Wasserstein Generative Models for Patch-based Texture Synthesis

Antoine Houdard, Arthur Leclaire, Nicolas Papadakis, Julien Rabin

Wasserstein Generative Models for Patch-based Texture Synthesis

Antoine Houdard
Univ. Bordeaux
CNRS, IMB
UMR 5251, France

Arthur Leclaire
Univ. Bordeaux
CNRS, IMB
UMR 5251, France

Nicolas Papadakis
Univ. Bordeaux
CNRS, IMB
UMR 5251, France

Julien Rabin
Normandie Univ., UniCaen
ENSICAEN, CNRS
GREYC, UMR 6072, France

Abstract

In this paper, we propose a framework to train a generative model for texture image synthesis from a single example. To do so, we exploit the local representation of images via the space of patches, that is, square sub-images of fixed size (e.g. 4×4). Our main contribution is to consider optimal transport to enforce the multiscale patch distribution of generated images, which leads to two different formulations. First, a pixel-based optimization method is proposed, relying on discrete optimal transport. We show that it is related to a well-known texture optimization framework based on iterated patch nearest-neighbor projections, while avoiding some of its shortcomings. Second, in a semi-discrete setting, we exploit the differential properties of Wasserstein distances to learn a fully convolutional network for texture generation. Once estimated, this network produces realistic and arbitrarily large texture samples in real time. The two formulations result in non-convex concave problems that can be optimized efficiently with convergence properties and improved stability compared to adversarial approaches, without relying on any regularization. By directly dealing with the patch distribution of synthesized images, we also overcome limitations of state-of-the art techniques, such as patch aggregation issues that usually lead to low frequency artifacts (e.g. blurring) in traditional patch-based approaches, or statistical inconsistencies (e.g. color or patterns) in learning approaches.

1 Introduction

Image synthesis consists in creating photorealistic pictures while prescribing some desired attributes. Two main strategies have been investigated in the literature. One can either sample an image distribution using a *generative model* learnt from a large image dataset, as in GANs [Goodfellow et al., 2014]. On the other hand, one can turn to *exemplar-based synthesis*, that is, generating new images which exhibit features that are similar to the ones of a *single example*, as in [Gatys et al., 2015, Shaham et al., 2019] (possibly using "perceptual features" extracted with a neural network trained on an image database [Johnson et al., 2016]). The latter is the main topic of this paper. In the literature of exemplar-based synthesis, most effort has focused on texture synthesis, which can produce a large texture sample from a small observation, in an efficient manner. The exemplar texture is often assumed to be perceptually stationary, *i.e.* with no large geometric deformations nor changes in lighting transformations, and we adopt this stationary setting here. Texture models can be broadly classified between parametric [Zhu et al., 1998, Portilla and Simoncelli, 2000, Gatys et al., 2015] and non-parametric [Efros and Leung, 1999, Kwatra et al., 2005] models. Related applications include dynamic texture synthesis [Xia et al., 2014], texture inpainting [Galerne and Leclaire, 2017], 3D texture mapping [Gutierrez et al., 2018], texture interpolation [Yu et al., 2019], morphing [Bergmann et al., 2017], expansion [Zhou et al., 2018] or procedural generation [Henzler et al., 2019].

Preprint. Under review.

Motivation In this work, we consider the local representation of images obtained by the extraction of patches, which are small sub-images of size $s \times s$ (where s usually ranges from 3 to 16). This representation takes profit of the self-similarities of natural images, and subsequently lies at the core of very efficient image restoration methods [Buades et al., 2005, Zoran and Weiss, 2011, Lebrun et al., 2013]. It is particularly well adapted to textural content where structural redundancies can be exploited to form new textures with a simple iterative copy/paste procedure [Efros and Leung, 1999]. Besides, considering global statistics on patches was proven fruitful in designing efficient and stable texture models [Galerne et al., 2018]. In the following, we use optimal transport (OT) distances to compare patch empirical distributions in a relevant way.

While deep learning approaches have recently shown impressive performance for image synthesis [Karras et al., 2019], patch-based methods are still competitive when only a single image is available for training [Bergmann et al., 2017, Shaham et al., 2019], both considering the computational cost and the visual performance. Moreover, deep learning methods are still difficult to interpret, whereas patch-based models offer a better understanding of the synthesis process and its cases of success and failure. However, patch-based approaches suffer from three main limitations in practice. To begin with, patches are often processed independently and then combined to form a recomposed image. This leads to low frequency artifacts such as blurring because the patches overlap [Kwatra et al., 2005]. In addition, optimization has to be performed sequentially in a coarse-to-fine manner (both in image resolution and patch size) starting from a good initial guess. Last, global patch statistics must be controlled along the optimization to prevent strong visual artifacts [Kaspar et al., 2015]. We tackle all these aspects in the proposed OT framework.

Related work The model proposed in this paper falls into the scope of texture optimization, which formulates synthesis as the minimization of an energy (that may encode visual features or global statistics) starting from a random initialization. Such a framework can embrace famous parametric texture synthesis algorithms through a pixelwise optimization, *e.g.* [Portilla and Simoncelli, 2000] (which matches responses to a bank of complex steerable filters) and [Gatys et al., 2015] (which matches responses to a pre-learnt neural network). A related framework has been introduced in [Kwatra et al., 2005], with an energy that reflects, at multiple scales, the distances of patches from the synthesis to the ones of the exemplar. The corresponding synthesis algorithm consists in iterating patch nearest-neighbor projections in a coarse-to-fine manner. The main drawback of this energy is that it is oblivious of the global statistics of the output image, and thus exhibits trivial minima. This model has been improved in [Gutierrez et al., 2017] where discrete OT plans are iteratively used to enforce the patch distribution of the exemplar. Because of the cost of discrete OT, this algorithm has strong limitations in terms of computational time and output size. In contrast, the models of [Galerne et al., 2018, Leclaire and Rabin, 2019] are based on semi-discrete OT maps which can be estimated offline, and thus copes with these constraints. In parallel, several models based on Generative Adversarial Networks [Goodfellow et al., 2014] have been proposed, which allow for feed-forward synthesis of general images [Shaham et al., 2019] or texture images [Ulyanov et al., 2016, Bergmann et al., 2017] from a single image example. When considering the training of a generative network on an image dataset based on the minimization of the discrepancy between distributions, the latter introduction of Wasserstein GANs [Arjovsky et al., 2017] (WGAN) has offered an elegant solution to mode collapse issues.

Alternative techniques to train generative convolutional neural network (CNN) also took profit from the OT framework (*e.g.* Sliced-Wasserstein distance in the latent space of auto-encoder [Kolouri et al., 2019]). Although achieving state-of-the art performance, GANs suffer from some limitations. To begin with, GANs require to optimize a discriminative network, which makes the process unstable and requires a large number of additional parameters [Goodfellow et al., 2014, Mescheder et al., 2018]. However the dual formulation of Wasserstein-1 distances allows to restrict to 1-Lipschitz discriminative networks. Different strategies has thus been proposed to enforce such a constraint (*e.g.* weight clipping or gradient penalty [Gulrajani et al., 2017]), thus only approximating the true Wasserstein-1 distance. In [Chen et al., 2019], the optimisation of the Wasserstein distance in WGAN is driven by the semi-discrete formulation of OT between the discrete distribution of training images and the density of generated images. Finally, we note that the OT approach in Generative Networks is mainly considered for comparing distributions of generated images and not for prescribing statistics on a single synthesized image.

Contributions In this context, we propose to use OT to constrain the patch distribution of the synthesized image to be close to the one of a unique example image. By proposing a formulation that directly handles the patches of the generated image, our work implicitly addresses the aforementioned limitations of patch-based methods (aggregation, multi-scale optimization, and statistical control). The main contributions are (i) A new and versatile framework for image synthesis from a single example using multi-scale patch statistics and optimal transport; (ii) The computation of the gradient in the discrete and semi-discrete settings used to derive stable algorithms for image synthesis and generative network training from a single example; (iii) Application of the proposed framework for image optimization and generative network for texture synthesis and inpainting.

