Réseaux de Neurones Légers et Modulaires pour le Transfert de Styles à Deux Échelles

Thibault DURAND, Julien RABIN, David TSCHUMPERLÉ

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

Résumé – Notre travail propose une approche originale pour le transfert de plusieurs styles. Nous utilisons deux architectures modulaires, légères et complémentaires de réseaux neuronaux permettant chacune de transférer les caractéristiques d'une image de style à une échelle donnée. Ainsi, nous permettons, par assemblage, la combinaison de structures larges d'une première image de style avec les détails fins d'une seconde image de style. Ce problème est à distinguer du principe de mélange de styles qui fusionne les styles à différentes échelles sans en préserver les caractéristiques individuelles.

Abstract – We introduce a new approach for multi-styles transfer through two modular, lightweight and scale-complementary neural network architectures for style transfer at specific scale. Our work is devoted to the combination of large-scale structures of a first style image with fine-scale textures of a second style image. Such problem is different from style mixing which consists in blending style features without conserving it independently.

1 Introduction

Le transfert de style (TS) consiste à éditer une image de « contenu » avec les caractéristiques d'une image dîtes de « style ». Depuis les travaux précurseurs de Gatys et al. [1], de nouvelles méthodes basées sur des réseaux de neurones [2, 3, 4] ont vu le jour à partir de caractéristiques dîtes « perceptuelles » apprises en amont sur des tâches annexes par exemple de classification. Ici, nous combinons différents styles en transférant leurs caractéristiques à des échelles différentes. Étonnamment, ce type de problème n'est considéré dans la littérature qu'en mélangeant les styles par interpolation des caractéristiques [5, 6, 7, 8]. Ces methodes synthétisent de nouvelles caractéristiques sans pour autant générer les caractéristiques de chaque style individuellement. Combiner les styles sans les mélanger est une tâche difficile, comme montré par Gatys et al. [9]. En effet, les caractéristiques géométriques à différentes échelles sont intriquées dans les couches profondes des encodeurs utilisés, de sorte qu'il est difficile de les séparer. Aussi, comme montré dans [8], l'utilisation naïve d'une pénalité perceptuelle [1] avec différents styles lors d'une optimisation commune résulte en la synthèse de chacune des caractéristiques dans des zones différentes.
Dans cet article, nous proposons une solution originale permettant de combiner des caractéristiques à différentes échelles, c'est à dire de conserver les caractéristiques générales de l'image d'entrée tout en y incorporant les structures d'une première image de style et les détails d'une seconde image de style (Fig. 1). Notre méthode repose sur deux réseaux légers (∼155k), permettant un apprentissage et une évaluation rapide. Puisque stylisant des échelles complémentaires, les réseaux sont optimisés indépendamment et combinés à souhait durant l'évaluation.

Un aperçu de la littérature est d'abord présenté en Section 2 avec une attention particulière sur la méthode [9]. Notre architecture alternative est présentée Section 3. Nous montrons finalement les résultats en Section 4.

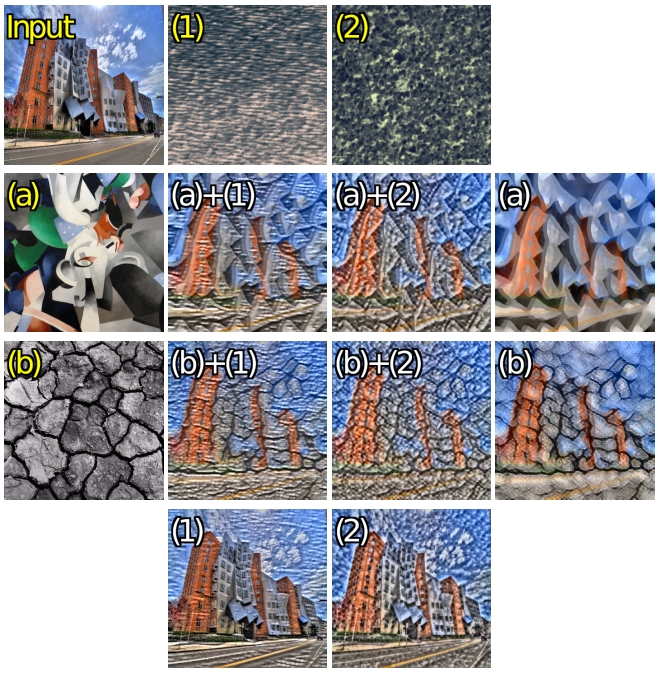


FIGURE 1 – *Résultats du transfert de styles à deux échelles. L'image de contenu* Input *est stylisée avec les styles* a *and* b *combinés aux textures* 1 *and* 2 *et ce à l'aide de réseaux légers et modulables. Le transfert de style pour les réseaux pris individuellement correspond à* a, b, 1, 2. *D'autres résultats sont disponibles sur [10].*

