A Generative Model for Texture Synthesis based on Optimal Transport between Feature Distributions

Antoine Houdard, Arthur Leclaire, Nicolas Papadakis, Julien Rabin

A GENERATIVE MODEL FOR TEXTURE SYNTHESIS BASED ON OPTIMAL TRANSPORT BETWEEN FEATURE DISTRIBUTIONS*

Antoine Houdard
Université de Bordeaux
Bordeaux INP, CNRS, IMB, UMR 5251
33400 Talence, France
antoine.houdard@ubisoft.com†

Arthur Leclaire
Université de Bordeaux
Bordeaux INP, CNRS, IMB, UMR 5251
33400 Talence, France
arthur.leclaire@u-bordeaux.fr

Nicolas Papadakis
Université de Bordeaux
Bordeaux INP, CNRS, IMB, UMR 5251
33400 Talence, France
nicolas.papadakis@u-bordeaux.fr

Julien Rabin
Université de Normandie
UniCaen, ENSICAEN, CNRS, GREYC, UMR 6072
14000 Caen, France
julien.rabin@unicaen.fr

ABSTRACT

We propose GOTEX, a general framework for texture synthesis by optimization that constrains the statistical distribution of local features. While our model encompasses several existing texture models, we focus on the case where the comparison between feature distributions relies on optimal transport distances. We show that the semi-dual formulation of optimal transport allows to control the distribution of various possible features, even if these features live in a high-dimensional space. We then study the resulting minimax optimization problem, which corresponds to a Wasserstein generative model, for which the inner concave maximization problem can be solved with standard stochastic gradient methods. The alternate optimization algorithm is shown to be versatile in terms of applications, features and architecture; in particular it allows to produce high-quality synthesized textures with different sets of features. We analyze the results obtained by constraining the distribution of patches or the distribution of responses to a pre-learned VGG neural network. We show that the patch representation can retrieve the desired textural aspect in a more precise manner. We also provide a detailed comparison with state-of-the-art texture synthesis methods. The GOTEX model based on patch features is also adapted to texture inpainting and texture interpolation. Finally, we show how to use our framework to learn a feed-forward neural network that can synthesize on-the-fly new textures of arbitrary size in a very fast manner. Experimental results and comparisons with the mainstream methods from the literature illustrate the relevance of the generative models learned with GOTEX.

Keywords Optimal Transport · Generative model · Texture Synthesis

1 Introduction

A lot of attention has been recently drawn on the problem of designing deep generative models from an image database [17, 2, 25]. In contrast, synthesizing a texture from a single sample is a long-standing image processing problem for which many solutions have been proposed, as we will recall below. The main purpose of this work is to discuss whether the methodology developed for deep generative models can adapt to the case of learning from a single texture sample, depending on the choice of textural features that one wishes to preserve. We will restrict to the

*This study has been carried out with financial support from the French 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.

†Current affiliation: Ubisoft La Forge, 33000 Bordeaux

relatively simple case of stationary textures (i.e. with no large geometric deformations nor lighting changes) which already benefits from powerful tools for analysis and synthesis.

1.1 Features for texture synthesis

In the stationary setting, the common point of view adopted in parametric texture models is to represent the textural aspect through the statistics of local features extracted from the neighborhoods of all pixels. Parametric texture models thus encompass the Gaussian model [11] (based on mean and covariances of pixel values), the Heeger-Bergen model [20] (based on first-order distributions of responses to a filter bank) and the Portilla-Simoncelli approach [38] (based on second-order statistics computed on complex wavelet filter responses).

More recently, features extracted with a deep convolutional neural network have permitted to accurately solve difficult imaging problems, with tremendous success in image classification [29, 45] or texture synthesis [14] for example. Such *deep features* are nevertheless complex to understand and to interpret. This makes difficult the prediction and the tuning of the results provided by methods based on deep features. An illustration of this major caveat is that, among all existing representation learning techniques, the only pre-trained neural features that are used in practice for texture synthesis (e.g. in [24, 48]) are solely based on the VGG network trained on ImageNet [45], as proposed in the seminal work of [14]. As shown in our experiments of section 6, Adversarial-based techniques are not competitive when training on a single image. Additionally, those features require GPUs with large memory to be computed efficiently. A question that naturally arises is then: do we actually *need* deep features to encode a texture?

Deep features are computed from the image on *patches*, which are small regions of size $s \times s$ around each pixel, also called the local receptive field of the feature. Patches of pixels are the simplest local feature that can be considered in this setting. Such a patch representation has originally be proposed to design texture synthesis methods based on simple iterative copy/paste operations or nearest-neighbor assignments [10, 31]. The patch representation has also been widely exploited for other purposes. It is indeed at the core of efficient image restoration methods [4, 32, 22]. Recently, it has also been shown to be powerful in comparison to representation learning techniques [47].

Current limitations Patch-based approaches generally suffer from three main practical limitations. First, the patches are often processed independently and then combined to form a recomposed image [13, 33]. The overlap between patches leads to low frequency artifacts such as blurring. Second, the 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 [19, 26].

In the deep neural network community, deep feature representations have overtaken the patch representation in most of recent texture synthesis methods. Patches may indeed be considered to be less informative than deep features. Popular texture synthesis methods such as [14], which enforces the Gram matrices of deep features from the synthesized texture, do not provide meaningful results if deep features are replaced by patches. Nevertheless, the use of deep features leads to visual artifacts such as color inconsistencies or checkerboard patterns on the generated texture. Post-processing steps such as histogram equalization as in [14] or the application of median filter are necessary to provide relevant synthesis [9].

1.2 Optimal transport for texture synthesis

Should we be working with patches or deep features, one common difficulty is to design tools that allow to compare the distribution of patches or feature responses (which both live in a high-dimensional space and have a strongly non-Gaussian behavior).

In this work, we propose to compare these distributions with an optimal transport (OT) cost. Contrary to divergences (e.g. Kullback-Leibler), the OT distance is a relevant tool for comparing distributions that have disjoint supports. It is also adapted for matching both discrete and continuous distributions. As we now detail, the use of OT cost for texture synthesis has already proven to be fruitful in the literature.

For example, the authors of [46] suggest to rely on discrete Wasserstein distances in order to measure the proximity of distributions of extracted features (thus reinterpreting the Heeger-Bergen algorithm [20] as an alternate gradient descent on a composite Wasserstein cost). In [40] a sliced Wasserstein distance is used on distributions of local features (responses to a steerable pyramid) in order to compute texture barycenters. Notice also that such a sliced Wasserstein distance was used in [21] to compare distributions of deep features, in order to address texture synthesis. Wasserstein distances can also be used to compare patch distributions, either with a discrete formulation [19] or a semi-discrete one [13, 33]. In [49], the authors proposed to extract the means and covariances of the feature responses and then to rely on the Wasserstein distance between the corresponding Gaussian distributions. This method exploits the closed-form formula of the Wasserstein distance between Gaussian distributions, which can be efficiently computed in any dimension (as already used in [51] for dynamic textures). This can be seen as an extension of the model from [14] that compares feature distributions by exploiting the Frobenius norm between Gram matrices of features. However, the approach in [49] cannot handle non-Gaussian behavior of the feature responses. In the following, we will instead consider the general optimal transport case with no assumption on the feature distribution.

Current limitations of Wasserstein Generative models In parallel, the use of Wasserstein distances has helped to improve models based on adversarial training. As introduced in [17] for image synthesis from a database, a generative adversarial network (GAN) is inherently trained to fool another neural network that is simultaneously optimized to discriminate between real images and synthetic images. Such an adversarial training can be formulated as the minimization of a discrepancy between distributions, namely the Jensen-Shannon divergence in the original work [17] or the 1-Wasserstein distance in the paper [2] introducing Wasserstein generative adversarial networks (WGAN). Both these works rely on a dual formulation of the chosen discrepancy and suggest to parameterize the corresponding dual variable by a neural network. Thanks to the properties of the Wasserstein distance, WGAN has offered an elegant solution to mode collapse issues related to GANs. Alternative techniques to train generative neural networks also took profit from using the OT framework. For instance, the Sliced-Wasserstein distance has been considered in the latent space of auto-encoders in [28].

Building on these ideas, adversarial models have been proposed for texture synthesis from a single example [3] or for feed-forward synthesis of general images [44]. Although achieving convincing performance on synthesis problems, the main limitation of GAN or WGAN is that they require to optimize a discriminative network, which makes the process unstable and requires a large number of additional parameters [17, 36]. In the case of WGAN, the discriminative network is theoretically related to the dual formulation of the Wasserstein-1 distance, and thus should represent a 1-Lipschitz mapping. Different strategies have thus been proposed to enforce such a constraint (*e.g.* weight clipping or gradient penalty [18]), thus only approximating the true Wasserstein-1 distance. Another strategy adopted by the authors of [43] is to rely on regularized optimal transport, which leads to an unconstrained dual problem. However, this new dual problem involves two dual variables that must be parameterized by two different neural networks, which leads to a non-convex problem with twice more variables. In contrast, in [6], the optimization 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. This has the benefit of keeping a convex formulation for the OT dual problem which stabilizes training [23], while not being specific to the L^1 cost. In the following we will adopt the same approach than [6, 23] to approximate the solution of OT, but we will include it in a more general framework able to learn a generative network. In addition, the OT distances will be used not to compare distributions of generated images but rather to constrain the feature distribution of synthesized images.

Table 1: Technical comparison of previous work based on the following criteria: Fast synthesis based on a *Feed-Forward* architecture trained offline; Optimal Transport (*OT*) based objective function; *Patch*-based representation; *Deep* features for optimization, where * indicates that such features are simultaneously learned during training using an adversarial loss, rather than defined from a pre-trained neural network.

Method	Feed-Forward	OT	Patch	Deep Features
Gram-VGG [14]	✗	✗	✗	✓
SINGAN [44]	✓	✗	✓	✓*
PSGAN [3]	✓	✗	✓	✓*
Texture Networks [48]	✓	✗	✗	✓
TexOptim [31]	✗	✗	✓	✗
OPA [19]	✗	✓	✓	✗
TexTo [33]	✓	✓	✓	✗
GOTEX	✓	✓	✓	✓

1.3 Contributions and outline

As summed up in Table 1, state-of-the-art texture synthesis methods are either based on patches or deep features. In this work, we propose a unified framework in order to address the limitations associated with both kinds of features with a formulation inspired by generative networks.

For this purpose, we introduce a Generative model based on Optimal transport for synthesizing TEXtures (GOTEX) while prescribing their feature distributions. The idea is to define a texture formation model as the push-forward of a latent distribution ζ by a measurable function g_θ and to consider its underlying feature distribution. Then the parameter θ of the model is optimized to enforce the feature distribution at different scales to be close to the one of the exemplar

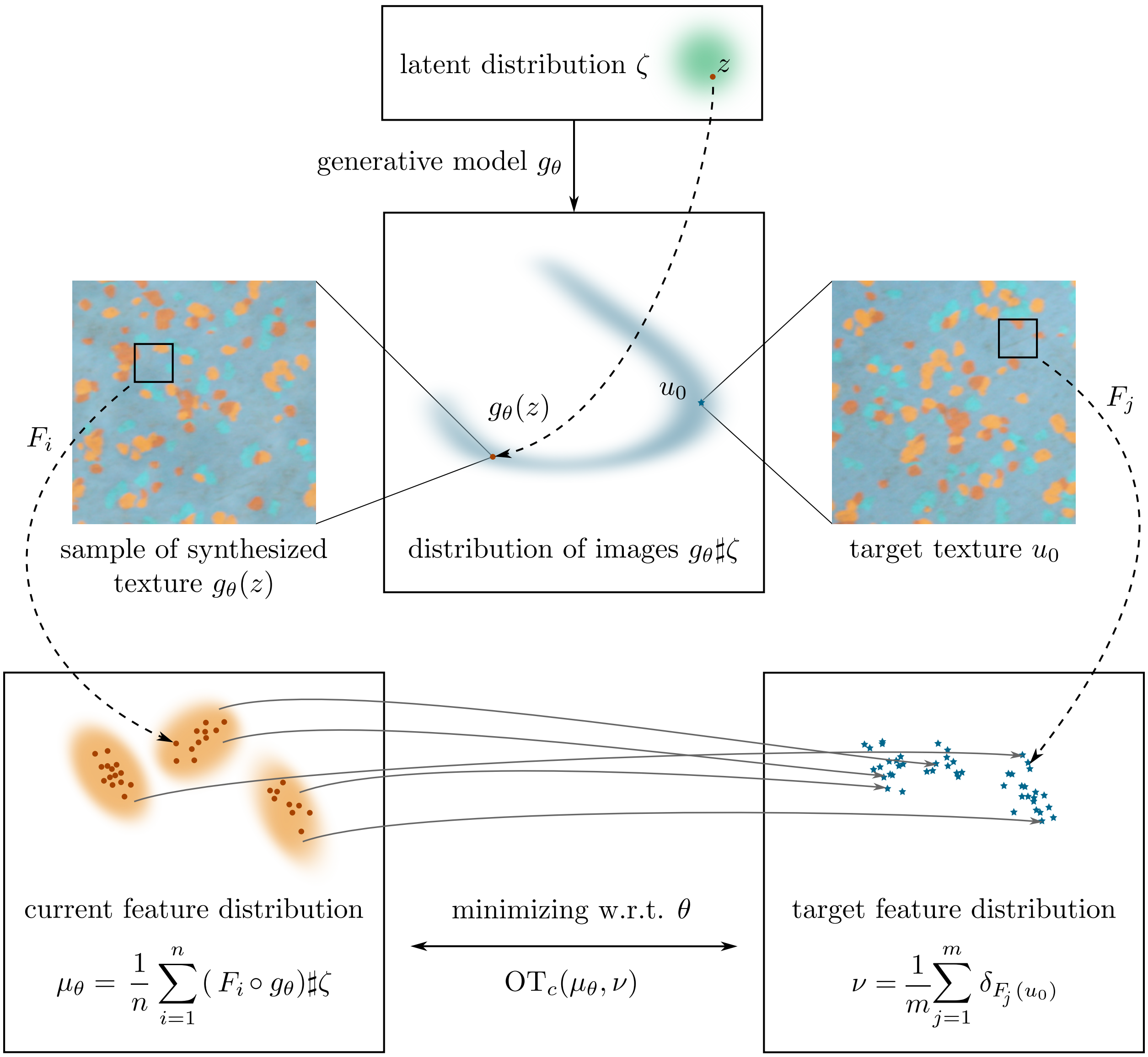


Figure 1: Summary of the proposed GOTEX framework. A texture formation model is encoded with a generative model g_θ and the distribution of texture images is represented through its feature distribution μ_θ. The objective is then to minimize the optimal transport cost $\mathrm{OT}_c(\mu_\theta, \nu)$ between the current feature distributions μ_θ and the (discrete) feature distribution ν of the example target texture u_0. This framework also encompasses the case where the optimization is done on the image pixels by taking the latent distribution as a Dirac (see section 5 for details).

image, in the sense of optimal transport. The proposed pipeline is illustrated in Fig. 1. The organization of the paper and the description of its main contributions are listed below.

