A PATCH-BASED APPROACH FOR ARTISTIC STYLE TRANSFER VIA CONSTRAINED MULTI-SCALE IMAGE MATCHING

Benjamin Samuth[⋆] *David Tschumperlé*[⋆] *Julien Rabin*[⋆]

[⋆] Normandie Univ., UNICAEN, ENSICAEN, CNRS, GREYC, 14000 Caen, France
{Benjamin.Samuth, David.Tschumperle, Julien.Rabin}@unicaen.fr

ABSTRACT

Since a few years and the advent of convolutional neural networks, algorithms for artistic style transfer between images have developed considerably. However, these methods require a relatively long training phase in order to succeed. This is why non-learning image processing approaches recently strove to propose patch-based algorithms able to aesthetically compete with neural methods. This paper goes one step further in this direction by introducing a new patch-based method for style transfer, using a constrained multi-scale version of the fast approximate nearest-neighbor algorithm *PatchMatch*, enforcing uniform sampling of style feature-patch. Our method also aims to mix the patch-based and neural paradigms by enabling the embedding of image patches in the feature space of the VGG-16 network.

Index Terms— Style Transfer, Patch-Based Method, Multi-Scale, Constrained Optimization, Perceptual Features.

1. INTRODUCTION

Style transfer, or image stylization, is one of the image manipulation processes that is able to drastically increase the aesthetic value of an image by borrowing the stylistic features of a *style* image and applying it over a *content* image.

A first approach to this process was to consider the problem as a proxy task of patch-based unsupervised texture synthesis [1, 2] and transfer [3, 4]. The idea is to locally preserve the coherence of the style just like a texture while matching the spatial structure of the content image. In that regard, multiple improvements of that principle were introduced. [5] uses an adaptative partition of the content image in order to mix larger patches of the style image in the less meaningful content-wise regions such as the sky or plain backgrounds. On the other hand, [6] proposes to synthesize a stylized image by using [7] texture optimization method, combined with a segmentation mask in order to preserve the content. This one can build new patterns from the patches of the style image even in plain regions. [8] introduces a multi-layer algorithm for semi-discrete optimal transport, and applies it in the patch space in the context of both texture synthesis and style transfer. This method then matches the patch distribution of the style image, while the content features are blended based on a edge detection.

Fig. 1: Proposed patch-based style transfer for different features Φ. Constrained matching ensures that the content image inherits the aesthetic of the style image. More results shown in [9].

In a broader sense, our method takes also inspiration of patch-based nearest neighbors algorithms such as [10, 11] for inpainting, [12, 13, 14] for texture generation, or [15, 16] for single image and video generation. We preserve the idea of using a multi-resolution nearest neighbor patch search, or in our case a fast approximation like [17] in order to match the style patches to the content patches, in a way that will promotes larger regions of style or texture while maintaining the geometrical coherency of the content along each resolution. [14, 18] also tackle the problem of optimal patch assignment that we introduce in our method in order to maintain the patch diversity distribution of the style image.

[19] introduced a new class of style transfer algorithms using deep neural networks like VGG [20], as they proved to encode the content and the style of images. Using a pixel-wise optimization process over these features, their method is able to synthesize convincing stylized images. This inspired many papers to improve over the issues of this technique [21, 22, 23]. Mainly, the training of such neural models