2 Etat de l'art et Motivations

2.1 Optimisation pour le transfert de style

Tout au long du papier, nous considérons la *pénalité perceptuelle* suivante introduite initialement dans [11] pour la synthèse de texture et le transfert de style :

$$\begin{aligned}\mathcal{L}_{\omega,\gamma}(C,S,Y) &= \sum_\ell \omega_\ell \|\phi_\ell(I) - \phi_\ell(Y)\|^2 \\ &+ \sum_\ell \gamma_\ell \|G(\phi_\ell(S)) - G(\phi_\ell(Y))\|^2 \end{aligned} \quad (1)$$

où $\|.\|$ correspond à la norme de Frobenius et $\phi_\ell(.)$ correspond aux descripteurs normalisés de la ℓ-ième couche du VGG-19 [12] (souvent notées `ReLu_11`, `ReLu_21`, `ReLu_31`, `ReLu_41`, `ReLu_51`). Le premier terme (pondéré par ω) permet la préservation de l'information spatiale de l'image de *contenu* I dans l'image stylisée Y. Le second terme (pondéré par γ) impose les caractéristiques de l'image de *style* de S à partir des matrices de Gram normalisées G.
Dans ce travail, nous considérons trois pénalités différentes notées $\mathcal{L}_A$, $\mathcal{L}_C$ et $\mathcal{L}_F$, associées aux coefficients suivants :

- $\mathcal{L}_A$ (toutes échelles) : $\omega = \omega_A = [0,0,0,1,0]$, $\gamma = \gamma_A = [1,1,1,1,1]$
- $\mathcal{L}_C$ (échelles Larges) : $\omega = \omega_C = [0,0,1,0,0]$, $\gamma = \gamma_C = [0,0,1,1,1]$
- $\mathcal{L}_F$ (échelles Fines) : $\omega = \omega_F = [0,0,0,1,0]$, $\gamma = \gamma_F = [1,1,0,0,0]$

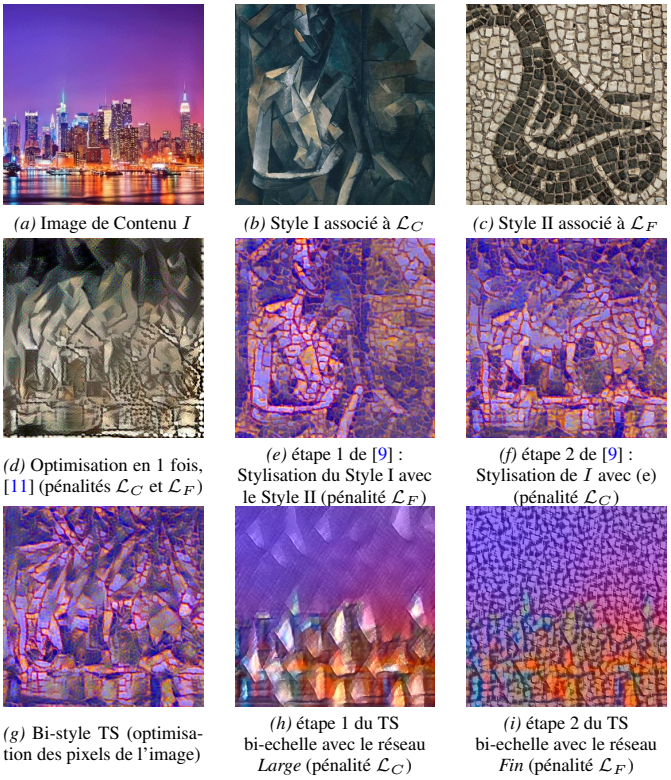
La plupart des méthodes de la littérature sont construites avec une pénalité proche de $\mathcal{L}_A$ comme décrit à l'origine dans [11].