In section 2 we introduce the GOTEX framework that enforces the feature distribution of generated textures and treats in the same way texture synthesis by pixel-wise optimization (section 2.1) and by learning a generative model (section 2.2). Both problems will respectively rely on the discrete and semi-discrete formulations of the optimal transport cost. In section 3 we state a differentiation result that gives a formula for the gradient of the optimal transport cost between feature distributions with respect to the parameter θ (see theorem 3).

In section 4 we present the GOTEX algorithm and detail the versatility of the framework, which can combine different distributions of features in the texture synthesis model. Our approach namely encompasses multi-scale procedures on patches or VGG-19 features. Contrary to previous methods relying on approximations of OT that are detailed in section 5.2, our framework allows for a more accurate optimization. The proposed model also offers a theoretical-sound framework to compute barycenters of texture models, thus providing a relevant way to synthesize interpolated textures.

In section 5, we focus on the image optimization setting where an optimization problem is solved for each new synthesis. This involves discrete optimal transport problems that can be efficiently solved with dedicated nearest-neighbor search libraries. We then propose an extensive analysis of the model including the comparison of different features and losses and the comparison of different numerical methods approximating the optimal transport. In addition to texture synthesis, we also generalize the model in order to address texture inpainting and interpolation.

In section 6, we finally demonstrate that the GOTEX framework is well suited to the training of a deep generative feed-forward convolutional neural network, as proposed in [48, 3, 44] for texture generation.

2 GOTEX: a Generative model based on Optimal transport for synthesizing textures

In this section, we present a generic framework formulating texture synthesis as the minimization of a loss function that reflects the proximity of a set of features of the synthesized image(s) to the ones of the example. As we will see, different choices of the loss functions can model the statistical behavior of the features in a parametric or non-parametric way. As we will see, when considering the feature distributions, the loss function can be expressed using optimal transport distances.

2.1 Texture synthesis by minimizing a distance between feature sets

We first consider the synthesis of a single image $u \in \mathbf{R}^n$ with n pixels. For each pixel i of the image, we consider a measurable map $F_i : \mathbf{R}^n \to \mathbf{R}^d$ that extracts a local feature of dimension d computed from the neighborhood of pixel i. For example, $F_i(u)$ may be a square patch of dimension $d = 3 \times s \times s$, or a collection of d neural responses computed at pixel i. The features of the image u will be gathered in a vector $F(u) = (F_i(u))_{1 \leq i \leq n} \in \mathbf{R}^{dn}$.

We also consider a cost function $\Lambda : \mathbf{R}^{dn} \times \mathbf{R}^{dn} \to \mathbf{R}_+$ that is chosen to assess the proximity between feature maps. Then we can define a loss function between two textures $u \in \mathbf{R}^n$ and $u_0 \in \mathbf{R}^n$ as

$$\mathcal{L}(u, u_0) = \Lambda\left(F(u), F(u_0)\right). \tag{1}$$

A new sample u of a given example texture u_0 may be obtained by minimizing (1) with respect to u. If we assume that Λ is differentiable and that for all i, F_i is differentiable with respect to the image $u \in \mathbf{R}^n$, the loss function (1) may be minimized by performing a gradient descent with respect to the pixels of the image u. Due to the potential non linearity of the feature extraction operators F_i and/or of the loss function Λ such a problem is typically non-convex. However, starting from a random initialization, gradient descent schemes can converge to local minima that will correspond to plausible syntheses of the exemplar texture.

Remark 1. *The seminal work of [14] is included in this framework. More precisely, it corresponds to take as features $F_i(u)$ the normalized outputs of a pre-trained VGG network [45] at different layers l, to define the Gram matrix of the features*

$$Gr(F(u)) = \frac{1}{n} \sum_{i=1}^{n} F_i(u) F_i^\top(u) \in \mathbf{R}^{d \times d}, \tag{2}$$

and to compare Gram matrices with the squared Frobenius norm $\|.\|_F^2$. A synthesized image is then obtained in [14] by minimizing with respect to u the quantity

$$\mathcal{L}_{Gram}(u, u_0) = \sum_l \|Gr(VGG_l(u)) - Gr(VGG_l(u_0))\|_F^2. \tag{3}$$

2.2 Generative models

The previous model (1) requires to perform an optimization each time a new texture u is synthesized. Hence, the authors of [48] have later proposed to first train a generative model and then realize new syntheses on-the-fly. The optimization is realized on the parameters of a feed-forward network rather than on the image pixels.

To that end, we assume that different samples of a given texture are actually samples of a probability distribution. This distribution is defined as the push-forward of a given random distribution ζ defined on $\mathcal{Z}$ (e.g. a uniform distribution ζ on a latent space $\mathcal{Z} = [0, 1]^M$ of dimension M), with a generator g to estimate. Let us consider a measurable generative function $g : \Theta \times \mathcal{Z} \to \mathbf{R}^n$ where Θ is a set of parameters. For a given parameter $\theta \in \Theta$ we write $g_\theta = g(\theta, \cdot)$ and we consider the output texture distribution as the push-forward $g_\theta \sharp \zeta$, which is given by $g_\theta \sharp \zeta(B) = \zeta(g_\theta^{-1}(B))$ for any Borel set B. A relevant generative model may thus be learned by minimizing with respect to the parameters θ the following objective function:

$$\mathcal{L}_{gen}(\theta, u_0) = \mathbf{E}_{Z \sim \zeta} \left[\Lambda \left(F(g_\theta(Z)), F(u_0) \right) \right]. \tag{4}$$

When considering such generative models, we face a *semi-discrete problem* as the data at hand u_0 is discrete while the generated distribution $g_\theta \sharp \zeta$ is expected to be absolutely continuous. In the following, we present a general framework that includes both models (1) and (4) based on the probabilistic distributions of features.

2.3 Probabilistic representation of the features

We now define the generic texture formation model that permits to encompass the optimization on the image pixels (1) and the optimization on the weights of a generative model (4). To that end, we propose to consider the probability distribution of the features F_i taken on the texture distribution $g_\theta \sharp \zeta$. Assuming that all F_i are measurable, we have for each i a local feature distribution given by

$$\mu_\theta^i = F_i \sharp \left(g_\theta \sharp \zeta \right) = \left(F_i \circ g_\theta \right) \sharp \zeta. \tag{5}$$

Then the whole feature distribution of the generative model writes

$$\mu_\theta = \frac{1}{n} \sum_{i=1}^{n} \mu_\theta^i = \frac{1}{n} \sum_{i=1}^{n} \left(F_i \circ g_\theta \right) \sharp \zeta. \tag{6}$$

Remark 2. *This formulation includes the case of the single image synthesis corresponding to the minimization with respect to image pixels (1). If θ denotes the image to optimize, then taking $\zeta = \delta_0$ and $g(\theta, z) = \theta - z$ gives $g_\theta \sharp \zeta = \delta_\theta$. Since we have $F_i \sharp \delta_\theta = \delta_{F_i(\theta)}$, the underlying feature distribution of the image θ writes as the discrete probability distribution*

$$\mu_\theta = \frac{1}{n} \sum_{i=1}^{n} \delta_{F_i(\theta)}. \tag{7}$$

To perform texture synthesis, we aim at minimizing with respect to θ a distance function between μ_θ and the distribution ν of features extracted from a target image u_0. In this setting, a natural tool for comparing probability distributions appears to be optimal transport. We now describe our formulation based on an optimal transport distance.

2.4 Optimal Transport cost for comparing feature distributions

Given an example texture u_0, we follow the previous section and denote as ν its feature distribution:

$$\nu = \frac{1}{m} \sum_{j=1}^{m} \delta_{F_j(u_0)}. \tag{8}$$

Our objective is to constrain the feature distribution μ_θ of the synthesized textures defined in (6) in order to match the target distribution ν. To do so, in the GOTEX framework, the loss function forces the patch distribution μ_θ of the synthesized textures to be close to the empirical example patch distribution ν for the optimal transport cost

$$\mathcal{L}_{\text{GOTEX}}(\theta, u_0) = \mathrm{OT}_c(\mu_\theta, \nu) = \inf_{\pi \in \Pi(\mu_\theta, \nu)} \int c(x, y) d\pi(x, y), \tag{9}$$

where $c : \mathbf{R}^d \times \mathbf{R}^d \to \mathbf{R}$ is a Lipschitz cost function (between features) and $\Pi(\mu_\theta, \nu)$ is the set of probability distributions on $\mathbf{R}^d \times \mathbf{R}^d$ having marginals μ_θ and ν. When using $c(x, y) = \|x - y\|^2$, as done for experiments in this paper, OT_c corresponds to the square of the Wasserstein-2 distance.

Minimizing the optimal transport cost in Equation (9) with respect to one of its argument is a difficult task in general. The situation is even harder in our case as we wish to differentiate (9) with respect to θ and we have to deal with a nonlinear mapping $\theta \mapsto g_\theta$. The dual formulation of OT will allow us to separate the problems of approximating the OT distance and minimizing w.r.t θ as in [2]. In this work, we will also exploit the discrete nature of the target distribution ν to rely on flexible algorithms for semi-discrete optimal transport. Before going into such technical details in the next section, we present below in (12) the final problem we will optimize.

Texture synthesis is obtained with the minimization of the optimal transport cost (9) with respect to μ_θ, its first argument. Hence we consider the semi-dual formulation of the optimal transport cost.

Remark 1 (Semi-dual formulation [42])**.** *If $\mathcal{X}$ and $\mathcal{Y}$ are compact and the cost c is continuous, then*

$$\mathrm{OT}_c(\mu,\nu) = \max_{\psi\in\mathcal{C}(\mathcal{Y})} \int \psi^c(x)d\mu(x) + \int \psi(y)d\nu(y), \tag{10}$$

where $\psi : \mathcal{Y} \to \mathbf{R}$ and its c-transform is defined by

$$\psi^c(x) = \min_{y\in\mathcal{Y}} \left[c(x,y) - \psi(y)\right]. \tag{11}$$

Semi-dual here refers to the fact that the dual problem is formulated with only one dual variable while the other dual variable is optimized through the c-transform. Combining (9) and (10), we get that estimating the variable θ (a new texture or the parameters of a generator) amounts to solving the following problem

$$\min_\theta \mathrm{OT}_c(\mu_\theta,\nu) = \min_\theta \max_{\psi\in\mathcal{C}(\mathcal{Y})} \mathcal{J}(\theta,\psi) := \int \psi^c(x)d\mu_\theta(x) + \int \psi(y)d\nu(y). \tag{12}$$

For a fixed θ, the function $\psi \mapsto \mathcal{J}(\theta,\psi)$ is concave and an optimal ψ^* can be approximated with an averaged stochastic gradient ascent as proposed in [16].

When the variable θ is an image, we propose in Section 5 to perform a gradient descent, whose outcome is illustrated in Fig. 1. A stochastic gradient-based algorithm is finally proposed in Section 6 to learn a generative model using a convolutional neural network. Both approaches exploit the property, demonstrated in Section 3, that, upon existence, the gradient of the optimal transport $\nabla_\theta \mathrm{OT}_c(\mu_\theta,\nu)$ coincides with the gradient $\nabla_\theta \mathcal{J}(\theta,\psi^*)$ of the function $\mathcal{J}$ at an optimal value ψ^*.

3 Gradients for the Semi-Dual Optimal Transport Cost

In this section we study the gradients with respect to θ and ψ of the optimal transport cost $\mathcal{J}(\theta,\psi)$ introduced in (12). In the whole section, we will assume that all feature extraction operators $F_i : \mathbf{R}^m \to \mathbf{R}^d$ are differentiable.

3.1 The semi-discrete formulation

When dealing with texture synthesis from a single image example, the target measure ν is an empirical feature distribution with finite support

$$\nu = \sum_{j=1}^m \delta_{F_j(u_0)},$$

composed of m features $y_j = F_j(u_0)$. In this case, the semi-dual formulation of optimal transport (12) simplifies to

$$\mathrm{OT}_c(\mu_\theta,\nu) = \max_{\psi\in\mathbf{R}^m} \mathcal{J}(\theta,\psi) = \int \psi^c(x)d\mu_\theta(x) + \frac{1}{m}\sum_{j=1}^m \psi_j, \tag{13}$$

where $\psi_j = \psi(y_j)$ and where the c-transform of ψ writes $\psi^c(x) = \min_j \left[c(x, F_j(u_0)) - \psi_j\right]$. The main interest of this formulation is that it involves only a finite-dimensional vector $\psi \in \mathbf{R}^m$, which can be numerically optimized. Notice also that the computation of the c-transform $\psi^c(x)$ boils down to a biased nearest-neighbor assignment in the feature space. Combining the texture formation model μ_θ introduced in (6) and the functional of interest $\mathcal{J}$ defined in (13), the final problem to optimize reads

$$\min_\theta \mathrm{OT}_c(\mu_\theta,\nu) = \min_\theta \max_{\psi\in\mathbf{R}^m} \mathcal{J}(\theta,\psi) = \mathbf{E}_{Z\sim\zeta}\left[\frac{1}{n}\sum_{i=1}^n \psi^c(F_i \circ g_\theta(Z)) + \frac{1}{m}\sum_{j=1}^m \psi_j\right], \tag{14}$$

where θ can be a single image (Section 5) or the parameters of a generative network with input noise Z (Section 6). The end of this section is focused on the computation of the gradients of this quantity with respect to θ and ψ.