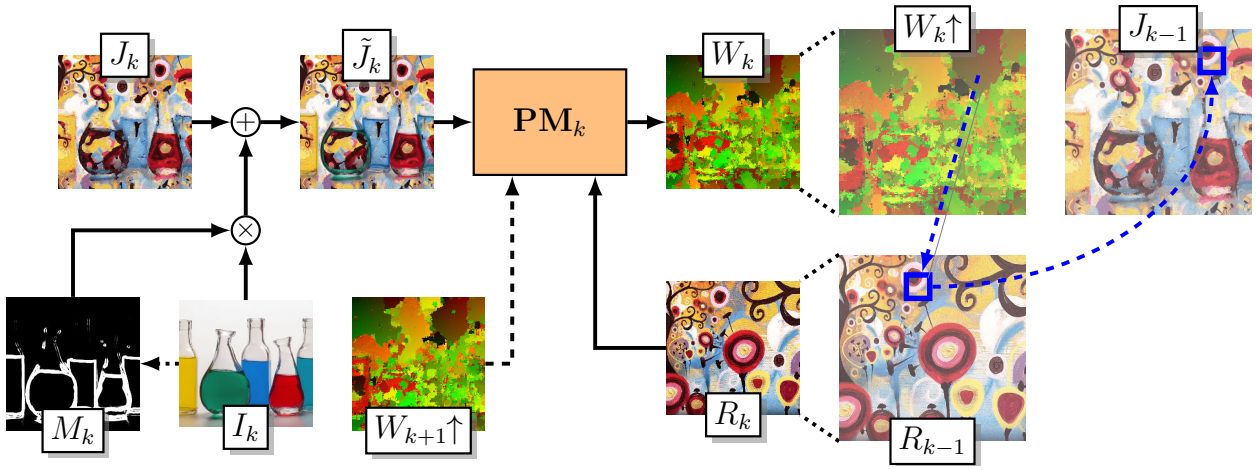


Fig. 2: Scheme of the algorithm at an intermediate scale k. The style transfer follows that sequence: the content detail injection, the nearest neighbor patch search, the upscaling of the nearest neighbor field and the synthesis by warping. There is no injection at the first scale $k = N$, and we directly use the original image at low resolution. Similarly, there is no upscaling process for $k = 0$.

is time consuming and requires a large image dataset.

We introduce in this paper a fast style transfer algorithm (Sec. 2.1) using a new constrained nearest neighbor search (Sec. 2.2). Our method is able to use neural features for its purpose as well (Sec. 3) as shown on Figure 1, which makes it a step towards hybrid geometrically explainable models.

2. METHOD

Let $I : \Omega_I \to \mathbb{R}^3$ the content image (*input*), R the style image (*reference*). We refer to Φ as the feature-patch extractor, such that $\Phi(I) : \Omega_I \to \mathbb{R}^{c\times\sigma^2}$ where c is the features dimensions and $\sigma \times \sigma$ indicates the patch dimensions.

Let F the set of field functions $W : \Omega_I \to \Omega_R$, which map a patch of I to a patch of R. At the core of the proposed framework is the question of finding such an optimal mapping W between $\Phi(I)$ and $\Phi(R)$. The content image is then *warped* into the stylized image J by averaging overlapping color patches from $\Phi(R)\circ W$. This work focuses on the patch nearest neighbor field (NNF) W^* defined as follows:

$$W^* \in \underset{W\in F}{\operatorname{argmin}} \sum_{p\in\Omega_I} ||\Phi(I)(p) - \Phi(R) \circ W(p)||^2 \quad (1)$$

NNFs have been successfully used for image processing (such as inpainting and texture synthesis) but suffer from several limitations that we address for style transfer. First, we resort to a multi-scale scheme in Section 2.1 to blend features from I and R at multiple-scale. In addition, in order to guarantee the diversity of used features, our method introduces an occurrence penalization inside the PatchMatch algorithm [17] in Section 2.2. For the sake of simplicity, we will omit the patch-feature operator Φ in the rest of the section. Its role is studied in further details in the experimental section.

2.1. Algorithm overview

In order to promote the creation of spatially continuous patch regions, we build a multi-scale representation $\{I_k\}$ of I at $N+1$ different resolutions ($k \in \{0, ...N\}$), with a scale factor of r. In a similar fashion as [11, 15, 16], by using a coarse-to-fine algorithm, a patch is able to capture the general layout of the content on the coarsest scales, as well as the details of the style image on the finest scales. The core of the patch style transfer at a scale k is described in Fig. 2. This scheme is repeated from N to 0 and will result in the synthesis of a stylized image. It can also be represented as a single module, in order to match with the paradigm of neural networks.