Outline The paper is organized as follows. To begin with, we recall the OT framework in Section 2. In Section 3, we propose a new model for image synthesis *via* explicit optimization of the discrete formulation of the Wasserstein distance between patch distributions, at different resolutions. Then, we consider in Section 4 the case where the synthesized images are generated by a convolutional neural network trained with a stochastic algorithm using the semi-discrete Wasserstein formulation. Finally, we discuss about qualitative evaluation of a texture synthesis method and we propose a new metric based on our framework for this purpose.

2 Background on Optimal Transport

Let $\mathcal{X}, \mathcal{Y}$ be two compact spaces included in $\mathbf{R}^d$ and $c : \mathcal{X} \times \mathcal{Y} \to \mathbf{R}$ be a continuous cost function. We consider two measures μ, ν supported on $\mathcal{X}, \mathcal{Y}$, respectively, and we denote by $\Pi(\mu, \nu)$ the probability distributions on $\mathcal{X} \times \mathcal{Y}$ having marginals μ and ν.

Definition 1 (OT cost and Wasserstein distance)**.** *The OT cost is defined by*

$$\mathrm{OT}_c(\mu, \nu) = \inf_{\pi \in \Pi(\mu,\nu)} \int c(x, y) d\pi(x, y). \tag{1}$$

If $c(x, y) = \|x - y\|^p$, $p \geq 1$, then $W_p(\mu, \nu) = \mathrm{OT}_c(\mu, \nu)^{\frac{1}{p}}$ defines the p-Wasserstein distance.

In the following, theoretical results will be formulated for a general OT cost but the experiments focus on the W_2 cost. Also, we will repeatedly use the dual formulation of OT.

Theorem 1 (Semi-dual formulation [Santambrogio, 2015])**.** *If $\mathcal{X}$ and $\mathcal{Y}$ are compact and the cost c is continuous, then*

$$\mathrm{OT}_c(\mu, \nu) = \max_{\varphi \in \mathcal{C}(\mathcal{Y})} \int \varphi^c(x) d\mu(x) + \int \varphi(y) d\nu(y), \tag{2}$$

where $\varphi : \mathcal{Y} \to \mathbf{R}$ and its c-transform is defined by $\varphi^c(x) = \min_{y \in \mathcal{Y}} [c(x, y) - \varphi(y)]$.

3 Image optimization

In this section we formulate an energy that will constrain the patch distribution of an image to be close to a target distribution for the OT cost. Using the semi-dual formulation of OT, we end up with a non-convex concave saddle-point problem. We then propose a pixelwise optimization algorithm for this energy, which exhibits a good empirical behavior (stability and convergence). Finally we extend this energy in a multiscale fashion in order to address texture synthesis, and we make the connection with the texture optimization framework of [Kwatra et al., 2005].

3.1 Setup of the problem

For an image $u \in \mathbf{R}^n$ with n pixels, we consider the collection of its patches $Pu = (P_1 u, \ldots, P_n u)$, that is, the list of all sub-images of size $s \times s$ extracted from u. To simplify, we consider periodic boundary conditions so that the number of patches is exactly n. Notice that P_j is a linear operator whose adjoint P_j^T maps a given patch q to an image whose j-patch is q and is zero elsewhere. Consider also the empirical patch distribution of an image $u \in \mathbf{R}^n$ defined by $\mu_u = \frac{1}{n} \sum_{i=1}^n \delta_{P_i u}$. Given an example texture image $v \in \mathbf{R}^m$, we aim at generating an image $u \in \mathbf{R}^n$ whose patch distribution μ_u is close to μ_v for the OT cost.

From the semi-dual formulation of Theorem 1, this amounts to minimize the function

$$w(u) = \mathrm{OT}_c\left(\mu_u, \mu_v\right) = \max_{\varphi \in \mathbf{R}^m} f(\varphi, u), \tag{3}$$

where

$$f(\varphi, u) = \frac{1}{n}\sum_{i=1}^{n} \varphi^c(P_i u) + \frac{1}{m}\sum_{j=1}^{m} \varphi_j. \tag{4}$$

Solving (3) is now equivalent to

$$\min_{u \in \mathbf{R}^n} \max_{\varphi \in \mathbf{R}^m} f(\varphi, u). \tag{5}$$

From the OT theory [Santambrogio, 2015], we know that f is concave in φ which will be helpful in practice. However, the function f is not convex in the second variable u, and thus we will focus on approaching local minima of w in (3).

3.2 Theoretical results

In this paragraph, we will study the differential properties of the function f introduced in (3). The first result, stated without proof, is a standard derivation of the gradient w.r.t. φ.

Theorem 2. *Let $u \in \mathbf{R}^n$ and $\varphi \in \mathbf{R}^m$. For any i, let $j^\star(i) \in \arg\min_j c(P_i u, y_j) - \varphi_j$ (with an arbitrary choice in case of* ex-aequo*). Then, $f(\cdot, u)$ admits a super-gradient at φ given by*

$$g_j(\varphi, u) = \frac{1}{m} - \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}_{j^\star(i)=j} \tag{6}$$

The regularity in u is essentially linked to the differential property of the Wasserstein distance w.r.t. its arguments, which is here linked to the positions of points in μ_u. To analyze the situation, we consider the open Laguerre cells $L_j(\varphi) = \{x \mid \forall k \neq j,\ c(x, y_j) - \varphi_j < c(x, y_k) - \varphi_k\}$. For any $\sigma : \{1, \dots, n\} \to \{1, \dots, m\}$, denote $\mathcal{A}_\sigma(\varphi) = \{u \in \mathbf{R}^n \mid \forall i, P_i u \in L_{\sigma(i)}(\varphi)\}$ and $\mathcal{A}(\varphi) = \sqcup_\sigma \mathcal{A}_\sigma(\varphi)$. In other words, $\mathcal{A}(\varphi)$ is the set of images whose patches have no *ex-aequo* values for φ^c. Since c is continuous, it is straightforward to see that $\mathcal{A}_\sigma(\varphi)$ is an open subset of $\mathbf{R}^n$.

Theorem 3. *Assume that c is differentiable w.r.t. the first variable. Let $\varphi \in \mathbf{R}^m$ and $u \in \mathcal{A}_\sigma(\varphi)$. Then $f(\varphi, \cdot)$ is differentiable at u and*

$$\nabla_u f(\varphi, u) = \frac{1}{n}\sum_{i=1}^{n} P_i^T \nabla_u c(P_i u, y_{\sigma(i)}). \tag{7}$$

Proof. Since $\mathcal{A}_\sigma(\varphi)$ is open, for v sufficiently close to u, we have also that $v \in \mathcal{A}_\sigma(\varphi)$ and thus

$$f(\varphi, v) = \frac{1}{n}\sum_{i=1}^{n} \left(c(P_i v, y_{\sigma(i)}) - \varphi_{\sigma(i)}\right) + \frac{1}{m}\sum_{j=1}^{m} \varphi_j \tag{8}$$

and the result follows by applying the chain-rule to the first term. □

Finally, the following theorem provides the gradient of w at particular points u where the associated optimal dual potential φ^* leads to no *ex-aequo*.