3.2 Gradient with respect to θ

This section discusses the computation of the gradient with respect to the parameter θ of the optimal transport cost $\mathrm{OT}_c(\mu_\theta,\nu)$. We provide a sufficient condition (related to the regularity of the generator g_θ and to the feature distribution μ_θ) that ensures existence of $\nabla_\theta\mathcal{J}(\theta,\psi)$. Besides, under the same condition, we show that

$\nabla_\theta \mathrm{OT}_c(\mu_\theta, \nu) = \nabla_\theta \mathcal{J}(\theta, \psi^*)$ with $\psi^* \in \arg\max_\psi \mathcal{J}(\theta, \psi)$ as soon as both terms are well defined.

As can be observed in expression (14), the computation of the gradient of $\mathcal{J}$ with respect to θ only involves the differentiation of $\psi^c(F_i \circ g_\theta(Z))$. In order to examine the regularity of ψ^c, we introduce the open Laguerre cells

$$L_i(\psi) = \{x \mid \forall k \neq i,\ c(x, F_i(u_0)) - \psi_i < c(x, F_k(u_0)) - \psi_k\}\,. \tag{15}$$

A simple but crucial remark directly follows from the definition of the c-transform: in the Laguerre cell $L_i(\psi)$, the c-transform expresses as

$$\forall x \in L_i(\psi), \quad \psi^c(x) = c(x, F_i(u_0)) - \psi_j. \tag{16}$$

Therefore, ψ^c inherits the regularity of c in the Laguerre cells, and thus, when c is smooth, differentiability problems can only appear at the boundaries of the Laguerre cells. In order to avoid such singularities, we formulate the following hypothesis that constrains the feature distribution of the texture model.

Hypothesis 1. *g satisfies Hypothesis 1 at (θ, ψ) if $\zeta\left((F_i \circ g_\theta)^{-1}\{\cup_j L_j(\psi)\}\right) = 1$ for any position i, that is, for a given variable θ, all the generated local features are almost surely within the Laguerre cells defined by ψ.*

For example, if $c(x, y) = \|x - y\|_p^p$ with $p > 1$, then ψ^c is smooth on $\cup_j L_j(\psi)$, whose complement is negligible for the Lebesgue measure. Indeed, in this case its complement is given by the union of the sets

$$B_{jk}(\psi) = \left\{x \mid \|x - F_j(u_0))\|_p^p - \psi_j = \|x - F_k(u_0)\|_p^p - \psi_k\right\} \quad (1 \leq j, k \leq m) \tag{17}$$

which has zero Lebesgue measure, because each B_{jk} is contained in a sub-manifold of dimension lower than p. Therefore, Hypothesis 1 is satisfied if, for any i, $(F_i \circ g_\theta)\sharp\zeta$ is absolutely continuous with respect to the Lebesgue measure.

We also introduce a regularity hypothesis for the generative model g_θ which will allow us to differentiate under the expectation.

Hypothesis 2. *There exists $K : \Theta \times \mathcal{Z} \to \mathbf{R}_+$ such that for all θ, there exists a neighborhood V of θ such that $\forall \theta' \in V$ and for ζ-almost every z,*

$$\|g(\theta, z) - g(\theta', z)\| \leq K(\theta, z)\|\theta - \theta'\| \tag{18}$$

with K verifying for all θ, $\mathbf{E}_{Z\sim\zeta}\left[K(\theta, Z)\right] < \infty$.

We can now express the gradient of $\mathcal{J}$ with respect to the parameter θ.

Remark 2. *Assume c to be $\mathscr{C}^1$ and assume that the features F_i are all differentiable and Lipschitz. Let g satisfy Hypothesis 2. Let θ_0 be a point where $\theta \to g_\theta(z)$ is differentiable $\zeta(dz)$-almost surely and let g satisfy Hypothesis 1 at (θ_0, ψ). Then $\theta \mapsto \mathcal{J}(\theta, \psi)$ is differentiable at θ_0 and*

$$\nabla_\theta \mathcal{J}(\theta_0, \psi) = \frac{1}{n}\sum_{i=1}^{n} \mathbf{E}_{Z\sim\zeta}\left[\partial_\theta(F_i \circ g)(\theta_0, Z)^T \nabla\psi^c(F_i \circ g(\theta_0, Z))\right] \tag{19}$$

with $\nabla\psi^c(F_i \circ g(\theta_0, z)) = \nabla_x c(F_i \circ g(\theta_0, z), F_{\sigma(i)}u_0)$

where $\sigma(i)$ is the unique index such that $F_i \circ g(\theta_0, z) \in L_{\sigma(i)}(\psi)$ (which exists $\zeta(dz)$-almost surely).

Proof. From expression (14), we see that the proof consists in differentiating the function

$$H_i(\theta, \psi) = \mathbf{E}\left[h_i(\theta, \psi, Z)\right], \quad \text{with} \quad h_i(\theta, \psi, Z) = \psi^c(F_i \circ g_\theta(Z)). \tag{20}$$

Thanks to Hypothesis 1, for ζ-almost all z, there exists an index j such that $F_i \circ g_\theta(z) \in L_j(\psi)$ and thus $L_j(\psi)$ is an open neighborhood of $F_i \circ g_\theta(Z)$ where ψ^c is differentiable. Using the chain rule, we get that for ζ-almost all z, $\theta \mapsto h_i(\theta, \psi, z)$ is differentiable at θ_0 and

$$\nabla_\theta h_i(\theta, \psi, z) = \partial_\theta(F_i \circ g)(\theta_0, Z)^T \nabla\psi^c(F_i \circ g(\theta_0, Z)). \tag{21}$$

In order to differentiate under the expectation in (20), we have to get an integrable bound on the finite differences of h_i. For that, let us denote by κ_c the Lipschitz constant of c and κ_i the Lipschitz constant of F_i. Let us recall from [42] that ψ^c is also κ_c-Lipschitz. Therefore, from Hypothesis 2, we get a neighborhood V of θ_0 such that for any $\theta \in V$ and for ζ-almost all z,

$$|h_i(\theta, \psi, z)) - h_i(\theta_0, \psi, z)| \leq \kappa_c\kappa_i\|g(\theta, z) - g(\theta_0, z)\| \leq \kappa_c\kappa_i K(\theta, z)\|\theta - \theta_0\|, \tag{22}$$

with $\mathbf{E}[K(\theta, Z)] < \infty$. This bound allows us to differentiate under the expectation and to get the expression

$$\nabla_\theta H_i(\theta_0, \psi) = \mathbf{E}[\partial_\theta(F_i \circ g)(\theta_0, Z)^T \nabla\psi^c(F_i \circ g(\theta, Z))]. \tag{23}$$

Gathering the terms for all i leads to the desired result. □ □

Finally, upon existence, we can relate the gradient of $\mathcal{J}$ to the gradient of the optimal transport.

Remark 3. *Let θ_0 such that $\theta \mapsto \mathrm{OT}_c(\mu_\theta, \nu)$ and $\theta \mapsto \mathcal{J}(\theta, \psi^*)$ are differentiable at θ_0 with $\psi^* \in \arg\max_\psi \mathcal{J}(\theta_0, \psi)$ then*

$$\nabla_\theta \mathrm{OT}_c(\mu_{\theta_0}, \nu) = \nabla_\theta \mathcal{J}(\theta_0, \psi^*) \tag{24}$$

Proof. Let us fix $\psi^* \in \arg\max_\psi \mathcal{J}(\theta_0, \psi)$. The function $H(\theta) = \mathcal{J}(\theta_0, \psi^*) - \mathrm{OT}_c(\mu_\theta, \nu)$ is differentiable at θ_0 and maximal at θ_0. Therefore we get $\nabla_\theta H(\theta_0) = 0$. □ □

The gradient expression found here will be later used to minimize $\theta \to \mathrm{OT}_c(\mu_\theta, \nu)$. A stochastic gradient-based algorithm will be used to reach a local minimum of this optimal transport cost and learn the texture model $g_\theta(Z)$. Notice that evaluating $\nabla_\theta \mathcal{J}(\theta_0, \psi^*)$ in (24) requires the knowledge of an optimal potential $\psi^*(\theta_0) \in \arg\max_\psi \mathcal{J}(\theta_0, \psi)$. The next section discusses how to approximate such an optimal potential.

3.3 Super-gradient with respect to ψ

The computation of the exact transport cost is a challenging task and it has been widely studied in the literature. Recently, a stochastic method for approximating the optimal dual potential for the semi-discrete case has been studied [16]. We propose to use this approach to approximate the optimal potential ψ^* with a stochastic gradient ascent scheme. Hereafter we recall with proofs some known facts about the concavity and the super-gradients of $\mathcal{J}$, which will be used in the stochastic optimization algorithm.

Let θ be a fixed parameter, in order to improve the readability we set $G_i = F_i \circ g_\theta$ and remove all the θ dependencies. The maximization problem we aim at solving writes

$$\max_{\psi \in \mathbf{R}^m} \mathcal{J}(\psi) = \mathbf{E}_{Z\sim\zeta}\left[J(\psi, Z)\right], \text{ where } J(\psi, z) = \frac{1}{n}\sum_{i=1}^n \psi^c(G_i(z)) + \frac{1}{m}\sum_{j=1}^m \psi_j. \tag{25}$$

We first recall the following result from the optimal transport theory.

Remark 4. *(i) For any $z \in \mathcal{Z}$, the function $\psi \to J(\psi, z)$ is concave on $\mathbf{R}^m$.*

(ii) The function $\psi \to \mathcal{J}(\psi)$ is concave on $\mathbf{R}^m$.

Proof. Let $\psi^1 \in \mathbf{R}^m$, $\psi^2 \in \mathbf{R}^m$ and $t \in [0,1]$ and fix $z \in \mathcal{Z}$. Recalling the c-transform definition $\psi^c(x) = \min_j \left[c(x, F_j(u_0)) - \psi_j\right]$, we have

$$\begin{aligned}
(t\psi^1 + (1-t)\psi^2, z) =& \frac{1}{n}\sum_{i=1}^n \min_j \left[c(G_i(z), F_j(u_0)) - t\psi_j^1 - (1-t)\psi_j^2\right] \\
&+ \frac{1}{m}\sum_{j=1}^m t\psi_j^1 + (1-t)\psi_j^2 && (26)\\
=& \frac{1}{n}\sum_{i=1}^n \left[c(G_i(z), F_{j^*(i)}(u_0)) - t\psi_{j^*(i)}^1 - (1-t)\psi_{j^*(i)}^2\right] \\
&+ \frac{1}{m}\sum_{j=1}^m t\psi_j^1 + (1-t)\psi_j^2, && (27)
\end{aligned}$$

where $j^*(i) \in \arg\min_j \left[c(G_i(z), F_j(u_0)) - t\psi_j^1 - (1-t)\psi_j^2\right]$. Splitting $c(\cdot,\cdot) = tc(\cdot,\cdot) + (1-t)c(\cdot,\cdot)$ and using the property of the minimum function, we get that

$$\begin{aligned}
J(t\psi^1 + (1-t)\psi^2, z) \geq& \frac{1}{n}\sum_{i=1}^n t \min_j \left[c(G_i(z), F_j(u_0)) - \psi_j^1\right] && (28)\\
&+ \frac{1}{n}\sum_{i=1}^n (1-t) \min_j \left[c(G_i(z), F_j(u_0)) - \psi_j^2\right] && (29)\\
&+ \frac{1}{m}\sum_{j=1}^m t\psi_j^1 + (1-t)\psi_j^2 && (30)\\
&= tJ(\psi^1, z) + (1-t)J(\psi^2, z), && (31)
\end{aligned}$$

which proves the first point. The second point follows by taking the expectation of both sides. □ □

We can now state the following result that gives a super-gradient for $\mathcal{J}$.

Remark 5. *Let us denote by $(e_j)_{1\leq j\leq m}$ the canonical basis of $\mathbf{R}^m$. Let $z \in \mathcal{Z}$, and $\psi \in \mathbf{R}^m$. Then a super-gradient of $J(\cdot, z)$ at point ψ is given by*

$$D(\psi, z) = \frac{1}{m}\mathbf{1}_m - \frac{1}{n}\sum_{i=1}^{n} \mathbf{e}_{j^*(G_i(z),\psi)}, \tag{32}$$

where

$$j^*(G_i(z), \psi) \in \arg\min_j \left(c(G_i(z), F_j(v)) - \psi_j\right). \tag{33}$$

It follows that

$$\mathcal{D}(\psi) = \mathbf{E}_{Z\sim\zeta}\left[D(\psi, Z)\right], \tag{34}$$

is a super-gradient of $\mathcal{J}$ at point ψ.