2.2 Contrôle en transfert de style

Le problème du contrôle en transfert de style a déjà été étudié, souvent en entrainant des réseaux profonds ou bien en optimisant directement les pixels de l'image. Par exemple, [13] propose des paramètres imposés pendant l'entraînement mais permettant à l'utilisateur de choisir le style et l'intensité de la stylisation pendant l'inférence. [14] intègre des portes activant ou désactivant différents sous-réseaux chacun spécialisé dans un style permettant d'encoder une palette de styles qui peuvent chacun être utilisé après entraînement. Concernant le mélange de caractéristiques, [15] propose un réseau permettant le mélange de plusieurs textures. Comme montré par [5], les méthodes basées sur le transport optimal permettent de définir et calculer la moyenne des distributions des caractéristiques de différents styles. Une approche similaire consiste à calculer le barycentre de différents styles avant mélange, au moment de l'inférence [7]. Ces approches permettent certes le mélange de différents styles, mais aucune ne permet à l'utilisateur de contrôler l'échelle des caractéristiques à transférer ou de conserver les caractéristiques de chacun des styles.

2.3 Transfert de styles à deux échelles

Le transfert de styles à plusieurs échelles a été très peu étudié dans la littérature. Il n'existe en effet - à notre connaissance - que le travail de Gatys et al. [9] qui propose une méthode pour combiner différents styles tout en conservant les caractéristiques de chaque style indidividuellement. Ce problème est non trivial : bien qu'issues de différentes couches



(a) Image de Contenu I — *(b)* Style I associé à $\mathcal{L}_C$ — *(c)* Style II associé à $\mathcal{L}_F$

(d) Optimisation en 1 fois, [11] (pénalités $\mathcal{L}_C$ et $\mathcal{L}_F$) — *(e)* étape 1 de [9] : Stylisation du Style I avec le Style II (pénalité $\mathcal{L}_F$) — *(f)* étape 2 de [9] : Stylisation de I avec (e) (pénalité $\mathcal{L}_C$)

(g) Bi-style TS (optimisation des pixels de l'image) — *(h)* étape 1 du TS bi-echelle avec le réseau *Large* (pénalité $\mathcal{L}_C$) — *(i)* étape 2 du TS bi-echelle avec le réseau *Fin* (pénalité $\mathcal{L}_F$)

FIGURE 2 – *Comparaison de différentes techniques pour mélanger des styles à différentes échelles, avec l'approche de [9], [11] et la notre (dernière ligne : optimisation pixel par pixel, puis via deux réseaux légers et indépendants).*

(i.e différentes résolutions), les caractéristiques perceptuelles de ϕ_ℓ dans (1) ne sont pas totalement indépendantes. En conséquence, les détails fins et les couleurs sont toujours en partie encodés dans les couches profondes. Comme illustré en Fig. 2*d*, minimiser simultanément les pénalités $\mathcal{L}_C$ et $\mathcal{L}_F$ génère les caractéristiques à différents endroits, sans les combiner.
Ainsi, [9] propose une approche en deux temps, et consistant à combiner deux styles (le style I en Fig. 2*b*, et le style II en Fig. 2*c*). Dans un premier temps, il effectue un TS sur le style aux détails fins avec la pénalité $\mathcal{L}_F$, puis en transférant les couleurs de l'image de contenu (Fig. 2*a*). Ensuite, cette dernière image (Fig. 2*d*) est utilisée pour effectuer un transfert de style aux échelles larges à l'aide de la pénalité $\mathcal{L}_C$. Cependant, l'approche consiste à optimiser directement les pixels de l'image et doit être reconduite pour chaque combinaison de styles et chaque image. Ainsi, nous proposons une méthode alternative d'optimisation où les caractéristiques sont introduites échelle par échelle : l'image Fig. 2*g* correspond ainsi à l'image de contenu d'abord stylisée avec les caractéristiques larges du premier style (pénalité $\mathcal{L}_C$) puis avec les caractéristiques fines du second style (pénalité $\mathcal{L}_F$). De la même manière, nous entraînons indépendamment les deux réseaux légers associés (Fig. 2*h*-&*i*), les rendant interchangeables et permettant des combinaisons arbitraires de styles durant l'inférence. Ces réseaux sont présentés dans la section suivante, et les résultats associés montrés et discutés Section 4.