Theorem 4. *Let $u \in \mathbf{R}^n$ and $\varphi^\star \in \arg\max_\varphi f(\varphi, u)$ such that $u \in \mathcal{A}(\varphi^\star)$. Then w in* (3) *is differentiable at u and $\nabla w(u) = \nabla_u f(\varphi^\star, u)$.*

Proof. Since $u \in \mathcal{A}(\varphi^\star)$ there exists a map $\sigma : \{1, \dots, n\} \to \{1, \dots, m\}$ such that $u \in \mathcal{A}_\sigma(\varphi^\star)$. Since $\varphi^\star$ is optimal and its c-transform makes no *ex-aequo* for the patches $P_i u$, we have in particular that the map $P_i u \mapsto y_{\sigma(i)}$ realizes the OT from μ_u to ν. Then, for v close to u, we still have $v \in \mathcal{A}_\sigma(\varphi^\star)$ and thus the same map also realizes the OT from μ_v to ν, which implies that $\varphi^\star \in \arg\max_\varphi f(\varphi, v)$. Therefore for v close to u, we have $w(v) = f(\varphi^\star, v)$ which suffices to conclude. □

Let us emphasize that the hypothesis of Theorem 4 can be true only in very specific cases. Indeed, the fact that $P_i u \mapsto y_{\sigma(i)}$ is an OT map from μ_u to ν implies, from mass conservation, that m divides n. Fortunately for the applications at hand, such condition is easily met as one can sample the target patch distribution accordingly. In addition, we only use the gradient $\nabla_u f(\varphi, u)$ during optimization.

3.3 A one-scale texture synthesis algorithm

We now detail our texture synthesis algorithm that estimates an image u minimizing (3), so that the patch distribution of u is close to the one of an example image v. To do so, we look for a local saddle point of the problem (5) with an iterative alternate scheme on φ and u, starting with an initial noise image u^0. For a fixed u^k, we perform a gradient ascent, using the super-gradient given in Theorem 2 to obtain an approximation φ^{k+1} of $\varphi^* \in \arg\max_\varphi f(\varphi, u^k)$. A gradient-descent step is then realized on u, using the gradient given in Theorem 3. In practice we consider the Adam optimizer [Kingma and Ba, 2014] with learning rate 0.01. For texture synthesis, in all the experiments, we consider the quadratic cost $c(x, y) = \frac{1}{2}\|x - y\|^2$. In this particular case, the gradient w.r.t. u reads

$$\nabla_u f(\varphi^{k+1}, u^k) = \frac{1}{n}\left(\sum_{i=1}^{n} P_i^T P_i u^k - \sum_{i=1}^{n} P_i^T y_{j_i^{k+1}}\right), \tag{9}$$

where

$$j_i^{k+1} = \arg\min_j \frac{1}{2}\|P_i u^k - y_j\|^2 - \varphi_j^{k+1}. \tag{10}$$

In (9), notice that $\sum_{i=1}^n P_i^T$ corresponds to an uniform patch aggregation. To simplify, we consider periodic conditions for patch extraction, so that $\sum_{i=1}^n P_i^T P_i = p\mathrm{I}$, where $p = s \times s$ denotes the number of pixels in the patches. Hence, from (9) and considering a step size $\eta\frac{n}{p}$, $\eta > 0$, the update of u through gradient descent can be formulated as:

$$u^{k+1} = (1 - \eta)u^k + \eta v^k, \tag{11}$$

where

$$v^k = \frac{1}{p}\sum_{i=1}^{n} P_i^T y_{j_i^{k+1}}, \tag{12}$$

can be interpreted as the image formed with the patches from the exemplar image v which are the nearest neighbor to the patches of u^k in the sense of (10). The gradient step then mixes the current image u^k with v^k. In the case $\varphi = 0$, the minimum in (10) is reached by associating to each patch of u^k its nearest neighbor in the set $\{y_1, \ldots, y_n\}$ patches of v, which exactly corresponds to the texture synthesis algorithm proposed in [Kwatra et al., 2005].

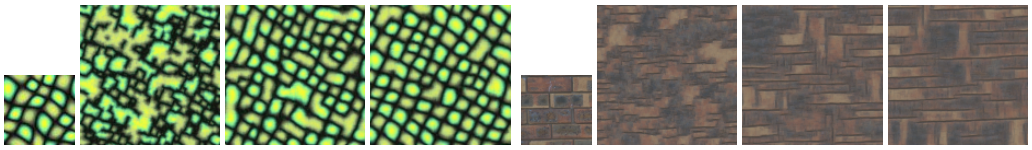

Figure 1: Influence of patch-size s for the single-scale method (Algorithm 1 with $L = 1$).

This image synthesis process is illustrated in Figure 1, with a comparison of image synthesis for different patch sizes. With this formulation, patch distributions are only considered for one given patch size $s \times s$. Therefore, this method cannot take into account variations that may occur at scales larger than s, a limitation which is overcome by the multi-scale extension introduced in the next section.

3.4 Multi-scale Texture Generation

In the patch-based literature, a common way to deal with multi-scale is to create a pyramid of down-sampled and blurred images. Let S_l be an operator that creates a down-sampled and blurred version of an image for a scale $l \in \{1, ..., L\}$. The image at scale l is of size $n/2^{l-1} \times n/2^{l-1}$ and denoted $u_l = S_l(u)$. The multi-scale texture synthesis is then obtained by minimizing

$$\mathcal{L}(u) = \sum_{l=1}^{L} \max_\varphi f_{v^l}(u^l, \varphi), \tag{13}$$

Algorithm 1 Multi-scale Texture synthesis procedure

Input: target image v, initial image u_0, learning rates η_u and η_φ, number of iterations N_u and N_φ, number of scales L
Output: image u
$u \leftarrow u_0$ and $\varphi_l = 0$ for $l = 1 \dots L$
for $k = 1$ **to** N_u **do**
  for $l = 1$ **to** L **do**
    for $j = 1$ **to** N_φ **do**
      compute g_l a super-gradient for scale l
      $\varphi_l \leftarrow \varphi_l + \eta_\varphi g_l(\varphi_l, S_l(u))$
    end for
    $G_l(u, \varphi_l) \leftarrow (D_u S_l(u))^T \nabla_u f_{v_l}(\varphi_l, S_l(u))$
  end for
  $u \leftarrow u - \eta_u(k) \sum_{l=1}^{L} G_l(u, \varphi_l)$ we
end for

where

$$f_{v^l}(u^l, \varphi) = \frac{1}{n} \sum_{j=1}^{n} \min_i \left[c(P_j u^l, P_i v^l) - \varphi_i \right] + \frac{1}{m} \sum_{i=1}^{m} \varphi_i. \tag{14}$$

This loss is the sum of OT costs defined on different scales of u. In the case where the cost is $\| \cdot \|^p$, this loss can be interpreted as the sum of W_p^p distances. In this work, we set $p = 2$ for the experiments since it lowers the computation time using the fact that $\|x - y\|^2 = \|x\|^2 + \|y\|^2 - \langle y|x \rangle$. However the method would still work for any continuous cost c. As for the single-scale case, an alternate scheme is considered to minimize the loss $\mathcal{L}$ in (13). Considering smoothing operators S_l, with differential at u given by $D_u S_l(u)$, the gradient descent update of u relies on the following relation combining different scales:

$$\nabla_u \mathcal{L}(u) = \sum_{l=1}^{L} (D_u S_l(u))^T \nabla_u f_{v^l}(u^l, \varphi_l^\star) = \sum_{l=1}^{L} G_l(u, \varphi_l^\star). \tag{15}$$

3.5 Experiments

The multi-scale synthesis process is presented in Algorithm 1. Figures 2 and 3 present comparisons for synthesis realized from 128×128 and 256×256 exemplar textures with the two methods [Kwatra et al., 2005] (patch size ranging from $s = 32$ to $s = 8$) and [Gatys et al., 2015] (VGG-19 features). In the same way than our Algorithm 1, these methods aim at optimizing the pixels of an image in order to constrain some information: the patches in a nearest neighboor way for [Kwatra et al., 2005] and the Gram matrices of VGG features for [Gatys et al., 2015]. We also show additional results of texture synthesis in Figures 4 obtained with Algorithm 1.