Proof. Take $z \in \mathcal{Z}$ and $\psi_0 \in \mathbf{R}^m$. In order to demonstrate that $D(\psi, z)$ is a super-gradient, we need to show that

$$\forall \psi' \in \mathbf{R}^m, \quad J(\psi', z) \leq J(\psi, z) + \langle D(\psi, z), \psi' - \psi\rangle. \tag{35}$$

We have

$$J(\psi, z) - J(\psi', z) + \langle D(\psi, z), \psi' - \psi\rangle = \frac{1}{n}\sum_{i=1}^{n}\min_j \left[c(G_i(z), F_j(u_0)) - \psi_j\right] \tag{36}$$

$$- \frac{1}{n}\sum_{i=1}^{n}\min_j \left[c(G_i(z), F_j(u_0)) - \psi'_j\right] \tag{37}$$

$$- \frac{1}{n}\sum_{i=1}^{n}\left(\psi'_{j^*(G_i(z),\psi)} - \psi_{j^*(G_i(z),\psi)}\right). \tag{38}$$

Then, $j^*(G_i(z), \psi)$ satisfies the min in (36) and the min in the second term (37) is by definition smaller than the value taken at $j^*(G_i(z), \psi)$. Therefore all terms compensate and we have

$$J(\psi, z) - J(\psi', z) + \langle D(\psi, z), \psi' - \psi\rangle \geq 0. \tag{39}$$

□ □

In order to approximate an optimal potential, we will rely on an averaged stochastic super-gradient ascent as proposed in [16]. More precisely, at each step k we sample $z \sim \zeta$ and we update ψ^k as follows

$$\psi^k = \psi^{k-1} + \frac{1}{\sqrt{k}}D(\psi^{k-1}, z) \tag{40}$$

The final estimate for ψ^* is obtained by averaging the estimates from a given point k_0 (set to 1 in experiments)

$$\hat{\psi}^k = \frac{1}{k - k_0 + 1}\sum_{\ell=k_0}^{k}\psi^\ell. \tag{41}$$

Finally, the combination of Theorem 3 and Theorem 5 provides us a way to approximate the gradient of the optimal transport cost $\mathrm{OT}_c(\mu_\theta, \nu)$ with respect to θ. Then, we propose to minimize this optimal transport cost by performing a stochastic gradient descent algorithm. In practice we will use the Adam algorithm [27]. We detail the proposed algorithm and its application to texture synthesis in the next section.

4 Combining Several Feature Distributions

In the previous section, we defined the problem and stated theoretical results for a single collection of local features. In practice, texture synthesis requires to simultaneously enforce different collections of features. For instance one can use local features at different scales in order to model image patterns of various sizes [31, 13, 44]. We now present a general framework able to combine several sets of features.

We therefore consider a set of N_F different features given by $F^l = (F^l_i)_{1\leq i\leq n_l}$, $l = 1, \dots, N_F$, each one of size n_l, and we propose to minimize the quantity

$$\mathcal{L}_{\text{GOTEX}}(\theta) = \sum_{l=1}^{N_F} \mathrm{OT}_c(\mu^l_\theta, \nu^l) = \sum_{l=1}^{N_F} \max_{\psi^l\in\mathbb{R}^{m_l}} \mathcal{J}^l(\theta, \psi^l), \tag{42}$$

where μ^l_θ (resp. ν^l) is the distribution relative to the feature l of the synthesis (resp. of the example u_0 that contains m_l features), i.e. $\mu^l_\theta = \frac{1}{n_l}\sum_{i=1}^{n_l}\delta_{F^l_i(\theta)}$, and where

$$\mathcal{J}^l(\theta, \psi^l) = \mathbf{E}_{Z\sim\zeta}\left[\frac{1}{n_l}\sum_{i=1}^{n_l}\psi^{l,c}(F^l_i\circ g_\theta(Z)) + \frac{1}{m_l}\sum_{j=1}^{m_l}\psi^l_j\right],$$

with the c-transform defined as $\psi^{l,c}(x) = \min_{1\leq j\leq m_l}\left[c(x, F^l_j(u_0)) - \psi^l_j\right]$.

Upon existence of all terms, the gradient of the quantity (42) then reads

$$\nabla_\theta\mathcal{L}(\theta) = \sum_{l=1}^{N_F}\nabla_\theta\mathcal{J}^l(\theta, \psi^{l,*}), \tag{43}$$

where $\psi^{l,*}$ is an optimal Kantorovich potential for $\mathrm{OT}_c(\mu^l_\theta, \nu^l)$. The potential $\psi^{l,*}$ can be approximated with a super-gradient ascent (using the super-gradient of $\mathcal{J}^l$ obtained in Theorem 5). Therefore, in order to minimize the loss defined in (42), we propose to repeat the two following steps:

1. for each l compute $\hat{\psi}^l$ that approximates $\psi^{l,*}$ with an averaged stochastic super-gradient ascent;
2. perform an optimization step with respect to θ using (43) with $\hat{\psi}^l$. In the experiments we use the L-BFGS [35] for image-based optimization in section 5 and the Adam algorithm [27] for neural-network training in section 6.

The multi-feature process is summarized in Algorithm 1. The notation $D(\psi, z)$ was defined in (32) but is changed here in $D(\theta, \psi, z)$ to recall the dependency on the current θ. We now study different choices for the features (F^l) for texture synthesis.

Algorithm 1 Texture Synthesis with Prescription of Several Feature Distributions

Input: target image u_0, initial parameter θ_0, learning rates η_θ and η_ψ, number of iterations N_u and N_ψ, number of features N_F, optimizer *Optim* (s.t. Adam or L-BFGS)
Output: learned parameter θ^*
$\theta \leftarrow \theta_0$ and $\psi^{l,0} = 0$ for $l = 1 \dots N_F$
for $k = 1$ **to** N_θ **do**
  for $l = 1$ **to** N_F **do**
    $\tilde{\psi}^{l,0} \leftarrow \psi^{l,k-1}$
    for $\ell = 1$ **to** N_ψ **do**
      Draw a sample $z \sim \zeta$.
      $\tilde{\psi}^{l,\ell} = \tilde{\psi}^{l,\ell-1} + \eta_\psi D(\theta^{k-1}, \tilde{\psi}^{l,\ell-1}, z)$
      ▷ Gradient ascent on $\psi^{l,k}$ (Theorem 5)
    $\psi^{l,k} \leftarrow \tilde{\psi}^{l,N_\psi}$
  $\theta^k \leftarrow Optim(\sum_{l=1}^{N_F}\nabla_\theta\mathcal{J}^l(\theta^{k-1}, \psi^{l,k}), \eta_\theta)$ ▷ Gradient-based update of θ (Theorem 2)

4.1 Gaussian pyramid of patches

The simplest local features we can define are patches. Patches are small square sub-images of size $s \times s$, and we define the ith patch extractor P_i as the linear operation that extracts the $s \times s$ pixels around the pixel i of an image. In this paragraph, the feature operators F_i are defined from such patch extractors operating at different scales.

As previously mentioned, prescribing the distribution of patches of different sizes is necessary to model image patterns of different scales [31, 13]. Then, in order to construct a multi-scale collection of patches, we create a pyramid of down-sampled and blurred images. For each scale $l = 1, \dots, N_F$, we use a linear blurring and down-sampling operator G_s that computes, for an image u with $n = n_W \times n_H$ pixels, a reduced version $u_l = G_l u$ with $n_l = \frac{n_H}{2^{l-1}} \times \frac{n_W}{2^{l-1}}$ pixels. We then define by P_i^l the operator that extracts the ith patch of the blurred and down-sampled image u_l, that is $P_i^l = P_i \circ G_l$. In [23] we demonstrated that state-of-the art texture synthesis results can be obtained with the collection of features $\{P_i^l\}_{i,l}$.

In the following, when using such patch features within our optimal transport framework to estimate an image θ, we consider the feature index l for scales and the loss we aim at minimizing writes

$$\mathcal{L}_{\text{patch}}(\theta) = \sum_{l=1}^{N_F} \mathrm{OT}_c(\mu_{\text{pat}}^l(\theta), \nu_{\text{pat}}^l), \quad \text{where} \quad \mu_{\text{pat}}^l(\theta) = \frac{1}{n_l} \sum_{i=1}^{n_l} (P_i \circ G_l \circ g_\theta) \sharp \zeta \tag{44}$$

and where similarly ν_{pat}^l is the blurred and down-sampled patch distribution of the example image example u_0. The loss (44) is referred to as the **GOTEX-patch** loss.

4.2 VGG features

The use of *deep features* extracted from a deep neural network has proven to be successful for texture generation. In [14], the authors proposed to use the outputs from different layers of the pre-trained VGG-19 network from [45]. This fully convolutional network was introduced by the Visual Geometry Group (VGG) from the University of Oxford. It consists of a sequence of sixteen convolutional layers of kernel size 3×3 with five 2×2 max-pooling layers with stride 2, and three final fully connected layers (see [45] for the detailed architecture).

In this work, we are using the VGG-19 network with weights that were pre-trained for an image classification task on the ImageNet dataset [8]. We also replaced the max-pooling layers with average-pooling layers as done in [14] in order to reduce checkerboard artifacts. For the features, we consider the outputs from five layers ($N_F = 5$): the first convolutional layer and then the first convolutional layer after each pooling layer (pool1 to pool4). The related loss to optimize reads

$$\mathcal{L}_{\text{VGG}}(\theta) = \sum_{l=1}^{N_F} \mathrm{OT}_c(\mu_{\text{VGG}}^l(\theta), \nu_{\text{VGG}}^l), \quad \text{with} \quad \mu_{\text{VGG}}^l(\theta) = \frac{1}{n_l} \sum_{i=1}^{n_l} (F_i^l \circ g_t) \sharp \zeta \tag{45}$$

and where F_i^l corresponds to the normalized[3] ith output from the layer l of the VGG-19 network and n_l the spatial size of this layer, and where similarly ν_{VGG}^l is the feature distribution of the example at layer l. The loss (45) will be referred to as the **GOTEX-VGG** loss.

4.3 Mixing features

The major flaw from VGG-19 feature decomposition (see Section 5.3.2) is that visual artifacts such as checkerboard patterns or color inconsistencies can appear on the synthesized textures. Generally speaking, these problems are solved with a posterior histogram equalization or with the use of a median filter on the final result. On the other hand, with the multiscale patch decomposition (Section 4.1) one may fail to recover thin details at larger scales (due to the blurring in the Gaussian pyramid). Conversely, patches at the image scale accurately restore both the local details and the color consistency of the image, while VGG features from deeper layers can represent well larger patterns of the image.

Since several distributions of features can be combined with the GOTEX model, we propose to both enforce the VGG-19 features from deep layers – which represent large structures and global geometry from the target texture – and patches features from the firsts scales of the image – which represent local details and color distributions from the target texture. We therefore propose to optimize the following loss

$$\mathcal{L}_{\text{mix}}(\theta) = \lambda \mathrm{OT}_c(\mu_{\text{pat}}^1(\theta), \nu_{\text{pat}}^1) + \sum_{l=2}^{N_F} \mathrm{OT}_c(\mu_{\text{VGG}}^l(\theta), \nu_{\text{VGG}}^l) \tag{46}$$

[3]Features maps are normalized by the tensor dimension, that is spatial and channels size.

where the scalar $\lambda \geq 0$ is used to balanced the discrepancy of the range values between patches and VGG features. The loss (46) with $\lambda = 1$ will be referred to as the **GOTEX-mix** loss.

For the sake of completeness, we also consider a loss that includes patch and VGG features at all scales:

$$\mathcal{L}_{\text{all}}(\theta) = \lambda \sum_{l=1}^{N_F} \mathrm{OT}_c(\mu_{\text{pat}}^l(\theta), \nu_{\text{pat}}^l) + \sum_{l=1}^{N_F} \mathrm{OT}_c(\mu_{\text{VGG}}^l(\theta), \nu_{\text{VGG}}^l) \tag{47}$$

For $\lambda = 1$, this will be referred to as the **GOTEX-all** loss.

4.4 Texture barycenters

In the framework of Gatys et al. [14], the Gram loss does not represent a distance between distributions. Therefore, it does not provide relevant results when dealing with texture interpolation. This issue has already been pointed out for instance in [49]. In contrast, when $c(x,y) = \|x-y\|^p$, the optimal transport cost is related to a true distance

$$W_p(\mu, \nu) = \mathrm{OT}_c(\mu, \nu)^{\frac{1}{p}} \tag{48}$$

called the p-Wasserstein distance, which allows to define relevant texture interpolation paths. Indeed, restricting to the cost $c(x,y) = ||x-y||^2$, we will now show the connection between 2-Wasserstein barycenters and the interpolation of K textures $u_0, \dots, u_{K-1}$. We recall that $\mathcal{L}_{\text{GOTEX}}(\theta, u_k)$ is the loss function (42) related to the exemplar texture u_k. This loss depends on the feature distribution $(\nu_k^l)_{1 \leq l \leq N_F}$ extracted from u_k, for any set of features described in the last paragraphs. Given also weights $\alpha_0, \dots, \alpha_{K-1} \geq 0$ such that $\alpha_0 + \ldots + \alpha_{K-1} = 1$, we propose to define a Wasserstein barycenter of the textures $(u_0, \dots, u_{K-1})$ with weights $(\alpha_1, \dots, \alpha_K)$ as the texture distribution $g_\theta \sharp \zeta$ where θ minimizes the quantity

$$\sum_{k=0}^{K-1} \alpha_k \mathcal{L}_{\text{GOTEX}}(\theta, u_k) = \sum_{k=0}^{K-1} \sum_{l=1}^{N_F} \alpha_k W_2^2(\mu_\theta^l, \nu_k^l) \tag{49}$$

In comparison, for each separate feature l, the usual Wasserstein barycenter of $\nu_1^l, \dots, \nu_K^l$ defined in [1] is a solution of

$$\arg\min_{\mu^l} \sum_{k=0}^{K-1} \alpha_k W_2^2(\mu^l, \nu_k^l). \tag{50}$$

Our definition of Wasserstein texture barycenter is thus an adaptation of this notion with two modifications. First, for each feature l, μ_θ^l is constrained to be the distribution of the feature l of $g_\theta \sharp \zeta$. Second, all features l are treated simultaneously. The barycentric loss (49) thus provides an interpolation path between textures $u_0, \dots, u_{K-1}$ while belonging to a set of texture models constrained by the choice of $g_\theta \sharp \zeta$.