3 Réseaux Modulaires à deux échelles

Notre réseau est construit par mise en cascade de deux réseaux complémentaires dont les entrées sont traitées à différentes échelles (aperçu Fig. 3).

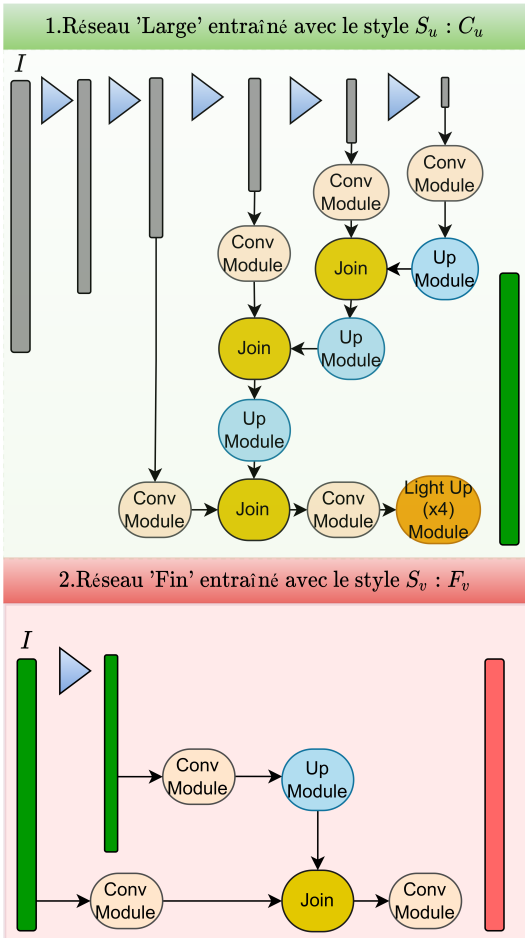


FIGURE 3 – *Architecture à deux échelles proposée. Le réseau 'Large' (resp. 'Fin') stylise/génère les caractéristiques larges échelles (resp. fines). Les deux réseaux sont entraînés indépendamment et combinés pendant l'inférence.*

Le premier réseau (Réseau 'Large' C_u, $\sim 110k$ paramètres, Fig. 3.1) génère les structures larges à partir du premier style S_u. Le second réseau génère les détails fins (Réseau 'Fin' F_v, $\sim 45k$ paramètres, Fig. 3.2) à partir d'un second style S_v. Les deux réseaux sont combinés par l'utilisateur pour générer le transfert de styles désiré. Cette architecture permet d'entraîner chaque module séparement une fois, sans réentrainer le modèle entier pour chaque combinaison de styles. L'architecture est inspirée du « Texture Network » V1 [2] pour lequel l'image d'entrée est utilisée à plusieurs échelles. L'architecture du réseau $\mathcal{C}$ (respectivement $\mathcal{F}$) permet de calquer les couches du réseau VGG utilisées dans le calcul des pénalités Larges $\mathcal{L}_C$ (resp. Fines $\mathcal{L}_F$). Ainsi, le réseau 'Fin', disposant d'un faible champ perceptuel, ne synthétise que les caractéristiques des deux premières couches du VGG. Le réseau 'Large' quant à lui, avec un champ perceptuel bien plus grand, génère les caractéristiques des couches plus profondes.

Details de l'architecture Sur la Figure 3, les modules de convolutions *Conv* sont composés de trois couches de convolution successives 3×3, chacune suivie d'opérateurs *batchNorm* et de fonctions d'activation *Relu*. Chaque module *Up* est composé d'un module de convolution suivi d'un sur-échantillonnage ($\times 2$) type plus proche voisin, ainsi qu'un opérateur *batchNorm*. Enfin, les modules de sur-échantillonnage légers *Light Up* sont construits avec deux modules *Up* successifs (équivalent à un sur-échantillonnage $\times 4$). Ces modules ont très peu de filtres, donc de paramètres, concentrant l'essentiel des paramètres avant les sur-échantillonnages et favorisant ainsi les structures larges.