For these experiments, we used the original code of [Gatys et al., 2015] (`https://github.com/leongatys/DeepTextures`) and our own implementation of Texture Optimization [Kwatra et al., 2005]. With our method, the evolution of the loss function is also drawn to illustrate the numerical stability of the optimization scheme. The value $f(\varphi^{k+1}, u^k)$, which is an approximation of $\sum_l W_2^2(\mu_{u_l^k}, \nu_l)$ between the distributions of patches from u^k and v at all scales $l = 1 \cdots L$, is almost always decreasing along the iterations k.

While it is already known [Liu et al., 2016] that the approach of [Gatys et al., 2015] might have color inconsistencies, it mostly suffers here from the small resolution of the input for which it is difficult to extract deep features. Additionally, one can observe that contrarily to our method and [Gatys et al., 2015], the approach of [Kwatra et al., 2005] does not rely on statistics and thus does not respect the distribution of features of the original sample. Therefore, it must be initialized with a good guess (permutation of patches) instead of any random image.

In all experiments, we used the `Adam` [Kingma and Ba, 2014] optimizer from the `torch.optim` package of PyTorch in order to perform the gradient descent for the image variable u in Algorithm 1.

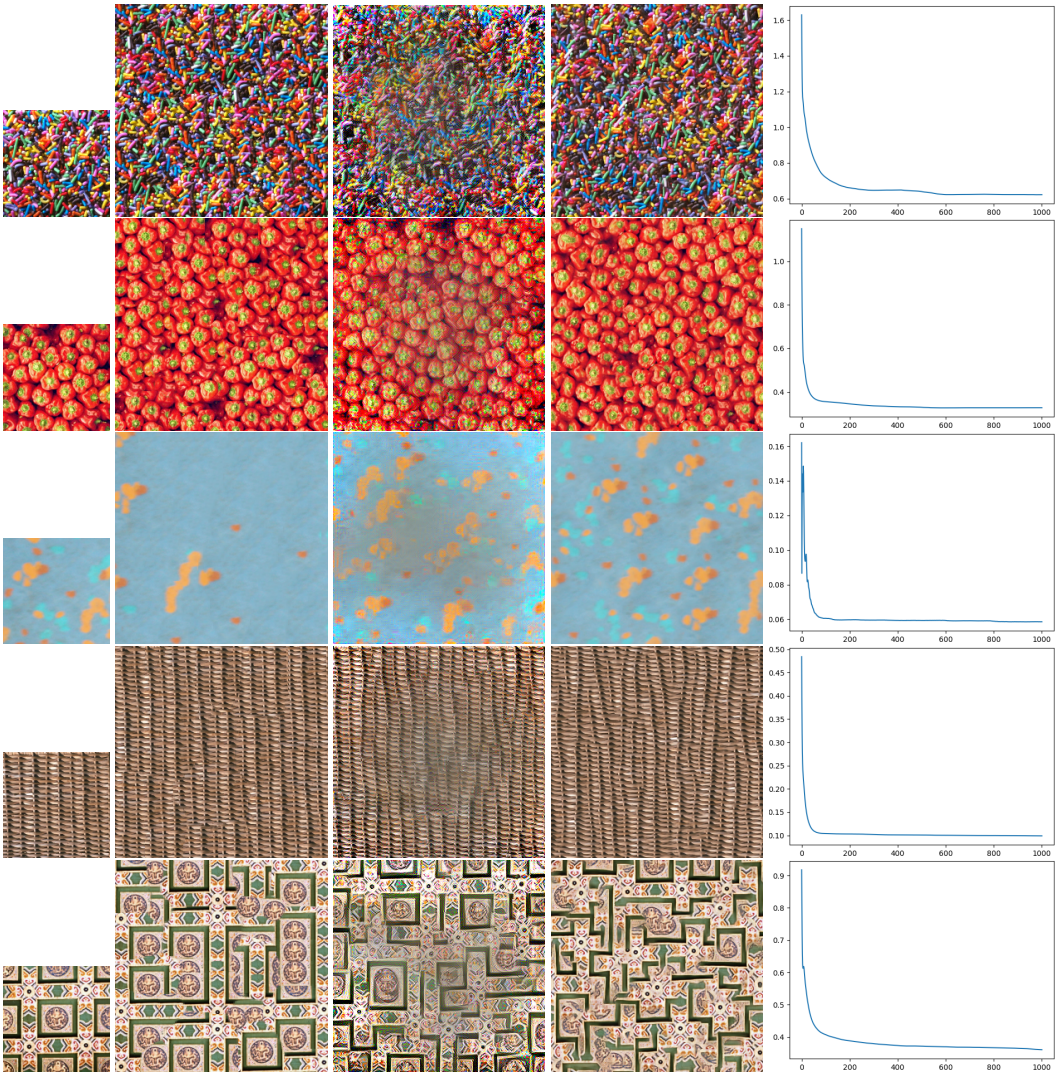

Figure 2: Texture synthesis from a 128×128 sample by image optimization. Comparison of our multi-scale approach using $s = 4$ and $L = 4$ (see Alg. 1) with Kwatra [Kwatra et al., 2005] (patch size ranging from $s = 32$ to $s = 8$) and Gatys [Gatys et al., 2015] (VGG-19 features).

We also used the KeOps library [Charlier et al., 2020] that performs efficiently the nearest neighbor projections involved in the computations of c-transforms.

Additionally, we used the following parameters:

- patch size $s = 4$;
- 100 iterations of gradient ascent on φ with gradient step $\frac{0.8}{\sqrt{k}}$ at iteration k;
- 200 iterations of gradient descent on the image with `Adam` optimizer with learning rate $lr = 0.05$;
- the number of scales $L = \log_2(R) - 3$ depends on the size R of the original image:
  - $L = 4$ for 128×128 images
  - $L = 5$ for 256×256 images
  - $L = 6$ for 512×512 images

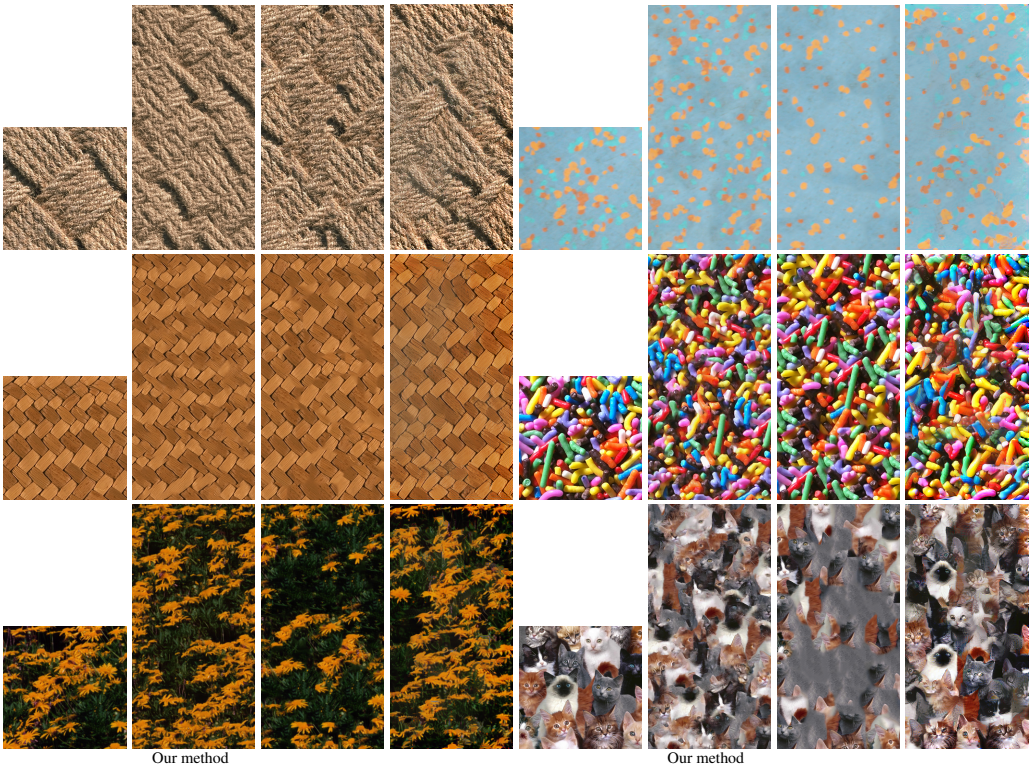

Figure 3: Texture synthesis from a 256×256 sample by image optimization. Comparison of our multi-scale approach using $s = 4$ and $L = 5$ (see Alg. 1) with [Kwatra et al., 2005] (patch size ranging from $s = 32$ to $s = 8$) and [Gatys et al., 2015] (VGG-19 features).