5 Single Image GOTEX

In this section we focus on the particular case of synthesizing a single image. This corresponds to the case where the model just generates a single image θ by taking $g_\theta(z) = \theta - z$ for all z and $\zeta = \delta_0$ (see remark 2). This amounts to minimize w.r.t. the image θ the optimal transport cost between its discrete feature distribution μ_θ and a discrete target feature distribution ν. This yields a pixelwise optimization algorithm that minimizes a fully discrete optimal transport cost between patch distributions.

We present in section 5.1 the single image setting of our GOTEX framework. We provide experimental results using the features described in section 4 and discuss their pros and cons. Related texture synthesis methods are briefly reviewed in section 5.2. In particular, we show that using image patches as features in this single image setting gives an elegant interpretation of the algorithm of [19] which combines the patch-based optimization framework from [31] with optimal transport.

Our Single Image GOTEX approach, using either patch or deep features, is compared in section 5.3 to the previously introduced state-of-the-art methods. We illustrate that the proposed hybrid losses (46) and (47) allows to capture higher order statistics than the perceptual loss proposed in [14]. Finally, we show in section 5.4 that our GOTEX framework is well-suited for other tasks related to texture imaging such as texture barycenters and texture inpainting.

5.1 Single texture synthesis

In the case of a single image generation, the GOTEX framework amounts to minimize a discrete optimal transport cost. Let $\theta \in \mathbf{R}^n$ be the image to synthesize with n pixels and $\mu_\theta = \frac{1}{n}\sum_{i=1}^{n} \delta_{F_i(\theta)}$ its discrete patch distribution. In order to impose on the image θ the patch distribution $\nu = \frac{1}{m}\sum_{j=1}^{m} \delta_{F_j(u_0)}$ of the exemplar image u_0, we aim at solving problem (14) that here simplifies to

$$\min_{\theta\in\mathbf{R}^n} \mathrm{OT}_c\left(\mu_\theta,\nu\right) = \min_{\theta\in\mathbf{R}^n} \max_{\psi\in\mathbf{R}^m} \mathcal{J}(\theta,\psi). \tag{51}$$

In this particular discrete case, the functional of interest writes

$$\mathcal{J}(\theta,\psi) = \frac{1}{n}\sum_{i=1}^{n} \psi^c(F_i(\theta)) + \frac{1}{m}\sum_{j=1}^{m} \psi_j, \tag{52}$$

with

$$\psi^c(F_i(\theta)) = \min_j \left[c(F_i(\theta), F_j(u_0)) - \psi_j\right]. \tag{53}$$

The minimization of (51) is then done with the proposed algorithm 1 which is in this case **not** stochastic. From now on and for all the experiments, we set the cost c to be the quadratic cost and in the following we illustrate the versatility of our framework with synthesis experiments on various textures and features. After introducing the experimental set-up in section 5.1.1, results obtained from different feature choices (exposed in section 4) are discussed in section 5.1.2. In section 5.1.3 we finally discuss the robustness of the method with respect to the choice of the initial image.

5.1.1 Experimental setting

In this section, the L-BFGS algorithm [35] is used as the optimizer in Algorithm 1. We resort to the PyTorch implementation with the default parameter setting (for instance, $\eta_\theta = 1$). The initial image θ^0 is randomly sampled from a Gaussian white noise of mean m_{target} and variance $0.01\sigma_{\text{target}}$ where m_{target} and σ_{target} are respectively the mean and the variance of the target image. The patch size used for the Gaussian pyramid of patches is 4×4 pixels at each scale, and the number of scales is $N_F = 4$. We also use the KeOps library [5] to define the cost matching kernel and compute the biased nearest neighbor map (57) efficiently on GPU. The number of inner iterations N_ψ is set to 10. Example images are all of size 256×256 in order to compute VGG features appropriately[4]. The VGG features are computed using the pre-trained VGG-19 network from the PyTorch library.

5.1.2 Experimental results with various features

Here we present the single image synthesis results from our GOTEX framework using the losses GOTEX-patch (44) , GOTEX-VGG (45), GOTEX-mix (46) and GOTEX-all (47). We present in Figure 2 these results for 4 given exemplar textures. Since we use the same framework either with patches or with VGG features, we are able to discuss precisely the pros and cons of each feature representation. The GOTEX-patch method (second column of Fig. 2) shows good color consistency and respects the texture statistics at each scale. On the other hand, it tends to produce slightly smoother results. The GOTEX-VGG approach (third column of Fig. 2) produces sharper results but at the cost of color inconsistencies and visual artifacts such as checkerboard artifacts [37]. These artifacts can be explained by the back-propagation through the non-linear VGG network and are known to appear with perceptual losses based on VGG, see e.g. [41, 24]. The over-smoothing effect when using patches is explained by the fact that the algorithm tends to make a compromise between the patches at each iteration (see section 5.2.1 for details on this point). Finally, by combining patches and VGG features (fourth column of Fig. 2), the GOTEX-mix method tackles both the color and artifact issues from the VGG features and the smoothing effect observed with GOTEX-patch. The GOTEX-all method (fifth column of Fig. 2) produces similar results but with long-range structures that are slightly better retrieved.

5.1.3 Importance of the initialization

When using the VGG features, the choice of the initial image θ^0 has a strong impact on the final result. This can be explained by the fact that deeper layers in the VGG network only encode structures and do not encode colors. Therefore, the minimization of GOTEX-VGG is likely to find a texture that is a local minimum but has the wrong color distribution. This behavior was already documented in [14] where the authors proposed to perform an histogram matching at the end of the synthesis process. On the contrary, we show in Figure 3 that using patches provides a very consistent synthesis method, not depending on the initial image θ^0 .

[4] Note that a multi-scale approach is required for high resolution synthesis with VGG features as studied in [15].

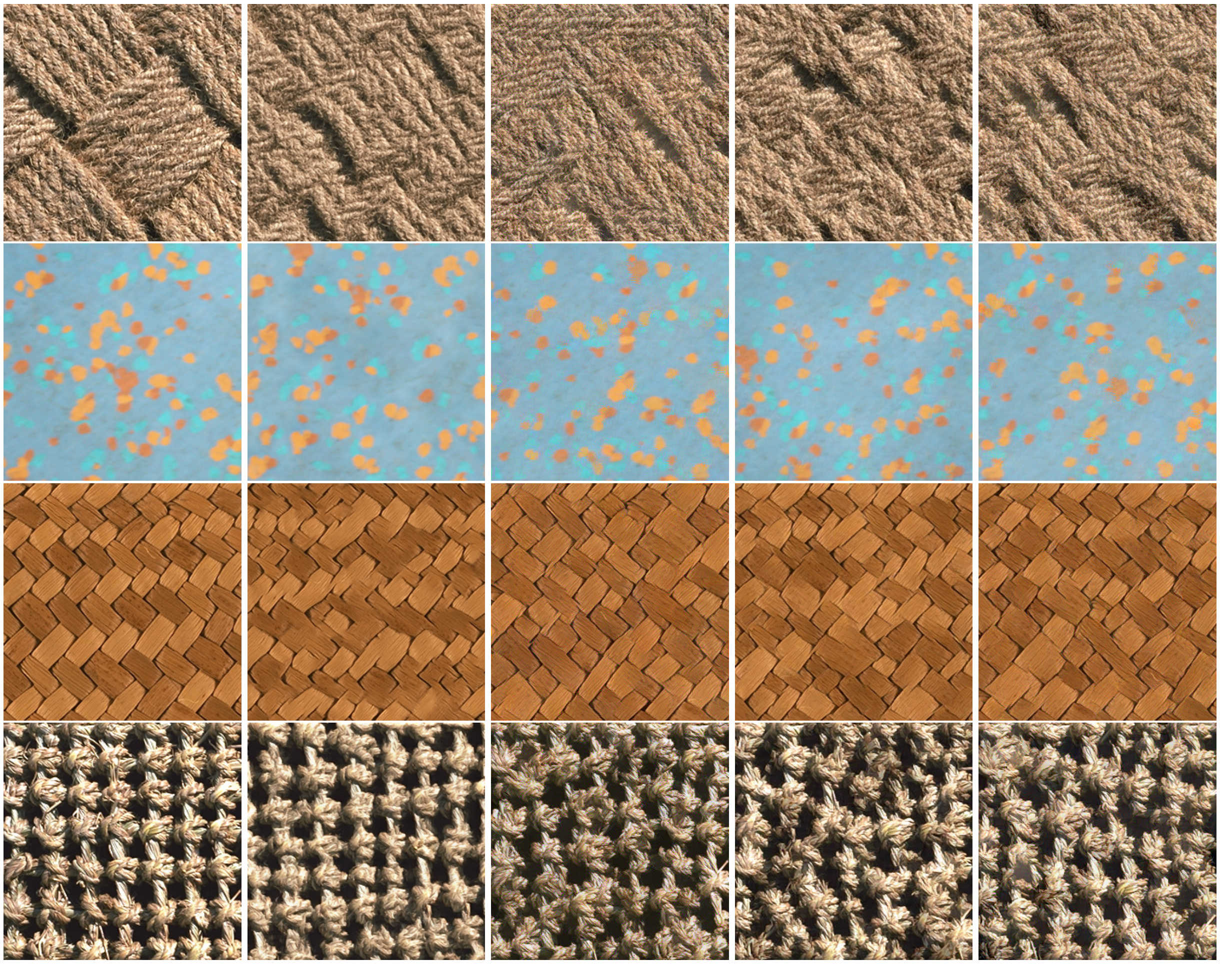


Figure 2: Results of GOTEX (Alg. 1) for image optimization using various combinations of features: patches (*GOTEX-patch*), VGG (*GOTEX-VGG*), mixing patches from higher scales with VGG from lower scales (*GOTEX-mix*), and combining all features (*GOTEX-all*).

5.2 Related works on optimal patch transport

We first show in section 5.2.1 that using image patches as features in the single image setting gives an elegant interpretation of the algorithm of [19] which combines the patch-based optimization framework from [31] with optimal transport. In section 5.2.2, we provide a review of models from the literature based on approximated optimal transport. We also detail how the proposed framework built upon semi-discrete optimal transport both encompasses these models and alleviates some of their caveats.

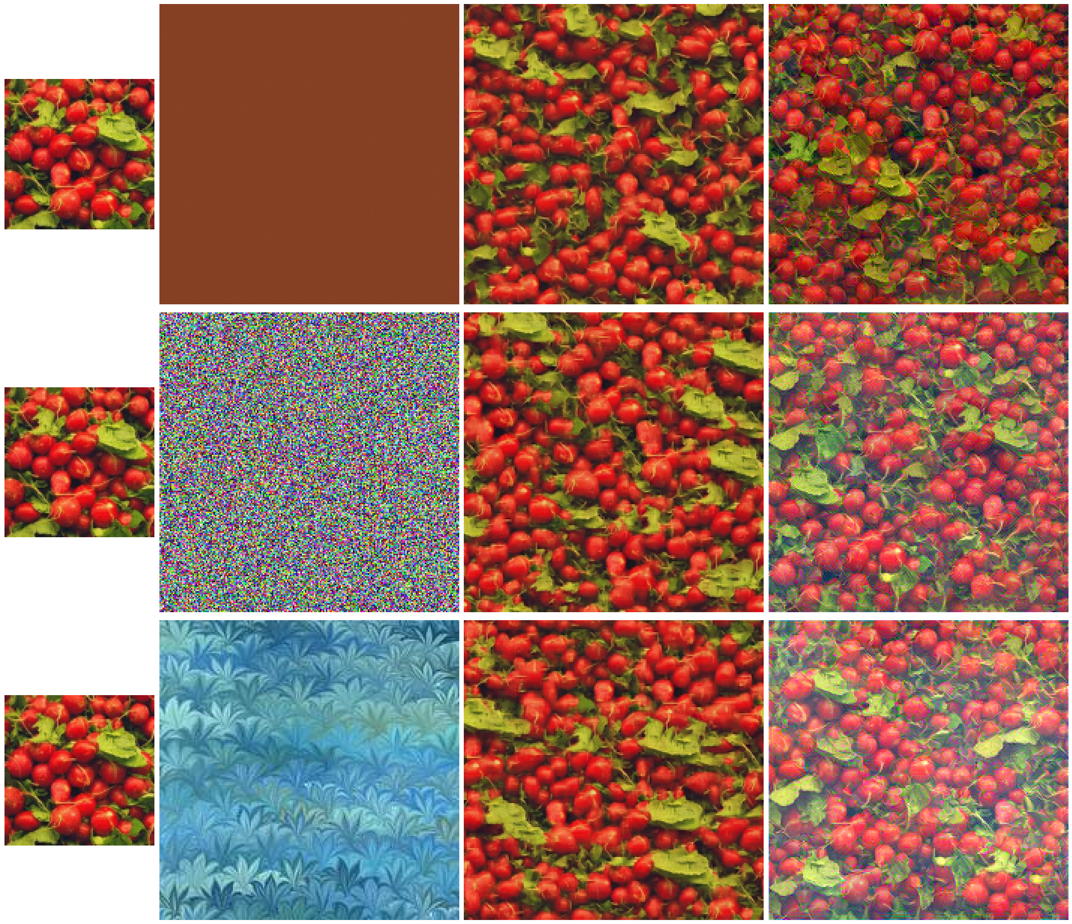
5.2.1 The specific case of patch representation

We detail here the single image optimization framework in the case where we use the patch representation (as defined in section 4.1). Recall that, as summarized in Alg. 1, all scales $l = 1, \dots, N_F$ are treated simultaneously with GOTEX, and we aim at solving as many optimal transport problems. For the sake of simplicity, we focus on the single scale $l = 1$ case in this paragraph. In this particular case with patch features, the functional $\mathcal{J}$ from (52) writes

$$\mathcal{J}(\theta, \psi) = \frac{1}{n} \sum_{i=1}^{n} \psi^c(P_i\theta) + \frac{1}{m} \sum_{j=1}^{m} \psi_j, \tag{54}$$

with

$$\psi^c(P_i\theta) = \min_j \left[c(P_i\theta, P_j\theta) - \psi_j\right]. \tag{55}$$



a. Sample b. Initialization c. GOTEX-patch d. GOTEX-VGG

Figure 3: Both GOTEX-patch and GOTEX-VGG are run for the same 100×100 sample (a) with three initial images (b). GOTEX-path produces faithful 200×200 synthesis (c) for any initialization whereas the GOTEX-VGG results (d) tends to produce color inconsistencies and artifacts when the color palette of the initial guess is not close enough to the target image.