Entrainement indépendant des réseaux Pendant l'entrainement, les paramètres de chaque réseau sont appris indépendamment pour un style S et des images de contenus I ($\sim 15k$ patches) issues de *DIV2K* [16]. En entrée, des batchs de 6 images de contenu, décomposées à différentes échelles à partir d'une image de 356×356 pixels. Ces images sont concaténées avec un bruit blanc gaussien pour chaque résolution, comme effectué par [2]. Le réseau 'Fin' $\mathcal{F}$ est entraîné avec la pénalité $\mathcal{L}_F(I, S, \mathcal{F}(I))$, le réseau 'Large' est entraîné avec la pénalité $\mathcal{L}_C(I, S, \mathcal{C}(I))$. Les paramètres des réseaux convergent en quelques milliers d'itérations à l'aide de l'optimiseur Adam avec un pas de descente fixé à $5e^{-2}$.

4 Expériences

Résultats pour le transfert de styles à deux échelles La figure 1 montre l'ensemble des combinaisons possibles pour le transfert de style à deux échelles sur une image de contenu donnée en haut à gauche, à partir de deux réseaux 'Fins' (associés aux styles *1* et *2*), ainsi que deux réseaux 'Larges' (associés aux styles *a* et *b*). Toutes les combinaisons ($\mathcal{F}_i \circ \mathcal{C}_x(I)$ avec $x =$*a* ou *b* et $i =$*1* ou *2*) sont montrées, notamment les résultat pour le transfert de style simple ($\mathcal{C}_x(I)$ et $\mathcal{F}_i(I)$).
Les images Figure 2*h&i* montrent les résultats du réseau 'Large' seul $\mathcal{C}_b(I)$ et couplé au réseau 'Fin' $\mathcal{F}_c \circ \mathcal{C}_b(I)$. Les caractéristiques des différentes images de style sont effectivement transférées et combinées sur l'image de contenu à deux échelles différentes, tout en étant individuellement préservées. La comparaison avec l'optimisation itérative directement sur les pixels de l'image ainsi que la même pénalité (Figures 2*h*) montre que l'architecture multi-échelle du réseau $\mathcal{F}$ permet bien de restreindre - bien que limitée par le nombre de paramètres - la modification de style aux échelles les plus fines. La taille des caractéristiques intégrées dépend bien du champ perceptuel de l'architecture proposée, ici volontairement réduite.
Pendant l'entraînement, les couleurs sont prédites à l'aide de la pénalité perceptuelle (1), et donc du style associé au réseau final.Il est aussi possible d'utiliser la luminance stylisée avec les canaux de chrominance (Cb,Cr) de l'image de contenu. Pour

éviter les éventuels artefacts obtenus en combinant des images avec des structures géométriques différentes, on ajoute possiblement le filtre NLMR [17] dont le fonctionnement est accéléré par un filtre guidé [18]. Sur la figure 1 par exemple, le filtre est appliqué avec les couleurs de l'image originale de contenu. Quant à la figure 4, les couleurs de l'image issue du réseau 'Large' sont utilisées.

Résultats pour la synthèse de textures à deux échelles La figure 4 illustre les résultats obtenus pour la combinaison de textures à deux échelles à partir de deux réseaux 'Larges' (associés aux styles (*a* et *b*) ainsi que deux styles 'Fins' (*1* et *2*). Contrairement au transfert de styles à deux échelles, seul un tenseur aléatoire est donné en entrée du réseau. Aussi, les poids associés au terme de fidélité au contenu dans (1) sont réduits à $\omega = 0$. Remarquons que la synthèse de texture à partir des 2 réseaux 'Larges' favorise les grandes échelles (les larges structures des briques sont générées alors que le modèle ne synthétise pas le grain des briques, de *b* vers *b*). Inversement, le réseau 'Fin' favorise les détails fins (les traits générés de *1* ne sont pas aussi longs que dans *1*).
Quant à la combinaison de textures, les réseaux 'Larges' (pour la synthèse de textures) sont combinés aux réseaux 'Fins' (pour le transfert de styles) et ce afin de générer toutes les combinaisons possibles. Encore une fois, nous pouvons observer le mélange des différentes caractéristiques ainsi que la préservation de chacune d'elles.