Verbatim copies We can notice that the proposed method yields results with some verbatim copies. This well-known phenomenon actually occurs for other image optimization approaches and can be observed for instance in the experiments of [Gatys et al., 2015] and [Kwatra et al., 2005]. This is also expected in our case since we only enforce the patch distribution and parts of the original image are local minima for the proposed loss. Although these local copies can clearly be observed when we have few samples, they become less obvious when the algorithm has access to more samples. To illustrate this point, Figure 5 shows synthesis results for the same texture, first from a 128×128 crop of the example texture, then from the full 256×256 texture.

Texture inpainting The framework proposed for texture synthesis can be extended to texture inpainting, by taking the patches outside a masked area as the target ones. By optimizing only the pixels within the masked area, the very same algorithm yields an efficient texture inpainting method, as illustrated in Figure 6.

Discussion As previously mentioned, the proposed approach can be seen as a generalization of the iterative texture synthesis method [Kwatra et al., 2005]. The visual results are competitive with the state-of-the-art of patch-based synthesis methods. The process requires 1 second to run one iteration for a 256×256 image on a GPU Nvidia K40m, and 150 iterations are enough to reach a good visual result, as illustrated by convergence plots in Figure 2. However, the whole optimization has to be done each time a new image is synthesized. In order to define a versatile algorithm that can generate new samples on the fly, we will now rely on continuous generative models and convolutional neural networks.

Figure 4: Texture synthesis of size 256×256 from a 128×128 sample by image optimization using our multi-scale approach with $s = 4$ and $L = 4$ (see Alg. 1).

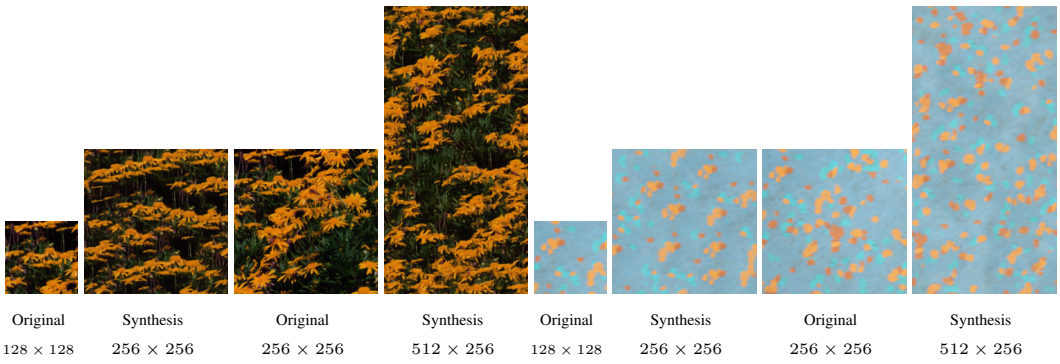

Figure 5: Comparison when using input images with different sizes, effect on verbatim copies. The 128×128 original image is a crop from the 256×256 original image. Since our method is patch-based, it is likely that the synthesized texture have local copies of the original one. This effect is more obvious when the input image is small, see for instance the same flower copied many times in the 256×256 version of the second image.

Figure 6: Texture inpainting results on various masked textures of size 128×128 with $s = 4$ and $L = 3$ using a slightly adapted version of Algorithm 1.

4 Generative Network

In this section, we consider the problem of training a network to generate images that have a prescribed patch distribution at multiple scales.

4.1 The semi-discrete formulation

Let us consider a generator g_θ, parameterized by $\theta \in \Theta \subset \mathbf{R}^d$ that goes from a latent space $\mathcal{Z}$, assumed to be compact, to the space of images. Given a random vector Z of distribution ζ in the latent space $\mathcal{Z}$, the generated random image is $g_\theta(Z)$, and has distribution $g_\theta \sharp \zeta$ (defined for any Borel set A by $g_\theta \sharp \zeta(A) = \zeta(g_\theta^{-1}(A))$). In the following, we will assume that for $d\zeta$-almost all z, $\theta \mapsto g_\theta(z)$ is differentiable and we denote by $J_\theta g_\theta(z)$ the corresponding Jacobian matrix. The patches of the output image are distributed according to

$$\mu_\theta = \frac{1}{n} \sum_{i=1}^{n} P_i \sharp g_\theta \sharp \zeta. \tag{16}$$

One can assume that μ_θ is absolutely continuous w.r.t. the Lebesgue measure; this is indeed the generic case, which happens except for very specific choices of g_θ and ζ. Since the target distribution ν is still discrete, computing $W_p(\mu_\theta, \nu)$ falls into the semi-discrete case of OT. The semi-dual formulation of OT then writes

$$\mathrm{OT}_c(\mu_\theta, \nu) = \max_{\varphi} F(\varphi, \theta) \quad \text{where} \quad F(\varphi, \theta) = \mathbf{E}_{Z \sim \zeta} \left[\frac{1}{n} \sum_{i=1}^{n} \varphi^c(P_i g_\theta(Z)) + \frac{1}{m} \sum_{j=1}^{m} \varphi_j \right] \tag{17}$$

Note that F can be related to the function f we introduced in the previous section, since

$$F(\varphi, \theta) = \mathbf{E}_{Z \sim \zeta} \left[f(\varphi, g_\theta(Z)) \right]. \tag{18}$$

Therefore we can consider the gradients of f w.r.t. u or φ as stochastic gradients for F. Indeed, from Theorem 2 we get that $g(\varphi, g_\theta(Z))$ is a stochastic super-gradient w.r.t. φ, and from Theorem 3 we get

$$\nabla_\theta f(\varphi, g_\theta(z)) = J_\theta \left(g_\theta(z) \right)^T \nabla_u f(\varphi, u = g_\theta(z)), \tag{19}$$

which exists as soon as $\varphi, g_\theta(z)$ satisfies the hypothesis of Theorem 3. Notice that, except in degenerate cases, this hypothesis is likely to be true (*e.g.* for the Euclidean cost, it will be true as soon as $g_\theta \sharp \zeta$ is absolutely continuous w.r.t. the Lebesgue measure).

In this semi-discrete framework of OT, $\theta \mapsto \mathrm{OT}_c(\mu_\theta, \nu)$ is expected to be smoother than in the discrete case. We do not provide a precise result here. However, if θ is a point of differentiability, the envelope theorem ensures that

$$\nabla_\theta \mathrm{OT}_c(\mu_\theta, \nu) = \nabla_\theta \mathbf{E}[f(\varphi^\star, g_\theta(Z))] \tag{20}$$

where $\varphi^\star$ is an optimal dual variable for the OT from μ_θ to ν, and if we can differentiate under the expectation, we get

$$\nabla_\theta \mathrm{OT}_c(\mu_\theta, \nu) = \mathbf{E}[\nabla_\theta f(\varphi^\star, g_\theta(Z))]. \tag{21}$$

4.2 Proposed algorithm

All the previous considerations together with the multiscale formalism we defined in Section 3 lead us to propose the Algorithm 2 for minimizing the following loss w.r.t. the parameters θ

$$L(\theta) = \sum_{l=1}^{L} \max_{\varphi_l} \mathbf{E}_{z \sim \zeta} \left[f_{v^l}(\varphi_l, g_\theta(z)^l) \right]. \tag{22}$$

In practice, for each iteration k and at each layer l we first update φ_l as proposed in [Genevay et al., 2016]. Then we sample an image and perform a stochastic gradient step in θ. The fully convolutional neural network designed for texture generation in [Ulyanov et al., 2016] is taken for g_θ. The main advantage of convolutional networks is that, once learnt, they can generate arbitrarily large images. In our PyTorch implementation, we use the Adam optimizer to estimate the parameters θ. We run the algorithm for 10000 iterations with a learning-rate 0.0005. An averaged stochastic gradient ascent (ASGA) with 10 inner iterations is used for computing $\varphi^\star$. In this setting, 30' are required to train our generator with a GPU Nvidia K40m.