Now, we will see that using the quadratic cost c, and using a simple gradient descent for the update on θ, the GOTEX algorithm 1 admits an interpretation in terms of iterated weighted nearest neighbor assignments. Indeed, in the case $c(x,y) = \frac{1}{2}\|x-y\|^2$, Theorem 3 ensures that

$$\nabla_\theta \mathcal{J}(\theta^{k-1}, \psi^k) = \frac{1}{n}\left(\sum_{i=1}^n P_i^T P_i \theta^{k-1} - \sum_{i=1}^n P_i^T P_{\sigma^k(i)} u_0\right), \tag{56}$$

at any point (θ^{k-1}, ψ^k) where we can uniquely define

$$\sigma^k(i) = \arg\min_j \frac{1}{2}\|P_i\theta^k - P_j u_0\|^2 - \psi_j^k \quad \forall i = 1, \ldots, n. \tag{57}$$

Notice that P_j is a linear operator whose adjoint operator P_j^T maps a given patch q to an image whose j-patch is q and is zero elsewhere. Therefore $\sum_{i=1}^n P_i^T$ corresponds to a uniform patch aggregation. To simplify, we consider periodic conditions for patch extraction, so that $\sum_{i=1}^n P_i^T P_i = s^2\mathrm{I}$, where s^2 denotes the number of pixels in the $s \times s$ patches. Hence, from (56) and considering a step size $\eta\frac{n}{s^2}$, $\eta > 0$, the update of u through gradient descent writes

$$\theta^k = (1-\eta)\theta^{k-1} + \eta v^k, \tag{58}$$

where $v^k = \frac{1}{s^2}\sum_{i=1}^n P_i^T P_{\sigma^k(i)} u_0$ is the image formed with patches from the exemplar image u_0 which are the nearest neighbors to the patches of θ^k in the sense of (57). The gradient step then mixes the current image θ^k with v^k. In the case $\psi = 0$, the minimum in (57) is reached by associating to each patch of θ^k its ℓ_2 nearest neighbor in the set $\{P_1 u_0, \dots, P_n u_0\}$. As described below, the case $\psi = 0$ exactly corresponds to the texture optimization method [31].

5.2.2 Comparison with previous works on optimal patch transport

Texture Optimization (TexOptim) In [31], a multi-scale algorithm similar to ours is used for synthesis. It includes additional optimization "tricks" to accelerate the practical convergence. First, the optimal transport cost between patch distributions is replaced by nearest-neighbor matching, which itself results in visible discrepancy between color distributions for texture synthesis (see e.g. Figure 4, column b). Additionally, a coarse-to-fine (a.k.a *multi-grid*) approach is used, whereas every scales are optimized simultaneously in the proposed algorithm. At a given scale l (using the Gaussian pyramid of patches defined in 4.1), the problem originally formulated in [31] reads as an explicit patch matching

$$\sum_{i=1}^{n_l} \min_j \|P_i^l \theta - P_j^l u_0\|^r \tag{59}$$

where $r = 0.8$ to enforce consistency between overlapping patches, in such a way that synthesized textures are local verbatim copies of the original image. In practice, this non-convex optimization problem is addressed using an iterative reweighted least square methods (IRLS): denoting $\tilde{\sigma}^k(i)$ the nearest-neighbor (NN) matching defined from (57) by setting $\psi_j^k = 0$

$$\tilde{\sigma}^k(i) = \arg\min_j \frac{1}{2}\|P_i^l \theta^k - P_j^l u_0\|^2 \quad \forall\, i = 1, \dots, n_l \ ,$$

the updated image θ^k at scale l reads (again, considering periodic conditions)

$$\theta^k(i) = \tfrac{1}{s^2} \sum_{i=1}^{n_l} w_i {P_i^l}^T P_{\tilde{\sigma}^k(i)}^l u_0$$

where weights w_i are defined according to IRLS

$$w_i = \|P_i^l \theta^k - P_{\tilde{\sigma}^k(i)}^l u_0\|^{r-2}.$$

Without statistical consistency guaranteed by optimal transport, an important aspect of this approach is the initialization: to ensure the statistical consistency at the coarsest scale, a random permutation of the patch of u_0 is used to set θ^0. Last, to avoid the excessive blurring resulting from overlapping patches, a stride of $s/4$ is used to sample large patches (ranging from $s = 8$ to $s = 32$ during optimization to enforce local copy).

Optimal Patch Assignment (OPA) To overcome the statistical inconsistency from nearest-neighbor matching, [19] enforces instead the optimal patch assignment between the discrete distribution of n_l patches. More precisely, the objective loss function now reads as (at a given scale l)

$$\min_{\sigma \in \Sigma_{n_l}} \sum_{i=1}^{n_l} \|P_i^l \theta - P_{\sigma(i)}^l u_0\|_{1,2} \tag{60}$$

where Σ_n is the set of n permutations, and $\|.\|_{1,2}$ stands for the sum, over pixel coordinates i, of the Euclidean norm of pixelwise color vectors. By setting $r = 1$, the problem of optimizing θ for a fixed assignment σ corresponds to computing a color median. This helps reducing the blur obtained from averaging overlapping patches when using $r = 2$ instead. Rather than resorting to the proposed semi-discrete formulation, an Hungarian algorithm [30] is used instead to solve the optimal assignment between patches at each iteration. Note that a similar but faster approach has been proposed in [50], mainly approximating optimal assignments by soft assignments computed from the Sinkhorn Algorithm [7]. The main drawbacks of these approaches is computation time and memory requirements to compute the assignment maps σ. For instance, both methods require to compute and store the cost matrix for finding patch correspondences. This becomes prohibitive for large image size and it cannot benefit from GPU acceleration.

To accelerate convergence, as in [31], large patches of $s = 8$ pixels are therefore sampled on a decimated grid (i.e. stride of 2). A Douglas-Rachford optimization algorithm is also required after each optimal patch assignment to compute the corresponding color median to update θ. In Figure 4, one can observe that the results are close to the ones obtained with the GOTEX Algorithm 1, yet only using 4×4 patches without stride nor median.

Sliced Wasserstein transport In [40], an approximate optimal transport cost function coined *Sliced Wasserstein distance* was introduced for texture synthesis and mixing. This framework is inspired by [38] where an image is progressively synthesized from coarse to fine scale by sequentially matching its statistics to an exemplar image. Originally, first and second order statistics of wavelet coefficients across scale and space were considered, restricting the generated textures to short range correlations. In [40], the distribution of patches of wavelet coefficients was considered instead, to successfully synthesize large range structures. The projection of patch distributions from the synthesized image to the desired distribution is ensured by minimizing the Sliced Wasserstein cost function using stochastic gradient descent. Since then, it has been used in various imaging problems involving statistical comparison, such as in [28] to train auto-encoders.

To appropriately compare this approach to the proposed multi-scale patch-based optimization, we now consider adaptation of the GOTEX Algorithm 1 to the case of Sliced Wasserstein, in such a way that the loss function to minimize becomes

$$\mathcal{L}_{\text{SW}}(\theta) = \sum_{l=1}^{N_F} \text{SW}^2(\mu_\theta^l, \nu^l). \tag{61}$$

At a given scale l, the quadratic Sliced Wasserstein cost SW^2 between discrete distributions $\mu_\theta^l = \frac{1}{n}\sum_{i=1}^n \delta_{P_i\theta}$ and ν^l writes as the expected transport cost of the projected distributions

$$\text{SW}^2(\mu_\theta^l, \nu^l) = \mathbf{E}_{\omega\sim\mathcal{U}(\mathbb{S}^d)} \min_{\sigma\in\Sigma_{n_l}} \frac{1}{n_l}\sum_{i=1}^{n_l} \langle P_i^l\theta - P_{\sigma(i)}^l u_0, \omega\rangle^2 \tag{62}$$

where $\mathcal{U}(\mathbb{S}^d)$ indicates the uniform distribution on the d-dimensional sphere. The gradient $\mathcal{L}_{\text{SW}}$ can be written explicitly [40], but one can also rely on auto-differentiation as done for instance in [28] and for our experiments. The main limitations of this approximation are, to begin with, that it requires the two distributions to have the same number of samples n with uniform probability, that the required number k of drawn directions ω for stochastic optimization increases with the dimension d of the features, and that it requires to sort values which cannot benefit from GPU acceleration.

5.3 Experimental comparisons

We now illustrate the benefit of the GOTEX method with respect to related methods in the literature. We first focus on patch-based method and then study VGG features.

5.3.1 Comparisons with patch-based methods

Figure 4 illustrates the results of different methods discussed in section 5.2.2 using image optimization based solely on patch representation. As already mentioned, when comparing TexOptim (Fig. 4.b) to OPA (Fig. 4.c), or any other method based on optimal transport (Fig. 4.d and e), the lack of statistical consistency is visually striking. While benefiting from a noticeable speed-up when approximating OPA with Sliced Wasserstein, the resulting texture is more blurry (mostly noticeable within the flowers in Fig. 4.d). This is mainly due to the fact that the optimization of SW is stochastic by nature, sampling randomly new directions at each step. By contrast, the proposed algorithm manages to produce qualitative results (Fig. 4.e), without any approximations.

5.3.2 Comparison with VGG-based optimization

Figure 5 compares the synthesized results from methods based on VGG features exclusively. As reported in [21] and illustrated here, using Sliced-Wasserstein gives slightly better results than the original approach based on Gram matrices regarding color consistency, even when using the post-processing based on color histogram matching advocated in [14]. Our GOTEX-VGG approach produces decent results, with the same color inconsistencies than the one observed with the method of Gatys et al. [14]. As discussed in section 5.1.2, this issue can be solved with the GOTEX-mix loss (46) that constrains both VGG and patch distributions.

5.4 Other applications

We finally present adaptations of the GOTEX framework to tackle two related synthesis problems: texture inpainting and texture interpolation.

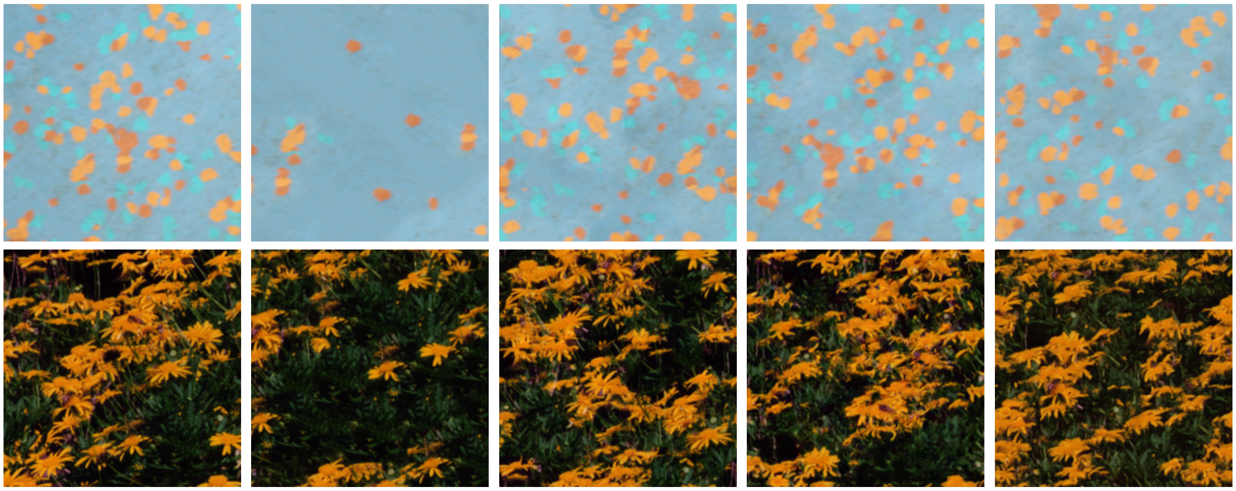


Figure 4: Comparison of exemplar-based texture synthesis methods using patch-based image optimization and various optimal transport approximations. a) displays the exemplar image. b) shows the texture optimization results from [31] using coarse-to-fine optimization and nearest-neighbor patch assignment. c) uses instead Optimal Patch Assignment (OPA) [19], enforcing the target patch distribution. d) is based on the proposed loss function (44) where OT is approximated by Sliced Wasserstein (SW) cost. e) is the proposed semi-discrete approach, solving more accurately the OT problem. See the text for more details.

Figure 5: As in Figure 4, we compare the results of the proposed Algorithm 1 (GOTEX-VGG) for image optimization based on VGG (already exposed in Fig. 2), as pioneered by [14] (Gram-VGG) and with the Sliced Wasserstein approximation (SW-VGG) studied in [21].

5.4.1 Texture inpainting

The framework using the OT-patch loss 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. Since the comparison of patch distributions with OT allows to capture highly non-Gaussian behavior, this texture inpainting scheme can treat highly structured textures, as opposed to the Gaussian model of [12] which can only deal with unstructured textures. Besides, the fact that the optimization problem can be naturally restricted to the masked area permits, again, to inherently solve patch aggregation problems. In contrast to the algorithm of [34] based on an ad-hoc aggregation technique at the border of the mask, our inpainting results do not suffer from the same blur artifacts on this region, as can be seen on Fig. 6.