At the output of PatchMatch, the current NNF is increased in scale using a special method. Indeed, to help the regions grow from a scale to another, we upscale the previous NNF (noted $W^*_{k-1}\uparrow^r$) rather than the warped image directly. The upscaling algorithm is specially conceived for NNF (2), because the usual interpolations are not pertinent here. We did not choose to upscale the warped output and then compare it with the style image of the next scale since it would result in a comparison of different modalities of details (*ie.* the upscale blurs the image) which affect the search.

$$W\uparrow^r(p) = rW\left(\left\lfloor\frac{p}{r}\right\rfloor\right) + p - r\left\lfloor\frac{p}{r}\right\rfloor \quad (2)$$

Consider that the operators are applied element-wise for higher dimensions of coordinates p. In practice, W is a 2D array of image coordinates in Ω_R. The same result can be obtained iteratively by expanding to neighbors the scaled values and adding the corresponding offset, filling the "holes" left by the upscale. The upscaled NNF will also be given as initialization to the next scale PatchMatch, so the coherence of the patches regions of the field can persist through the different scales. The first initialization is a random field.

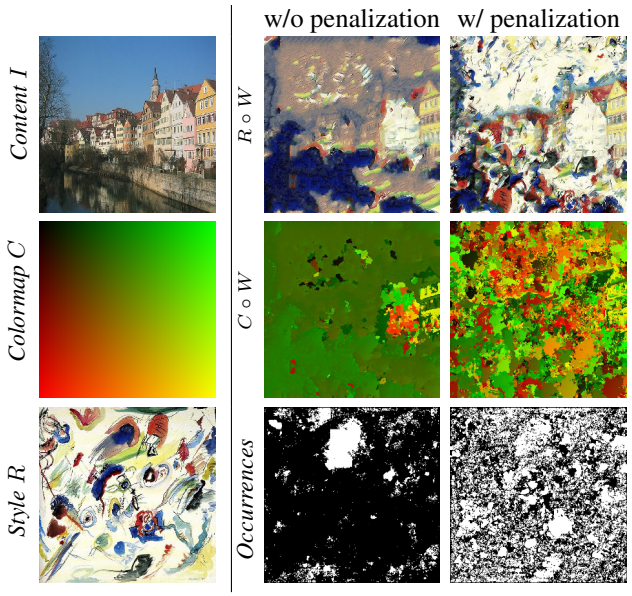


Fig. 3: Comparison of final NNF with/without occurrence penalization (Sec. 2.2). The penalization increases diversity of used patches. (1st and 2nd row) The color of a pixel indicates the patch warped from an image (style or colormap) on that coordinate. (3rd row) White pixels indicate a style patch being used at least once.

Let $k \in \{N, \dots, 0\}$ from N the coarsest scale to 0 the finest scale. The best NNF at the scale k is expressed as:

$$W_k^* \in \underset{W \in F}{\operatorname{argmin}} \sum_{p \in \Omega} ||\tilde{J}_k(p) - R_k \circ W(p)||^2 \tag{3}$$
$$\tilde{J}_k = (1 - \max(M_k, \rho_k)) \odot J_k + \max(M_k, \rho_k) \odot I_k$$

where $\tilde{J}_k$ is the linear blending of J_k (the output of the patch warping $R_k \circ W_{k+1}\uparrow^r$) and I_k the content image at scale k. This linear combination is weighted ($\odot$ indicates pixel-wise multiplication) by a mask M_k computed from the gradients of I_k as done in [8]. This mask is clamped with the parameter $\rho_k \in [0, 1]$ which controls the amount of injected geometric details from I_k. ρ_k typically decreases at smaller scales to preserve only large structures from the content image I.

2.2. Occurrence penalization

To accelerate the computation of the multi-scale NNFs, we use the fast approximate nearest-neighbor algorithm PatchMatch [17]. More precisely, our implementation of PatchMatch is using their proposed GPU scheme.

