta_{i,2}} \circ R_{\theta_{i,1}}](x_k)$. Since high frequencies are the most important missing data to recover, the number of channels in each branch increases for smaller scales. The total number of parameters is about 120k.

Finally, the YCbCr color output of the SR network (for pixel indexed by t) is the sum of outputs from parallel branches, concatenated with the input color channels

$$\text{SR}_\theta(x_k)(t) = x_k(t) + \left[\sum_{i=1}^{n} (\text{SR}_\theta)_i(x_k)(t);\ 0;\ 0\right] \in \mathbb{R}^3.$$

SR Training. As previously mentioned in the introduction, we combine MSE with a perceptual loss to train the SR network, as it is widely known that optimizing MSE alone favors texture-less reconstruction in SR [13]. The SR Network is optimized by solving: $\min_\theta \mathcal{L}_{\text{SR}}(\mathbf{X}, \text{SR}_\theta(\mathbf{x}))$, with the following objective function

$$\mathcal{L}_{\text{SR}}(\mathbf{X}, \mathbf{Y}) = \sum_{k=1}^{K} \|X_k - Y_k\|^2 + \lambda_{\text{SR}} \mathcal{L}_{\text{Perc}}(X_k, Y_k). \quad (1)$$

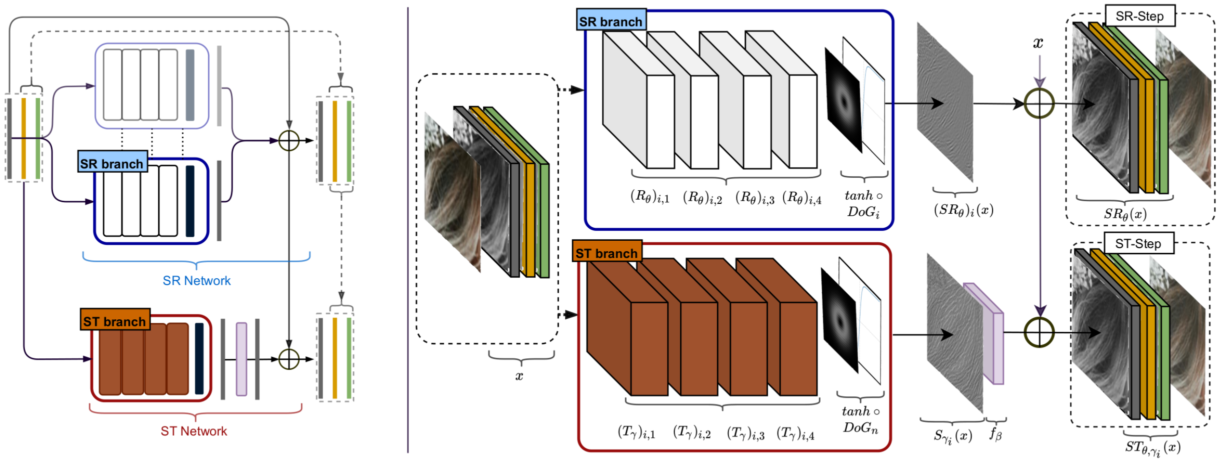

SR branch
SR Network
ST branch
ST Network
SR branch
ST branch
x
$(R_\theta)_{i,1}$ $(R_\theta)_{i,2}$ $(R_\theta)_{i,3}$ $(R_\theta)_{i,4}$
$tanh \circ DoG_i$
$(SR_\theta)_i(x)$
x
SR-Step
$SR_\theta(x)$
$(T_\gamma)_{i,1}$ $(T_\gamma)_{i,2}$ $(T_\gamma)_{i,3}$ $(T_\gamma)_{i,4}$
$tanh \circ DoG_n$
$S_{\gamma_i}(x)$
f_β
ST-Step
$ST_{\theta,\gamma_i}(x)$

Fig. 2: *Overview of the proposed architecture (left) composed of parallel branches SR and ST (detailed on right), respectively for super-resolution and stylization. The output is synthesized using multi-scale representation based on DoG filters.*

where $\|.\|$ stands as the Frobenius norm, $\mathcal{L}_{\text{Perc}}(x, y) = \sum_{\ell \in L_{\text{Perc}}} \|\phi_\ell(x) - \phi_\ell(y)\|^2$ is the perceptual loss, and $\phi_\ell(.)$ corresponds to the normalized feature maps at the ℓ-th layer of VGG-16 [15]. Large scale details are mainly provided by the upsampled input image. Driven by the MSE loss function, the first branches of the network learn mostly to remove artifacts from the interpolated input such as aliasing. We consider features from different layers of the VGG network in order to capture different scale details : $\text{L}_{\text{Perc}} = \{5, 9, 13\}$. As reported by [13], such perceptual loss induces checkerboard artifacts that we suppress using a 2×2 median filter.