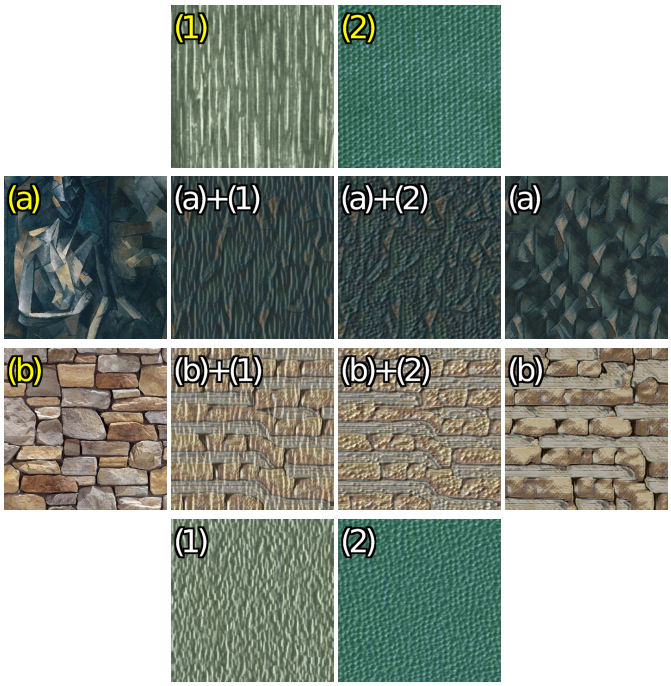


FIGURE 4 – *Résultats obtenus pour la synthèse de textures ainsi que le mélange de textures à deux échelles. Deux réseaux* Larges a,b *sont combinés avec deux réseaux* Fins 1,2. *L'ensemble permet de mélanger au moment de l'inférence des caractéristiques à différentes échelles tout en les préservant.*

Références

[1] L Gatys, A.S Ecker, and M Bethge, "Texture synthesis using convolutional neural networks," *Advances in neural information processing systems*, vol. 28, 2015.

[2] D Ulyanov, V Lebedev, A Vedaldi, and V Lempitsky, "Texture networks : Feed-forward synthesis of textures and stylized images," 03 2016.

[3] T.R Shaham, T Dekel, and T Michaeli, "Singan : Learning a generative model from a single natural image," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 4570–4580.

[4] J Johnson, A Alahi, and L Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *European Conference on Computer Vision*, 2016.

[5] J Rabin, G Peyré, J Delon, and M Bernot, "Wasserstein barycenter and its application to texture mixing," in *International Conference on Scale Space and Variational Methods in Computer Vision*. Springer, 2011, pp. 435–446.

[6] N Yu, C Barnes, E Shechtman, S Amirghodsi, and M Lukac, "Texture mixer : A network for controllable synthesis and interpolation of texture," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 12164–12173.

[7] Y Mroueh, "Wasserstein style transfer," in *AISTATS*, 2020.

[8] A Houdard, A Leclaire, N Papadakis, and J Rabin, "A generative model for texture synthesis based on optimal transport between feature distributions," SSVM 2021.

[9] L Gatys, A Ecker, M Bethge, A Hertzmann, and E Shechtman, "Controlling perceptual factors in neural style transfer," 07 CVPR 2017.

[10] https://durand192.users.greyc.fr/smsst/.

[11] L.A Gatys, A.S Ecker, and M Bethge, "A neural algorithm of artistic style.," *CoRR*, vol. abs/1508.06576, 2015.

[12] S Karen and Z Andrew, "Very deep convolutional networks for large-scale image recognition," in *International Conference on Learning Representations*, 2015.

[13] M Babaeizadeh and G Ghiasi, "Adjustable real-time style transfer," in *International Conference on Learning Representations*, 2020.

[14] X Chen, C Xu, X Yang, L Song, and D Tao, "Gated-gan : Adversarial gated networks for multi-collection style transfer," *IEEE Transactions on Image Processing*, vol. 28, pp. 1–1, 09 2018.

[15] Y Li, C Fang, J Yang, Z Wang, X Lu, and M-H Yang, "Diversified texture synthesis with feed-forward networks," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 266–274, 2017.

[16] 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 *CVPR*, June 2018.

[17] J Rabin, J Delon, and Y Gousseau, "Removing artefacts from color and contrast modifications," *IEEE Transactions on Image Processing*, vol. 20, no. 11, pp. 3073–3085, 2011.

[18] K He, J Sun, and X Tang, "Guided image filtering," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 6, pp. 1397–1409, 2012.