Figure 6: Texture inpainting on various masked textures of size 128×128 with $s = 4$ and $L = 3$ using a slightly adapted version of Alg. 1. The method is able to fill properly the masked region while agreeing quite convincingly with the surrounding content around the mask boundary.

5.4.2 Texture barycenters

We now tackle the problem of interpolating between different exemplar textures. To that end, we use the GOTEX barycentric loss (49) presented in section 4.4. For $K = 2$, it corresponds to finding a new texture θ whose feature distribution is a Wasserstein interpolation (in the sense of (49)) of the feature distributions extracted from the two exemplar textures u_0 and u_1. We provide in Fig. 7 the interpolation results θ_t between various textures u_0 and u_1, for both patchs and VGG-19 features. In these experiments, the barycenter θ_t corresponds to the solution of (49) for different weights $\alpha_0 = 1 - \alpha_1 = t \in \{0.2, \cdots, 0.8\}$. We also compare these results to the framework of [14] that realizes texture interpolation by minimizing the weighted sum of Gram losses (3) of VGG features. Since the distance between Gram matrices of features does not represent a distance between feature distributions, this method produces textures with distinct spatial parts that either belong to one or to the other texture u_0 and u_1. On the other hand, our approach properly interpolates between texture contents.

6 Generative model

In this section, we consider the problem of training a network to generate images that have prescribed feature distributions at multiple scales. To do so, we use our framework with the generative model g_θ being the feed-forward deep convolutional network introduced in [48] for texture generation. Then we present some visual results together with a comparison with existing methods.

Estimating a generative model corresponds to a semi-discrete problem, where one to be able to sample a continuous distribution $g_\theta \sharp \zeta$ which feature distribution $\mu_\theta = \frac{1}{n} \sum_{i=1}^{n} (F_i \circ g_\theta) \sharp \zeta$ fits the one of a discrete example distribution $\nu = \sum_{j=1}^{m} \delta_{y_j}$. The semi-discrete formulation of optimal transport is thus perfectly adapted to deal with the estimation of a generative model.

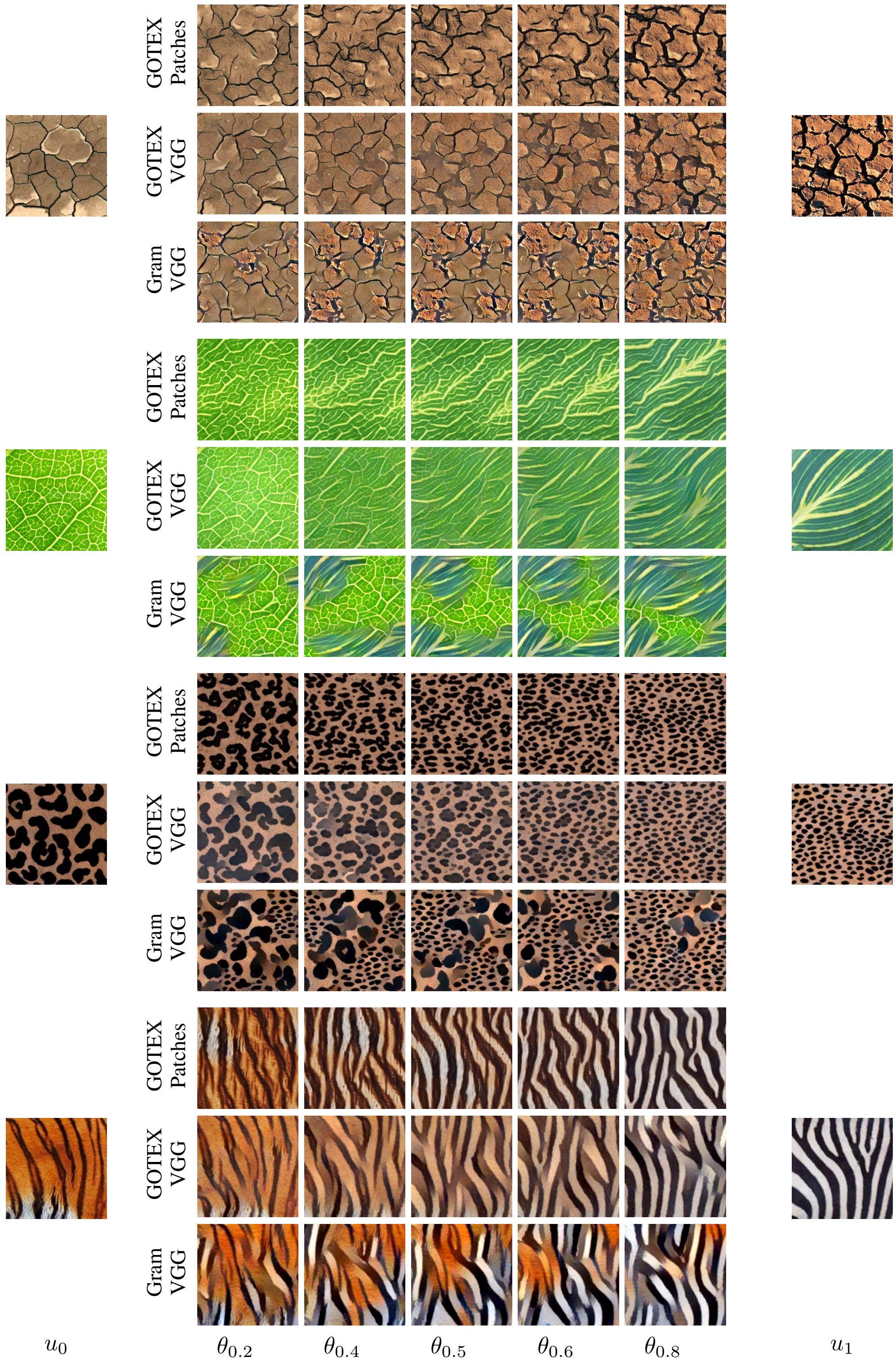


Figure 7: Interpolations θ_t between exemplar textures u_0 and u_1, obtained by minimizing the barycentric GOTEX loss (49) with $\alpha_0 = 1 - \alpha_1 = t \in \{0.2, \cdots, 0.8\}$ for either patches or VGG features. Barycenters computed with respect to Gram losses (3) of VGG features, as in [14], are presented for comparison. The approach of [14] just copies and pastes different parts of the two exemplar textures, whereas the GOTEX framework realizes visually plausible texture interpolations for any kind of involved feature.

6.1 Experimental setting

Neural network architecture We consider the feed-forward texture synthesis model introduced in [48] that we refer to as *TexNet*. This model takes as input a latent variable $z = \{z_0, \ldots, z_M\}$ constituted of $M+1$ inputs z_l of size $\propto 2^l$ (with $M = 4$ as advocated in [48] for texture synthesis). The first input z_0 passes through a convolutional block and a block of upsampling by a factor 2, then the result is concatenated with z_1 and passes again through a convolutional and upsamling block and so on. For the detailed architecture, we refer to the Figure 2 of [48].

One of the key advantage of this network in comparison to other methods is its small number of parameters (around 65K), enabling fast convergence during optimization on a single image. It has been originally designed to synthesize textures by minimizing the Gram-VGG loss introduced in [14]. We next demonstrate that the parameters of such a generative network can be learned with the GOTEX algorithm, *i.e.* by only enforcing the feature distributions at various scales.

Optimization In our PyTorch implementation, we use this time the Adam optimizer [27] to estimate the parameters θ. For the GOTEX-patch algorithm, 10000 iterations have been used with a learning-rate $\eta_\theta = 0.01$. An averaged stochastic gradient ascent with 100 inner iterations is used for computing ψ^*. In total, one million 4×4 patches at $N_F = 4$ different scales are therefore sampled to train the neural network. For GOTEX-VGG, 10000 images are sampled during training (batch of 1) with the same optimizer. In these setting, 10 hours are required to train each generator with a GPU Nvidia K40m.

Compared methods For experiments, Pytorch implementations of SinGAN [5], PSGAN [6] and TexNet [7] were run with their default parameters. Regarding TexTo [34], a similar experimental setting has been used, with 10^7 iterations of ASGD and 7×7 patches at 4 different scales. Note that a pre-processing step is required to perform the bi-level clustering of the target patch distribution.

The Sliced Wasserstein loss used to approximate optimal transport within our framework is optimized using 5000 iterations of the Adam optimizer. At each step, all the 4×4 patches are extracted from a 256×256 synthesized texture (i.e. a batch of 1 image). The loss itself is based on d random directions in addition to the canonical basis, where $d = 48$ is the patch dimension.

6.2 Experimental results and discussion

Figures 8 and 9 display four synthesized textures with the proposed GOTEX framework and five relevant synthesis methods from the literature based on patches or deep features.

Patch representation We first compare GOTEX-patch (Fig. 8.b) with the Sliced Wasserstein approximation (Fig. 8.c) discussed in the previous section. Since the SW cost is computed at each iteration, the optimization problem now requires random sampling of directions for patch projections in addition to the image sampling itself. The same architecture has been used to train both network. While the results obtained with GOTEX tends to be slightly over-smoothed, it is much more noisier with SW. This is particularly noticeable in the results of columns 2 and 3 where the original texture is regular and the syntheses with SW contain high frequencies artifacts.

We also compare to TexTo, the multi-layer OT network (Fig. 8.d) proposed in [34] where the convolutional network is replaced by approximated multi-level optimal transport maps on 7×7 patches that are averaged to synthesize an image. A major difference with the proposed method is that GOTEX solves a global optimization problem that tackles the multiscale patch distributions in a direct manner. On visual inspection, results are fairly similar, even if the proposed method does not rely on any coarse to fine optimization nor multi-level approximation. The patch aggregation step from TexTo may yield more blur than using a generative network which inherently deals with this issue. This is supported by the in-depth comparison provided in [23] based on various similarity metrics.

Perceptual representation with VGG The comparison is now carried on for VGG features, including the proposed GOTEX-mix model (Fig. 9.b), the GOTEX-VGG model (Fig. 9.c), its approximation using Sliced Wasserstein loss (SW-VGG in Fig. 9.d) that was also recently studied in [21], and the original Texture Networks from [48] (TexNet in Fig. 9.e).

[5] github.com/tamarott/SinGAN
[6] github.com/zalandoresearch/famos
[7] github.com/JorgeGtz/TextureNets_implementation

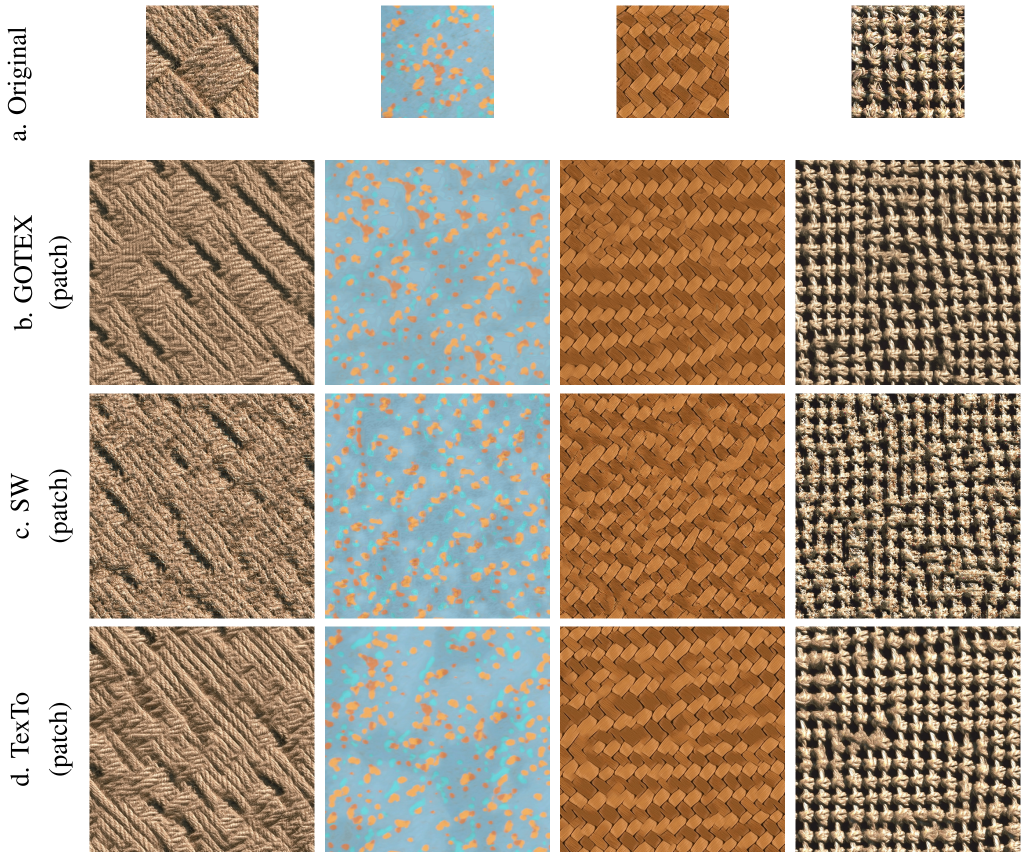


Figure 8: Texture synthesis from a generative neural network trained on a single 256×256 sample (a). Our GOTEX-patch multi-scale approach (b) using 4×4 patches (see Alg. 1) is first compared with (c) multi-scale Sliced Wasserstein (SW) approximation, and (d) TexTo [34], using an explicit network of multi-layer transportation maps on 7×7 patches rather than a convolutional network.

To begin with the latter, one can observe that the TexNet architecture creates pseudo-periodic patterns that are not visible in the original texture, as for instance in the second and fourth examples of Fig. 9.e. This limitation is already reported in [48] and has been linked to overfitting with VGG features. This effect seems to be mitigated when using optimal transport optimization.