2.2. Style Branches (ST-step)

In order to transfer details from user-defined reference images into the SR output, we now design additional style branches operating in parallel of the previous SR network. For the sake of simplicity, only one of such branch is represented in Fig. 2. Each style branch is trained independently in adding coherent details to the output of the pre-trained SR network.

Denoting γ the trainable parameters of the style branches, $T_{\gamma_{i,k}}$ refers to a residual module (number k for branch i), consisting in two 3×3 convolutional layers, as proposed in [16]. The 1-channel residual output of the i-th style branch can be written as followed: $S_{\gamma_i}(x_k) = [tanh \circ DoG_5 \circ T_{\gamma_{i,4}} \circ T_{\gamma_{i,3}} \circ T_{\gamma_{i,2}} \circ T_{\gamma_{i,1}}](x_k)$. Note the *DoG* filter used at the end of the branch to generate only small scale details. The number of parameters γ per style branch is less than 35K.

In order to control the residual, a normalization module f_β is used. It aims at enforcing the first and second order moment ($\overline{y_k}$ and σ_{y_k}) of the output residual batch of patches (number p) $y_p = \text{S}_{\gamma_i}(x_p)$ to be close to 0 and $\sigma_{x_p} \odot \beta$.

$$f_\beta(y_p) = \beta \odot \frac{\sigma_{x_p}}{\sigma_{y_p}}(y_p - \overline{y_p})$$

where $\odot$ indicates pixelwise multiplication, and β is a user-defined pixel map, as illustrated in Fig 1. Considering m style branches, the output of the proposed SMS-SR network at pixel t is given by

$$\text{ST}_{\theta,\gamma}(x_k)(t) = \text{SR}_\theta(x_k)(t) + \left[\sum_{i=1}^{m} f_\beta(S_{\gamma_i}(x_k)(t)); 0; 0\right].$$

ST Training. During training we set $\beta = 1$.We define the following objective function $\mathcal{L}_{ST}$, for a given reference style image Y_i

$$\mathcal{L}_{\text{ST}}(\mathbf{X}, Y_i, \mathbf{Z}) = \sum_{k=1}^{K} \lambda_{ST} \mathcal{L}_{\text{Perc}}(X_k, Z_k) + \mathcal{L}_{\text{Tex}}(Y_i, Z_k).$$

where the texture function is defined with normalized Gram matrix G, accordingly to [12]

$$\mathcal{L}_{\text{Tex}}(x, y) = \sum_{\ell \in L_{\text{Tex}}} \|G(\phi_\ell(x)) - G(\phi_\ell(y))\|^2. \qquad (2)$$

In order to favor small scale details synthesis from the reference image, we consider $\text{L}_{\text{Tex}} = \{2, 5, 7, 9\}$ and set $\text{L}_{\text{Perc}} = \{7\}$ to preserve large scale information from the SR output. The style branch is then optimized by solving $\min_{\gamma_i} \mathcal{L}_{\text{ST}}(\mathbf{X}, Y_i, \text{ST}_{\theta,\gamma_i}(\mathbf{x})), \forall\, 1 \leq i \leq m$.

3. EXPERIMENTS

Data and training setup. To evaluate our method, we train and test our model on the *DIV2K* dataset [1] for the ×4 SR bicubic challenge. Note that it only provides LR and HR image pairs for training and validation datasets. As a result, the last 150 images (out of 800) from the training set were hold