As we transfer large continuous regions of patches due to our coarse-to-fine architecture, one known issue is that these regions tends to repeat themselves, making noticeable copy artifacts [5, 13, 14]. This is why we introduce inside the PatchMatch algorithm an occurrence penalization that will favor patch diversity (Fig. 3), as well as making the style transfer more aesthetically pleasing and believable.

In [8], optimal patch assignment is enforced using semi-discrete optimal transport. The solution of this problem simply consists in penalizing each squared norm in (1) with a scalar variable $\lambda_W(y)$. This dual variable is optimized by stochastic gradient ascend in [8].

Given W the current estimation of the NNF by PatchMatch and $\nu \in \mathbb{N}^{\Omega_R}$ the occurrence constraints applied to the patches, in our setting, that principle transposes in the following update rule at each iteration of PatchMatch:

$$\lambda_W(y) \leftarrow \lambda_W(y) + \delta \partial_y \lambda_W(y) \tag{4}$$
$$\partial_y \lambda_W(y) = \frac{|\{p \mid y = W(p)\}|}{|\Omega_I|} - \frac{\nu(y)}{|\Omega_R|}$$

where δ is the gradient step. Theorically, we should have imposed $\nu(y) = 1$ to promote the uniform sampling of reference patches, but in practice, the boundary condition requires a different constraint on some patches.

Finally, the new optimization problem is now expressed as the following equation, using (3) and the optimal λ_W from (4):

$$W_k^* \in \underset{W \in F}{\operatorname{argmin}} \sum_{p \in \Omega} \frac{D_k(p)}{Var(D_k)} - \lambda_W \circ W(p) \tag{5}$$
$$\text{with} \quad D_k(p) = ||\tilde{J}_k(p) - R_k \circ W(p)||^2$$

$D_k(p)$ is divided by the variance to normalize the expression. In such a way, if $\lambda_W(y) > 0$, the patch y is favorised. Otherwise, when $\lambda_W(y) < 0$, it is penalized.

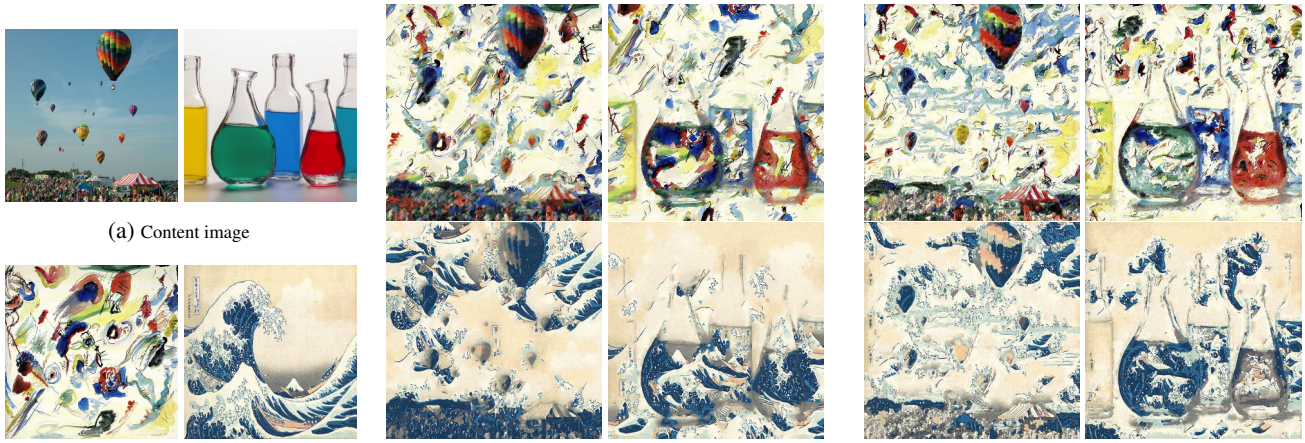
As shown in Fig. 3, the proposed penalization method effectively avoid artefacts during transfer (1st row) and promotes uniform sampling of patches (3rd row). The size of the transferred regions (2nd row) decreases when increasing δ.