Algorithm 2 Multi-scale Convolutional Texture generation with stochastic gradient descent

Input: target image v, initial weight θ_0, learning rate η_θ, number of iterations N_u and N_φ, number of scales L
Output: generator parameters θ
for $k = 1$ **to** N_u **do**
  for $l = 1$ **to** L **do**
    compute $\varphi_l^\star$ with ASGA
    sample z from ζ
    $G_l(\theta) \leftarrow \nabla_\theta f_{v_l}(\varphi_l^\star, g_\theta(z)_l)$
  end for
  $\theta \leftarrow \theta - \eta_\theta(k) \sum_{l=1}^{L} G_l(\theta)$
end for

4.3 Experimental results and discussions

In Figure 7 , comparisons with two methods are presented for input textures of size 128×128. We first consider the Texture Networks method [Ulyanov et al., 2016], which consists in training a generative network using VGG-19 feature maps computed on a texture sample. Note that the very same CNN architecture has been used for our model. We also compare to SinGAN [Shaham et al., 2019], a recent GAN technique to generate images from a single example relying on patch sampling.

Figure 8 then presents more detailed comparisons for input textures of size 256×256, with two additional synthesis methods from the literature: PSGAN [Bergmann et al., 2017] which is a previous approach applying GAN framework to the learning of a CNN, and Texto [Leclaire and Rabin, 2019] which also constrains patch distributions with OT. In all our experiments, we used the official PyTorch implementations of SINGAN (`github.com/tamarott/SinGAN`), PSGAN (`github.com/zalandoresearch/famos`) and Texture Networks (`github.com/JorgeGtz/TextureNets_implementation`), with their default parameters. For all the experiments of this section realized with Algorithm 2, we used the same generative network than the one proposed in [Ulyanov et al., 2016] with the following parameters

- patch size $s = 4$
- 10 iterations of stochastic gradient ascent on φ with gradient step $\frac{0.8}{\sqrt{k}}$ with randomly generated images of size 256×256 at each iteration
- 10000 iterations of gradient descent on the parameters of the generator G with *Adam* optimizer with learning rate $lr = 0.0005$
- number of scales $L = 5$

Figure 7 shows that small sample resolution (128×128) makes difficult the extraction of relevant deep features, and impacts negatively the quality of Texture Networks. On the other hand, the two other methods based on patches are more robust and give satisfying results. It is interesting to notice that, for all generators, some specific patterns are learnt and repeated across the generated images. More specifically, when using patches, those patterns are almost local copies of the original image. For instance, we can observe on the radishes image in Figure 7 that one particular leave seems to appear many times in our case, and patches on the border of images generated with SinGAN are copies from the input. This seems natural considering that the optimization is only performed on sampled patches, but it also shows that there is room for improvement.

As illustrated in Figure 8, the results obtained with our method are visually close to the ones from Texto [Leclaire and Rabin, 2019] which also minimizes OT distance between patch distribution. However, the patch-aggregation step from Texto makes the results blurrier than our method since we overcome the aggregation issue by design. Although Texture networks [Ulyanov et al., 2016] produce textures that look sharper than our results, they may fail to reconstruct larger structures as in the fourth image. Observe as well that patch-based networks are less likely to create visual artifacts (checkerboard patterns, false colors, etc). These experiments show that enforcing the patch distribution permits to learn convolutional neural network for the purpose of texture generation. Despite the already nice results we obtained here, we think that more thorough investigation on the network architecture and the learning parameters could yield qualitatively better results.

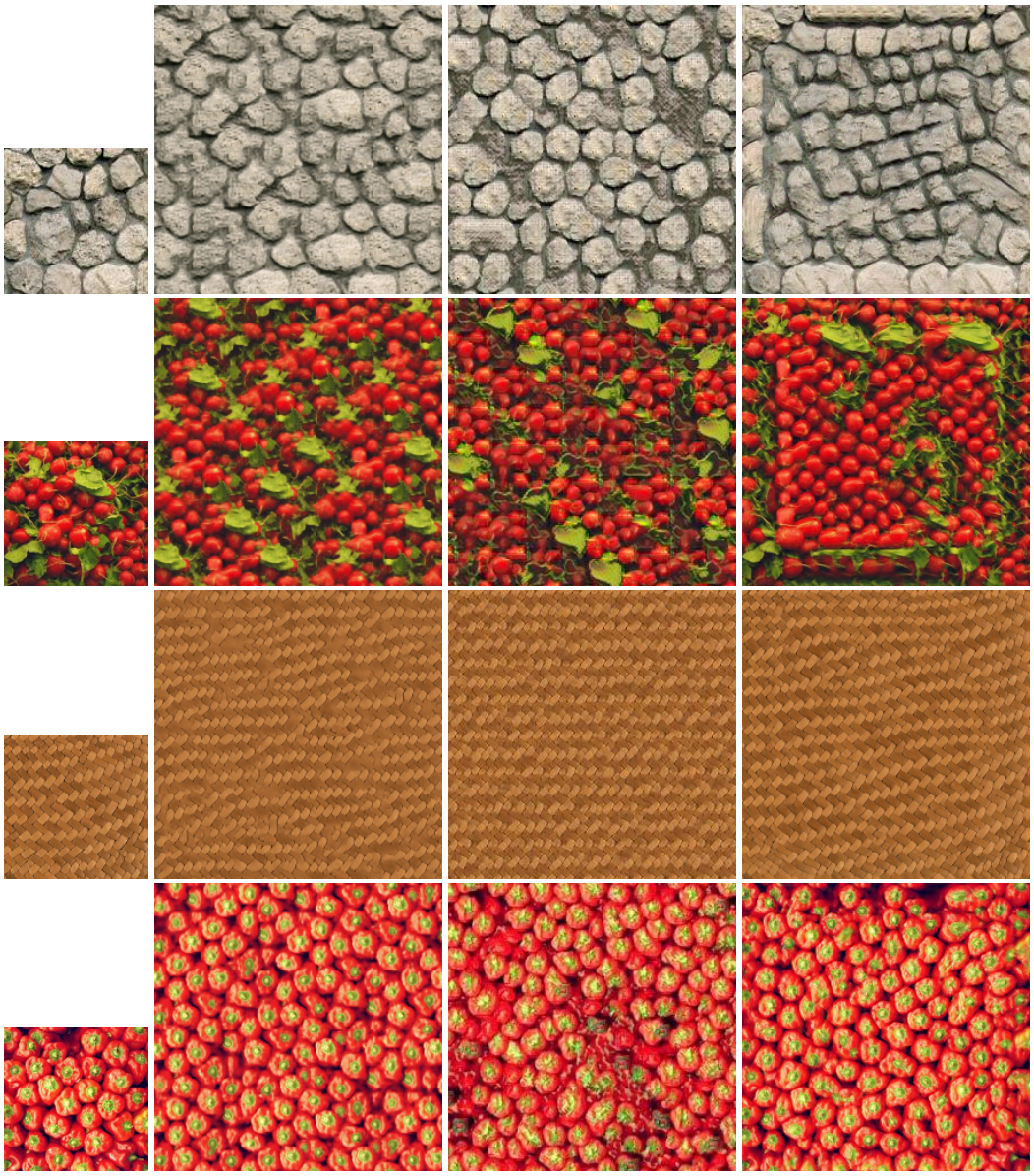

Figure 7: Texture synthesis from a generative network trained on a single 128×128 sample. Comparison of our multi-scale approach Alg. 2 (with $s = 4$, $L = 5$) with [Ulyanov et al., 2016] (using VGG-19 features) and SinGAN [Shaham et al., 2019].