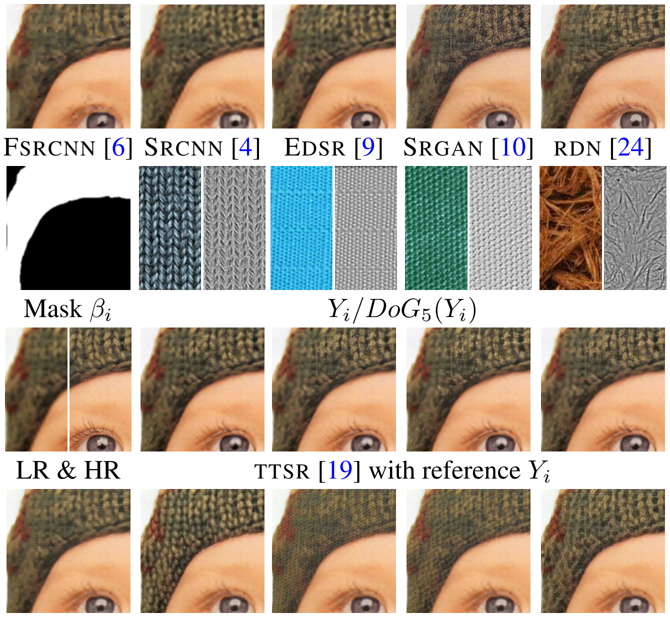


Fig. 3: *Comparison of $4\times$ SR results on cropped `baby` (Set5 dataset) for various SISR methods (first row) and RefSR methods (TTSR on third row, and proposed on fourth) with different styles Y_i (on second row). Zoom on the images to appreciate the details. More examples can be found at [3].*

Model	# Param.	DIV2K	Set5	Set14	Bsd100
SMS-SR (MSE)	120K	0.86	2.28	0.98	0.59
SRCNN [4]	440K	0.72	1.95	0.81	0.54
EDSR [9]	1517K	1.54	4.71	1.86	1.24
SRGAN [10]	1554K	-0.50	2.00	-0.19	-0.48
RDN [24]	2205K	1.59	4.76	1.83	1.29

Table 1: *Comparison of Average PSNR gains on different datasets for $4\times$ SR. Methods are ranked based on the number of parameters.* **SMS-SR (MSE)** *corresponds to the proposed method restricted to the SR-step (without the stylization step).*

out to build a testing dataset with ground-truth HR images. During training, square patches (254x254) are extracted from training image indexed from `001` to `650` and fed through the network. More than 20K patches (with a minimum variance) are used for training.

SR-Step Evaluation. While the main purpose of the proposed method is to offer a user-friendly controllable environment for stylized super-resolution, we first investigate here the quality of the proposed shallow SR network using PSNR. Even if PSNR is somewhat a flawed metric (as illustrated in Table 1 for [10]), it remains a standard metric for benchmarks.

Table 1 shows the average PSNR gains compared to bicubic upsampling for various methods showcased in the first row of Fig. 3. The values correspond to the PSNR of the average MSE computed on Y channel. The evaluation is conducted on the DIV2K test set described above and on three other standard benchmark datasets (*Set5*, *Set14*, and *BSD100*). The number of parameters (*#Params*) is rounded up to thousandth of parameters. We denote SMS-SR (MSE), our multi-scale model trained optimizing MSE only, *i.e.* $\lambda_{SR} = 0$ in (1). Note that we have used tensor-flow implementations of other methods, trained on the whole training set of DIV2K.

Observe that, while having significantly less parameters than other methods, the proposed shallow network still achieves interesting performance. Indeed Fig. 3 shows that, similarly to SRCNN [4] (large configuration) and EDSR [9] which are trained with pixel-wise loss, SMS-SR (MSE) allows a good reconstruction of simple structures (such as edges and lines) for a good amount of parameters. However and as expected, the proposed method restricted to the SR-step is not able to generate missing textures, contrary to very-deep adversarial methods such as RDN [24] which has approximately 18 times more parameters. Instead, the proposed network lets the user choose the desired type of generated details by selecting the appropriate branch (which adds barely 35K parameters to the model), as shown in Fig. 3 (ST-step) and discussed in the next paragraph.

Stylization with ST-step. We now consider the full SR network trained with the perceptual loss (*i.e.* setting $\lambda_{SR} = 1$ in (1)). As described in § 2.2, style branches are trained one by one after training the SR network. Then, as illustrated in Fig. 1 and 3, the user may use a mask or a brush to apply the desired missing texture. This idea is similar to [25] where textures are transferred locally, using patch based optimization methods. TTSR [19], which has more than 9M parameters, is used here as state-of-the-art baseline for Ref-SR. Observe how it does not allow to enforce the style of the reference image texture locally, contrarily to the proposed framework.

4. DISCUSSION AND CONCLUSION

We have proposed a shallow network architecture for stylized super-resolution. It is composed of parallel and independent SR branches combined in a multi-scale representation of the image. Stylization branches, trained independently, allows to generate texture details being lost in the degradation process. While having significantly less parameters, the proposed method competes favorably with Ref-SR based approaches and offers local control of the output result, as opposed to fully automated methods from the literature. The multi-scale architecture makes the method simple to train and easy to update by adding stylization branches.

Future works include the use of adversarial techniques, which remains a challenging problem with shallow networks, and guided segmentation to assist the user in generating masks.

5. REFERENCES

[1] R. Timofte, S. Gu, J. Wu, L. Van Gool, L. Zhang, M.-H. Yang, M. Haris, et al., "Ntire 2018 challenge on single image super-resolution: Methods and results," in *Proceedings of CVPR Workshops*, June 2018.