Related works Our approach is different from other diversification methods like the normalization of scores of [15, 16] because we are directly using the occurrence count of the style patches in our comparison. Observe that, contrary to [14, 8] which require a large number of iterations to converge to the optimal solution, only a few (5 to 8) are enough in our setting to get a sufficient approximation for our purpose.

The proposed approach is however somehow similar to [13] in which an occurrence map is iteratively updated to penalize patch distances. Unfortunately, this strategy is not suitable for parallel computing, and we experimentally found it to be unstable. In our setting, the penalization occurs once the parallelized propagation and random search are done.

3. EXPERIMENTS AND DISCUSSION

Parameters As presented before, our method contains several parameters to alter the style transfer process. For the results showed in this paper, we set the scale ratio to $r = 1.3$ and the number of scales to $N = 14$ in the model. The patch



(a) Content image (b) Style images (c) Results using RGB+Gradients channels (d) Results using VGG-16 features (relu_1_1)

Fig. 4: Sample results of style transfer on 512×512 images (style and content). Our method manages to preserve the global aesthetic of the style by copying large regions of patches. The image were generated using the default parameters specified in Sec. 3. We are using a pre-trained VGG-16 model, as well as the pre-processing required by it, but our model does not need any explicit training by itself.

(a) Content image (b) Style image (c) [19] (d) [22] (e) [23] (f) Ours (relu_1_1)

Fig. 5: Comparisons with other methods. Thanks to the use of patches for our style transfer, our outputs are able to consistently match with the aesthetic properties of the style image.

size is fixed to $\sigma = 5$. Deeper scales or higher patch size generally imply larger copied regions of patches.

The user can also control the injection of content details with parameters ρ_k in Eq. (3). The parameter ρ_k decreases linearly between $\rho_{N-1} = 1.0$ (coarsest) to $\rho_0 = 0.5$ (finest scale), to ensure that the salient part of the content image is still noticeable in most of the cases. Finally, $\delta = 10^{-12}$ in our setup.

Patch-Features The results of the style transfer can be altered by using different features Φ than RGB. Other color spaces such as a weighted Lab color space can give finer tuning over the selection of patches. A color transfer can also largely improve the synthesized result in some cases and is a strategy used in the previous patch-based style transfer method [5, 6, 8]. Our method (Figs. 3,4c) uses by default the RGB channels along with the gradients of the image that we concatenate as supplementary channels (then, $c = 5$).

Motivated by the approach of neural style transfer of [19], we use the first convolution layer of VGG-16 [20] (hence $c = 64$). The feature space shows significant differences with the RGB channel in the synthesized image. The most notable one is a better understanding of flat colors from the neural layer (Fig. 4d). However, using a different space requires some tuning, especially in the learning step of the occurrence penalization, as its value become much more sensible. Generally speaking, the use of a too large feature space in dimensions unfortunately clashes with the process of growing regions of patches.

Comparison Our algorithm is able to synthesize convincing style transfer outputs (Fig. 4, 5) with a reasonable computation time compared to the neural approach whose training is time-consuming, especially if a database is needed.

Implementation Our method use a custom PatchMatch algorithm that internally includes our occurrence penalization (Sec. 2.2). The computation time on 512×512 RGB images is 20∼30 seconds. However, the parallelized patch extraction process of our algorithm is naturally expensive in memory due to the fact that it extracts the patches all at once, but a sequential implementation of PatchMatch would handle this.

Acknowledgment This work is partially supported by the project ANR-19-CHIA-0017.

4. REFERENCES

[1] Alexei A Efros and Thomas K Leung, "Texture synthesis by non-parametric sampling," in *Proceedings of the seventh IEEE international conference on computer vision*. IEEE, 1999, vol. 2, pp. 1033–1038.

[2] Michael Ashikhmin, "Synthesizing natural textures," in *Proceedings of the 2001 symposium on Interactive 3D graphics*, 2001, pp. 217–226.