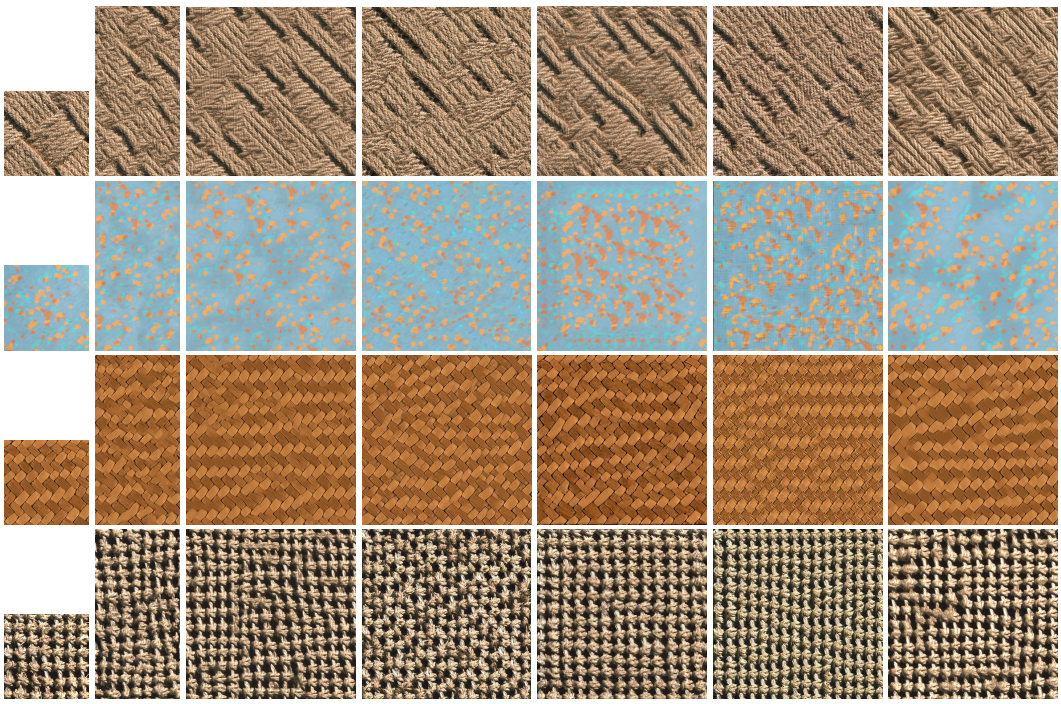

Figure 8: Texture synthesis from a single 256×256 sample of a high-definition texture. Comparison of Alg. 1 and Alg. . 2 (with $s = 4$ and $L = 5$) with [Ulyanov et al., 2016] (using VGG-19 features), SinGAN [Shaham et al., 2019], PSGAN [Bergmann et al., 2017] and Texto [Leclaire and Rabin, 2019].

4.4 Evaluation of texture synthesis methods

The precise evaluation of texture synthesis methods is a complex and open question. The visual quality is strongly subjective and there exists no universal perception metric. Some metrics have been proposed to evaluate generated images, with for instance the SIFID (Single Image FID) introduced in the SINGAN paper [Shaham et al., 2019] or the metric given by the feature correlations of VGG as used in [Gatys et al., 2015]. We give in Table 1 the scores for these two metrics corresponding to the results presented in Figure 8. As expected, these metrics favor the methods that actually optimize the given features. Therefore, we also propose a new evaluation metric for texture synthesis that measures the Wasserstein distance between patch distribution at each scale, which should favor our method. The corresponding results are also given in Table 1. As expected, our methods and the Texto one, which also minimizes OT distance between patches distribution, perform better for the proposed OT loss, whereas the Texture Networks [Ulyanov et al., 2016] obtain better results with the metric based on VGG features and inception network. However, our methods are competitive with the state-of-the art methods for all these metrics since we obtain the best results for two of the four proposed textures with Algorithm 1. Moreover, we can notice that existing metrics are not always relevant. In the last texture of Figure 8, [Ulyanov et al., 2016] presents smaller values for both SIFID and VGG score, whereas the synthesized texture does not match the input one in term of large-scale coherence.

5 Conclusion

In this paper, we propose an original image generation framework from a single sample, which combines three popular topics in image processing and machine learning: patch-based methods, optimal transport and deep generative models. For both pixel optimization and convolutional network optimization, we achieve results comparable to the state-of-the-art in texture synthesis. The flexibility of our method also allowed us to propose a texture inpainting algorithm.

Table 1: Evaluation of texture synthesis methods for the results of Figure 8. The best result is given in bold font and the second best is underlined. The last line shows the mean over all images.

SIFID (first max pooling features) proposed in SINGAN [Shaham et al., 2019]

	Our (Alg. 1)	Our (Alg. 2)	Ulyanov	SINGAN	PSGAN	Texto
	0.43	1.13	**0.11**	0.93	<u>0.27</u>	1.22
	**0.02**	<u>0.06</u>	0.08	0.10	0.91	0.07
	**0.08**	0.18	0.18	<u>0.17</u>	1.14	0.18
	1.23	1.82	**0.17**	<u>0.37</u>	0.49	1.67
	0.44	0.80	**0.14**	<u>0.39</u>	0.70	0.79

VGG score (cross-correlation of VGG features as in [Gatys et al., 2015] and Ulyanov)

($\times 10^3$)	Our (Alg. 1)	Our (Alg. 2)	Ulyanov	SINGAN	PSGAN	Texto
	**122**	233	<u>218</u>	299	224	260
	**6**	19	9	<u>8</u>	512	24
	<u>141</u>	151	**54**	207	753	152
	865	922	**190**	<u>394</u>	1366	1030
	283	331	**118**	<u>227</u>	714	367

Proposed: Multiscale OT distance

	Our (Alg. 1)	Our (Alg. 2)	Ulyanov	SINGAN	PSGAN	Texto
	**0.45**	<u>0.48</u>	0.65	0.54	0.68	0.49
	**0.15**	<u>0.16</u>	0.24	0.24	0.43	<u>0.16</u>
	**0.09**	<u>0.10</u>	0.17	0.26	0.34	0.11
	**0.74**	0.78	1.22	0.79	1.19	<u>0.75</u>
	0.36	<u>0.38</u>	0.57	0.46	0.66	<u>0.38</u>

Dealing only with the patch distribution of the synthesized textures through OT costs, the pixel optimization method relates to a classical patch-based model [Kwatra et al., 2005]. In many patch-based synthesis methods, the different scales and the aggregation of the patches are treated separately. The strength of our approach then comes from the integration of the different synthesis steps into a single gradient descent on the image. Notice that we have restricted conditions regarding the number of target patches on the existence of the functional gradient. However, this is not a limitation for considered applications and numerical experiments show a clear convergence.

Still considering the optimal transport framework, we then propose to learn a convolutional neural network for texture synthesis. Our method produces images of arbitrary size, whose patch distributions at various scales are close in 2-Wasserstein distance to the ones of the reference image used for training. This is a significant advantage over several methods in the literature relying on the VGG-19 deep network [Simonyan and Zisserman, 2014], which then require a more complex optimization and a larger memory capacity.