[2] D. Tschumperlé and S. Fourey, *G'MIC (GREYC's Magic for Image Computing): A Full-Featured Open-Source Framework for Image Processing*, https://gmic.eu.

[3] https://durand192.users.greyc.fr/SMS-SR/.

[4] C. Dong, C. Loy, K. He, and X. Tang, "Image Super-Resolution Using Deep Convolutional Networks," *IEEE TPAMI*, vol. 38, 2016.

[5] W. Shi, J. Caballero, F. Huszar, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, "Real-Time Single Image and Video Super-Resolution Using an Efficient Sub-Pixel Convolutional Neural Network," in *Proceedings of CVPR*, Las Vegas, NV, USA, June 2016, pp. 1874–1883, IEEE.

[6] C. Dong, C. C. Loy, and X. Tang, "Accelerating the Super-Resolution Convolutional Neural Network," in *Proceedings of ECCV*, Aug. 2016.

[7] J. Kim, J. Kwon Lee, and K. Mu Lee, "Accurate Image Super-Resolution Using Very Deep Convolutional Networks," in *Proceedings of CVPR*, 2016, pp. 1646–1654.

[8] J. Kim, J. K. Lee, and K. M. Lee, "Deeply-Recursive Convolutional Network for Image Super-Resolution," in *Proceedings of CVPR*, Las Vegas, NV, USA, June 2016, pp. 1637–1645, IEEE.

[9] B. Lim, S. Son, H. Kim, S. Nah, and K. M. Lee, "Enhanced Deep Residual Networks for Single Image Super-Resolution," in *Proceedings of CVPR Workshops*, July 2017.

[10] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, et al., "Photo-realistic single image super-resolution using a generative adversarial network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4681–4690.

[11] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *NIPS*, vol. 27, pp. 2672–2680, 2014.

[12] L. Gatys, A. Ecker, and M. Bethge, "Texture synthesis using convolutional neural networks," *Advances in neural information processing systems*, vol. 28, pp. 262–270, 2015.

[13] M. Sajjadi, B. Scholkopf, and M. Hirsch, "Enhancenet: Single image super-resolution through automated texture synthesis," in *Proceedings of ICCV*, 2017, pp. 4491–4500.

[14] L. Gatys, A. Ecker, and M. Bethge, "A neural algorithm of artistic style," *arXiv preprint arXiv:1508.06576*, 2015.

[15] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *arXiv:1409.1556 [cs]*, Apr. 2015, arXiv: 1409.1556.

[16] J. Johnson, A. Alahi, and Li Fei-Fei, "Perceptual Losses for Real-Time Style Transfer and Super-Resolution," in *Proceedings of ECCV*, vol. 9906, pp. 694–711. 2016.

[17] H. Zheng, M. Ji, L. Han, Z. Xu, H. Wang, Y. Liu, and Lu Fang, "Learning Cross-scale Correspondence and Patch-based Synthesis for Reference-based Super-Resolution," in *Procedings of BMCV*, London, UK, 2017, p. 138.

[18] Z. Zhang, Z. Wang, Z. Lin, and H. Qi, "Image super-resolution by neural texture transfer," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[19] F Yang, H. Yang, J. Fu, H. Lu, and B. Guo, "Learning Texture Transformer Network for Image Super-Resolution," in *Proceedings of CVPR*, Seattle, WA, USA, June 2020, pp. 5790–5799, IEEE.

[20] R. Zhang, P. Isola, and A. A. Efros, "Colorful Image Colorization," in *Proceedings of ECCV*, 2016, vol. 9907, pp. 649–666.

[21] D. Ulyanov, V. Lebedev, A. Vedaldi, and V. Lempitsky, "Texture networks: Feed-forward synthesis of textures and stylized images.," in *ICML*, 2016, vol. 1, p. 4.

[22] W.-S. Lai, J.-B. Huang, N. Ahuja, and M.-H. Yang, "Deep laplacian pyramid networks for fast and accurate super-resolution," in *Proceedings of CVPR*, 2017.

[23] D. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[24] Y. Zhang, Y. Tian, Yu Kong, B. Zhong, and Y. Fu, "Residual dense network for image super-resolution," in *Proceedings of CVPR*, 2018, pp. 2472–2481.

[25] Y. HaCohen, R. Fattal, and D. Lischinski, "Image upsampling via texture hallucination," in *2010 IEEE International Conference on Computational Photography (ICCP)*. IEEE, 2010, pp. 1–8.