[3] Alexei A Efros and William T Freeman, "Image quilting for texture synthesis and transfer," in *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, 2001, pp. 341–346.

[4] Aaron Hertzmann, Charles E Jacobs, Nuria Oliver, Brian Curless, and David H Salesin, "Image analogies," in *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, 2001, pp. 327–340.

[5] Oriel Frigo, Neus Sabater, Julie Delon, and Pierre Hellier, "Split and match: Example-based adaptive patch sampling for unsupervised style transfer," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[6] Michael Elad and Peyman Milanfar, "Style transfer via texture synthesis," *IEEE Transactions on Image Processing*, vol. 26, no. 5, pp. 2338–2351, 2017.

[7] Vivek Kwatra, Irfan Essa, Aaron Bobick, and Nipun Kwatra, "Texture optimization for example-based synthesis," in *ACM SIGGRAPH 2005 Papers*, pp. 795–802. 2005.

[8] Arthur Leclaire and Julien Rabin, "A Stochastic Multilayer Algorithm for Semi-Discrete Optimal Transport with Applications to Texture Synthesis and Style Transfer," *Journal of Mathematical Imaging and Vision*, July 2020.

[9] Benjamin Samuth, *Project Web page*, 2022, https://samuth211.users.greyc.fr/2022/StyleTransfer/.

[10] Yonatan Wexler, Eli Shechtman, and Michal Irani, "Space-time completion of video," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 29, no. 3, pp. 463–476, 2007.

[11] Alasdair Newson, Andrés Almansa, Matthieu Fradet, Yann Gousseau, and Patrick Pérez, "Video inpainting of complex scenes," *Siam journal on imaging sciences*, vol. 7, no. 4, pp. 1993–2019, 2014.

[12] Johannes Kopf, Chi-Wing Fu, Daniel Cohen-Or, Oliver Deussen, Dani Lischinski, and Tien-Tsin Wong, "Solid texture synthesis from 2d exemplars," in *ACM SIGGRAPH 2007 Papers*, 2007.

[13] Alexandre Kaspar, Boris Neubert, Dani Lischinski, Mark Pauly, and Johannes Kopf, "Self tuning texture optimization," *Computer Graphics Forum*, vol. 34, no. 2, pp. 349–359, 2015.

[14] Jorge Gutierrez, Julien Rabin, Bruno Galerne, and Thomas Hurtut, "Optimal patch assignment for statistically constrained texture synthesis," in *International Conference on Scale Space and Variational Methods in Computer Vision*. Springer, 2017, pp. 172–183.

[15] Niv Granot, Ben Feinstein, Assaf Shocher, Shai Bagon, and Michal Irani, "Drop the gan: In defense of patches nearest neighbors as single image generative models," *arXiv preprint arXiv:2103.15545*, 2021.

[16] Niv Haim, Ben Feinstein, Niv Granot, Assaf Shocher, Shai Bagon, Tali Dekel, and Michal Irani, "Diverse generation from a single video made possible," *arXiv preprint arXiv:2109.08591*, 2021.

[17] Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman, "Patchmatch: A randomized correspondence algorithm for structural image editing," *ACM Trans. Graph.*, vol. 28, no. 3, pp. 24, 2009.

[18] Ryan Webster, "Innovative non-parametric texture synthesis via patch permutations," *arXiv preprint arXiv:1801.04619*, 2018.

[19] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge, "Image style transfer using convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[20] Karen Simonyan and Andrew Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[21] Xun Huang and Serge Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 1501–1510.

[22] Hang Zhang and Kristin Dana, "Multi-style generative network for real-time transfer," in *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, 2018, pp. 0–0.

[23] Songhua Liu, Tianwei Lin, Dongliang He, Fu Li, Meiling Wang, Xin Li, Zhengxing Sun, Qian Li, and Errui Ding, "Adaattn: Revisit attention mechanism in arbitrary neural style transfer," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 6649–6658.