This work tries to make the best of two worlds: on one hand, it makes use of generative models based on CNN which yield state-of-the-art performance, but are usually difficult to train (requiring large dataset, relying on adversarial unstable optimization, prone to mode collapse, etc); on the other hand, our model explicitly uses a patch based representation combined with an optimal transport distance which makes it possible to train on a single image and guarantees the quality of the generated images.

Our model can be framed as an exact Wasserstein-GAN model for patch, without the limitation of parameterizing the discriminator, nor the need for regularization (WGAN-GP). When optimizing

directly on an image, i.e. without the limitation of the generative network, the generated images are local copies of the exemplar one. This demonstrates the limitations of high-capacity GANs that can overfit training datasets, which raises interesting questions about privacy.

Additionally, as already shown with inpainting examples, the proposed approach has many direct applications in computer graphics and vision, and can be helpful for artists or designers (texture generation in video games, image restoration).

Acknowledgements

This study has been carried out with financial support from the French State, managed by the French National Research Agency, through the GOTMI project (ANR-16-CE33-0010-01). The authors also acknowledge the French GdR ISIS through the support of the REMOGA project.

References

[Arjovsky et al., 2017] Arjovsky, M., Chintala, S., and Bottou, L. (2017). Wasserstein generative adversarial networks. In *International Conference on Machine Learning*, pages 214–223.

[Bergmann et al., 2017] Bergmann, U., Jetchev, N., and Vollgraf, R. (2017). Learning texture manifolds with the periodic spatial gan. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 469–477. JMLR. org.

[Buades et al., 2005] Buades, A., Coll, B., and Morel, J.-M. (2005). A non-local algorithm for image denoising. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 60–65. IEEE.

[Charlier et al., 2020] Charlier, B., Feydy, J., Glaunès, J. A., Collin, F.-D., and Durif, G. (2020). Kernel operations on the gpu, with autodiff, without memory overflows. *arXiv preprint arXiv:2004.11127*.

[Chen et al., 2019] Chen, Y., Telgarsky, M., Zhang, C., Bailey, B., Hsu, D., and Peng, J. (2019). A gradual, semi-discrete approach to generative network training via explicit wasserstein minimization. *arXiv preprint arXiv:1906.03471*.

[Efros and Leung, 1999] Efros, A. A. and Leung, T. K. (1999). Texture synthesis by non-parametric sampling. In *IEEE International Conference on Computer Vision*, page 1033.

[Galerne and Leclaire, 2017] Galerne, B. and Leclaire, A. (2017). Texture inpainting using efficient gaussian conditional simulation. *SIAM Journal on Imaging Sciences*, 10(3):1446–1474.

[Galerne et al., 2018] Galerne, B., Leclaire, A., and Rabin, J. (2018). A texture synthesis model based on semi-discrete optimal transport in patch space. *SIAM Journal on Imaging Sciences*, 11(4):2456–2493.

[Gatys et al., 2015] Gatys, L., Ecker, A. S., and Bethge, M. (2015). Texture synthesis using convolutional neural networks. In *NIPS*, pages 262–270.

[Genevay et al., 2016] Genevay, A., Cuturi, M., Peyré, G., and Bach, F. (2016). Stochastic optimization for large-scale optimal transport. In *Advances in neural information processing systems*, pages 3440–3448.

[Goodfellow et al., 2014] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680.

[Gulrajani et al., 2017] Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. C. (2017). Improved training of wasserstein gans. In *Advances in neural information processing systems*, pages 5767–5777.

[Gutierrez et al., 2017] Gutierrez, J., Galerne, B., Rabin, J., and Hurtut, T. (2017). Optimal patch assignment for statistically constrained texture synthesis. In *Scale-Space and Variational Methods in Computer Vision*.

[Gutierrez et al., 2018] Gutierrez, J., Rabin, J., Galerne, B., and Hurtut, T. (2018). On demand solid texture synthesis using deep 3d networks. In *Computer Graphics Forum*. Wiley Online Library.

[Henzler et al., 2019] Henzler, P., Mitra, N. J., and Ritschel, T. (2019). Learning a neural 3d texture space from 2d exemplars. *arXiv preprint arXiv:1912.04158*.

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

[Karras et al., 2019] Karras, T., Laine, S., and Aila, T. (2019). A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4401–4410.

[Kaspar et al., 2015] Kaspar, A., Neubert, B., Lischinski, D., Pauly, M., and Kopf, J. (2015). Self tuning texture optimization. In *Computer Graphics Forum*, volume 34, pages 349–359. Wiley Online Library.

[Kingma and Ba, 2014] Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

[Kolouri et al., 2019] Kolouri, S., Pope, P. E., Martin, C. E., and Rohde, G. K. (2019). Sliced wasserstein auto-encoders. In *International Conference on Learning Representations*.

[Kwatra et al., 2005] Kwatra, V., Essa, I., Bobick, A., and Kwatra, N. (2005). Texture optimization for example-based synthesis. In *ACM SIGGRAPH 2005 Papers*, pages 795–802.

[Lebrun et al., 2013] Lebrun, M., Buades, A., and Morel, J.-M. (2013). A nonlocal bayesian image denoising algorithm. *SIAM Journal on Imaging Sciences*, 6(3):1665–1688.

[Leclaire and Rabin, 2019] Leclaire, A. and Rabin, J. (2019). A fast multi-layer approximation to semi-discrete optimal transport. In *International Conference on Scale Space and Variational Methods in Computer Vision*, pages 341–353. Springer.

[Liu et al., 2016] Liu, G., Gousseau, Y., and Xia, G. (2016). Texture synthesis through convolutional neural networks and spectrum constraints. In *Int. Conf. on Pattern Recognition (ICPR)*, pages 3234–3239. IEEE.

[Mescheder et al., 2018] Mescheder, L., Geiger, A., and Nowozin, S. (2018). Which training methods for gans do actually converge? *arXiv preprint arXiv:1801.04406*.

[Portilla and Simoncelli, 2000] Portilla, J. and Simoncelli, E. (2000). A parametric texture model based on joint statistics of complex wavelet coefficients. *International Journal of Computer Vision*, 40(1):49–70.

[Santambrogio, 2015] Santambrogio, F. (2015). Optimal transport for applied mathematicians. *Progress in Nonlinear Differential Equations and their applications*, 87.

[Shaham et al., 2019] Shaham, T. R., Dekel, T., and Michaeli, T. (2019). Singan: Learning a generative model from a single natural image. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4570–4580.

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

[Ulyanov et al., 2016] Ulyanov, D., Lebedev, V., Vedaldi, A., and Lempitsky, V. (2016). Texture networks: feed-forward synthesis of textures and stylized images. In *Proc. of the Int. Conf. on Machine Learning*, volume 48, pages 1349–1357.

[Xia et al., 2014] Xia, G., Ferradans, S., Peyré, G., and Aujol, J. (2014). Synthesizing and Mixing Stationary Gaussian Texture Models. *SIAM Journal on Imaging Sciences*, 7(1):476–508.

[Yu et al., 2019] Yu, N., Barnes, C., Shechtman, E., Amirghodsi, S., and Lukác, M. (2019). Texture mixer: A network for controllable synthesis and interpolation of texture. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 12164–12173.

[Zhou et al., 2018] Zhou, Y., Zhu, Z., Bai, X., Lischinski, D., Cohen-Or, D., and Huang, H. (2018). Non-stationary texture synthesis by adversarial expansion. *arXiv preprint arXiv:1805.04487*.

[Zhu et al., 1998] Zhu, S., Wu, Y., and Mumford, D. (1998). Filters, random fields and maximum entropy (FRAME): Towards a unified theory for texture modeling. *International Journal of Computer Vision*, 27(2):107–126.

[Zoran and Weiss, 2011] Zoran, D. and Weiss, Y. (2011). From learning models of natural image patches to whole image restoration. In *2011 International Conference on Computer Vision*, pages 479–486. IEEE.