Now, as for the image-based optimization results detailed in the previous section, it is interesting to observe that patch-based synthesis can be close to synthesis obtained when using more sophisticated features relying on deep neural networks (comparing for instance Fig. 8.b to Fig. 9.c). This is also true when considering Sliced Wasserstein approximation (Fig. 8.c vs Fig. 9.d). More precisely, multi-scale patch distributions are more effective to capture long range patterns, while VGG features manage to synthesize photo-realistic high-resolution details.

We used the histogram equalization technique discussed previously for GOTEX-VGG to avoid color inconsistencies. The benefit of mixing features is here less striking than for single image optimization shown in Section 5.1.2, as the generated samples are very close to the ones obtained by using only VGG features. Besides, the stochastic SW approximation results in noticeable decrease in synthesis quality, regarding artifacts (blur, color inconsistencies and high frequency artifacts) and long range correlations.

Adversarial techniques For the sake of completeness, we also present results from adversarial techniques that simultaneously train a discriminative network on generated patches. The first method is SinGAN [44] (Fig. 9.f), a recent generative adversarial network (GAN) technique generating images from a single example and relying on patch sampling. Note that this method, mainly focused on image reshuffling, learns implicitly during training to copy the border of the example texture by combining an adversarial loss with a least square criterion. This effect can still be noticed when generating a larger sample, especially in the corners. The second method is PSGAN [3] (Fig. 9.g), a previous approach that similarly adapts the GAN framework to the training of a single image, purely based on an

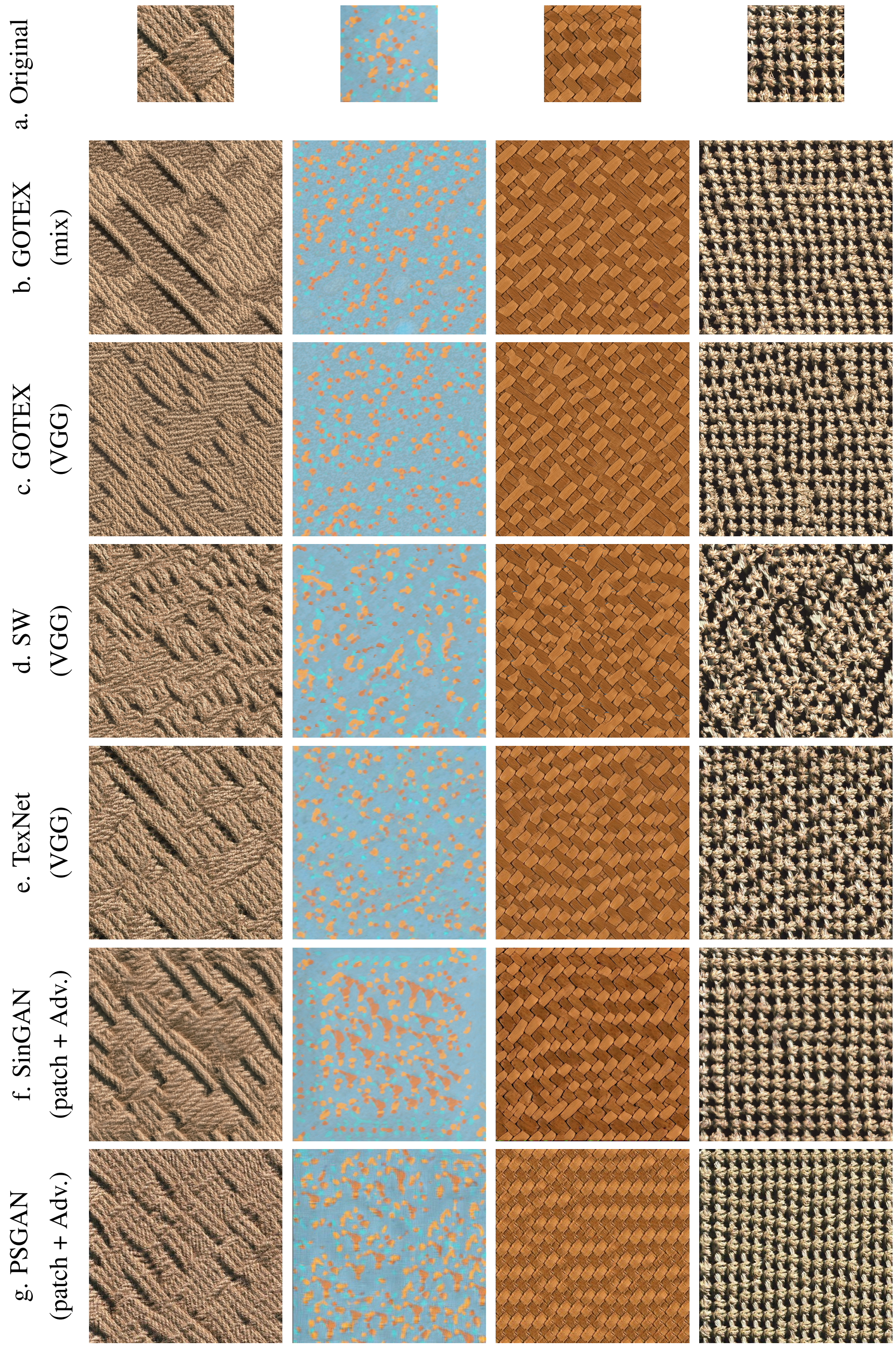


Figure 9: Texture synthesis from a generative neural network trained on a single 256×256 sample (a). Methods (b) GOTEX-mix, (c) GOTEX-VGG, (d) SW-VGG and (e) TexNet [48] use pretrained VGG features. Deep neural features are used for (f) SinGAN [44] and (g) PSGAN [3], where an adversarial network is trained on patches.

adversarial loss. As shown here, GAN struggles to train on small samples (here 256×256 pixels), which results in a lot of noticeable high frequency artifacts. Such high-frequency artifacts with GANs were already reported in [39].

7 Conclusion and discussion

In this work, we proposed a general framework for texture synthesis by optimization that allows to constrain the distributions of high-dimensional features through the use of optimal transport distances. The main contribution of this work is to exploit the semi-dual formulation of optimal transport in order to get a min-max problem with an inner concave maximization problem. This min-max problem can be solved with an efficient alternate algorithm. Besides, we provided explicit formulae for the gradients of this functional, which are useful to study and interpret the optimization algorithm. Contrary to previous methods based on approximations of the optimal transport (like the Sliced Wasserstein distance), the stochastic algorithm used here for the inner problem is guaranteed to converge towards the true optimal transport cost, which makes the global learning process more accurate. Another interest of this framework is that it is adapted to the learning of a generative model. As we have seen, such a formulation encompasses the case where one wishes to generate a single image (by optimizing directly on the pixel values) and the case where one wishes to learn a convolutional neural network that can later serve for on-the-fly synthesis. Experiments showed that both cases lead to high-quality synthesized textures. Since this method can naturally deal with various sets of features, we were able to compare synthesis results obtained by using patches or features extracted from a pre-learned neural network. This comparison showed that using multiscale patch distributions is sufficient to synthesize a wide class of textures, while avoiding low-level artifacts (e.g. drifts in the color distribution). Finally, the image-based optimization can be easily adapted to other applications, and we provided successful results of texture inpainting and texture interpolation.

This work also raises several questions, both on the theoretical and practical aspects. While the convergence of the inner maximization algorithm is proved, the convergence of the alternate algorithm (which we observed empirically) remains to be investigated. In particular, it would be interesting to see if one can justify the use of the algorithm alternating single gradient steps on each variable. On the practical side, one may try to adapt the proposed framework to more general image synthesis problems where the database is much larger. Indeed, one of the limitations of this work is that the number of variables in the semi-dual optimal transport problem corresponds to the number of points in the target distribution. Therefore, in order to deal with a much larger database, one may either resort on a batch strategy to estimate the gradient on θ, or to work with another parameterization of the dual variables (as in [43]). Finally, the different feature-based losses proposed in this work could be used for quantitative evaluation of texture synthesis methods. The design of a quality measure for texture synthesis requires a more thorough comparison of the various possible features, and may also be driven by a precise perceptual study in order to wisely combine the chosen features, and to validate the resulting criterion.

References

[1] Agueh, M., Carlier, G.: Barycenters in the wasserstein space. SIAM Journal on Mathematical Analysis **43**(2), 904–924 (2011)

[2] Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein generative adversarial networks. In: International Conference on Machine Learning. pp. 214–223 (2017)

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

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

[5] Charlier, B., Feydy, J., Glaunès, J., Collin, F.D., Durif, G.: Kernel operations on the gpu, with autodiff, without memory overflows. Journal of Machine Learning Research **22**(74), 1–6 (2021)

[6] Chen, Y., Telgarsky, M., Zhang, C., Bailey, B., Hsu, D., Peng, J.: A gradual, semi-discrete approach to generative network training via explicit wasserstein minimization. In: International Conference on Machine Learning. pp. 1071–1080. PMLR (2019)

[7] Cuturi, M.: Sinkhorn distances: Lightspeed computation of optimal transport. Advances in neural information processing systems **26**, 2292–2300 (2013)

[8] Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. Ieee (2009)

[9] Durand, T., Rabin, J., Tschumperlé, D.: Shallow multi-scale network for stylized super-resolution. In: IEEE International Conference on Image Processing (2021)

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

[11] Galerne, B., Gousseau, Y., Morel, J.M.: Random phase textures: Theory and synthesis. IEEE Trans. Image Proc. **20**(1), 257 – 267 (2011)

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

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

[14] Gatys, L., Ecker, A.S., Bethge, M.: Texture synthesis using convolutional neural networks. In: NIPS. pp. 262–270 (2015)

[15] Gatys, L.A., Ecker, A.S., Bethge, M., Hertzmann, A., Shechtman, E.: Controlling perceptual factors in neural style transfer. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3985–3993 (2017)

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

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

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

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

[20] Heeger, D.J., Bergen, J.R.: Pyramid-based texture analysis/synthesis. In: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques. pp. 229–238. ACM (1995)

[21] Heitz, E., Vanhoey, K., Chambon, T., Belcour, L.: A sliced wasserstein loss for neural texture synthesis. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 9412–9420 (2021)

[22] Houdard, A., Bouveyron, C., Delon, J.: High-dimensional mixture models for unsupervised image denoising (hdmi). SIAM Journal on Imaging Sciences **11**(4), 2815–2846 (2018)

[23] Houdard, A., Leclaire, A., Papadakis, N., Rabin, J.: Wasserstein generative models for patch-based texture synthesis. In: Elmoataz, A., Fadili, J., Quéau, Y., Rabin, J., Simon, L. (eds.) Scale Space and Variational Methods in Computer Vision (SSVM'2. pp. 269–280. Springer International Publishing, Cham (2021)

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

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

[26] Kaspar, A., Neubert, B., Lischinski, D., Pauly, M., Kopf, J.: Self tuning texture optimization. In: Computer Graphics Forum. vol. 34, pp. 349–359 (2015)

[27] Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: ICLR (2014)

[28] Kolouri, S., Pope, P.E., Martin, C.E., Rohde, G.K.: Sliced wasserstein auto-encoders. In: International Conference on Learning Representations (2018)

[29] Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems **25**, 1097–1105 (2012)

[30] Kuhn, H.W., Yaw, B.: The hungarian method for the assignment problem. Naval Res. Logist. Quart pp. 83–97 (1955)

[31] Kwatra, V., Essa, I., Bobick, A., Kwatra, N.: Texture optimization for example-based synthesis. In: ACM SIGGRAPH 2005 Papers, pp. 795–802 (2005)

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

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

[34] Leclaire, A., Rabin, J.: A stochastic multi-layer algorithm for semi-discrete optimal transport with applications to texture synthesis and style transfer. Journal of Mathematical Imaging and Vision **63**(2), 282–308 (2021)

[35] Liu, D.C., Nocedal, J.: On the limited memory BFGS method for large scale optimization. Mathematical programming **45**(1), 503–528 (1989)

[36] Mescheder, L., Nowozin, S., Geiger, A.: Which training methods for gans do actually converge? In: International Conference on Machine Learning (ICML) (2018)

[37] Odena, A., Dumoulin, V., Olah, C.: Deconvolution and checkerboard artifacts. Distill **1**(10) (2016)

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

[39] Raad, L., Davy, A., Desolneux, A., Morel, J.M.: A survey of exemplar-based texture synthesis. Annals of Mathematical Sciences and Applications **3**(1), 89–148 (2018)

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

[41] Sajjadi, M.S., Scholkopf, B., Hirsch, M.: Enhancenet: Single image super-resolution through automated texture synthesis. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 4491–4500 (2017)

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

[43] Seguy, V., Damodaran, B.B., Flamary, R., Courty, N., Rolet, A., Blondel, M.: Large-scale optimal transport and mapping estimation. In: Proceedings of the International Conference in Learning Representations (2018)

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

[45] Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: Bengio, Y., LeCun, Y. (eds.) International Conference on Learning Representations (ICLR'15) (2015)

[46] Tartavel, G., Peyré, G., Gousseau, Y.: Wasserstein loss for image synthesis and restoration. SIAM Journal on Imaging Sciences **9**(4), 1726–1755 (2016)

[47] Thiry, L., Arbel, M., Belilovsky, E., Oyallon, E.: The unreasonable effectiveness of patches in deep convolutional kernels methods. In: International Conference on Learning Representation (2021)

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

[49] Vacher, J., Davila, A., Kohn, A., Coen-Cagli, R.: Texture interpolation for probing visual perception. Advances in Neural Information Processing Systems (2020)

[50] Webster, R.: Innovative non-parametric texture synthesis via patch permutations. arXiv preprint arXiv:1801.04619 (2018)

[51] Xia, G.S., Ferradans, S., Peyré, G., Aujol, J.F.: Synthesizing and mixing stationary gaussian texture models. SIAM Journal on Imaging Sciences **7**(1), 476–508 (2014)