Wasserstein Projection for texture synthesis

Thiziri Naït Saada [*†]
Raphaël Barboni [‡§]

February 11, 2021

Abstract

This paper provides a detailed explanation of a texture synthesis method that was developed in [6], inspired by the so-called Steerable Pyramid. This method is based on multi-resolution wavelet decompositions. The input image, starting from a white noise, is iteratively projected onto the statistical constraints set of the reference image in order to have a make it look *similar* to the reference image. The main contribution of the article of interest [6] is that it introduces a new projection operator that enables computations in high dimensional settings. Strictly speaking, the matching between images is not anymore considered anymore only through the matching of their histogram matching but it can involve additional statistical constraints such as second order statistics. In the following, we will first provide a short context of the problem that is scrutinized. Then, a brief introduction to Optimal Transport in the discrete setting of image distributions is given to lead to a proper definition of this new projection operator that will be used to project any image onto the statistical constraints. We will also shortly discuss the mathematical setting that is usually retained to define properly what is it for two textures to look *similar* before providing the whole details of the Steerable Pyramid. Finally, we will go into the details of the implementations that we have driven and comment our main results.

Keywords: Texture Synthesis, Statistical Method, Sliced Wasserstein distance, Steerable Pyramid, Wavelet decomposition, Multi-resolution analysis, Histogram matching, Optimal Transport

1 Introduction

1.1 Problem studied

Several approaches for texture synthesis have been developed in the last years. These approaches can be decomposed into three classes : neighborhood-based methods, that reorganize patches from the reference image to reproduce a similar one, statistical methods that tend to capture statistical descriptors of the reference image that will become statistical constraints for the image to be synthesized, and example methods that find and copy pixels with the most similar local neighborhood as the synthetic texture in scan-line order, based on a Markov Field assumption. In the following, we will consider a statistical approach thus we aim at projecting a random white noise onto the set of statistical constraints induced by the texture examplars we were given. This objective is the result of a highly topical discussion in cognitive sciences to define mathematically when two images (that are two realizations of a random process) can be considered to look *similar*. In some way, Julesz ended it in 1962 with his conjecture that we developed further, by considering a set of statistical constraints as the key point.

[*]École Normale Supérieure Paris-Saclay - Télécom Paris. Email: thiziri.naitsaada@telecom-paris.fr.
[†]Master 2 MVA.
[‡]École Normale Supérieure Paris-Saclay - ENS Ulm. Email: raphael.barboni@ens.fr.
[§]Master 2 MVA.

This report is written to the attention of Professor Yann Gousseau, in the context of an academic project, related to the Numerical Images course.

1.2 Related works

The use for texture synthesis of the so-called Steerable Pyramid was introduced in [2]. The idea is to build a self-inverting pyramid of one reference texture image, for which a multi-scale wavelet decomposition is performed through multiple oriented filters. This is motivated by the wavelet analysis theory, stating the wavelet functions form an orthonormal basis of particularly regular functions (whose energy can be integrated). One can find the formalism and further details in [3].

The authors used this pyramid to decompose the reference image, into a serie of low-resolution sub-images. Then, starting from a random white noise, the idea to generate new texture samples is to project this noise onto the statistical constraints set of the reference image, which is done through histogram matching between all the sub-images of the reference image and all the sub-images of the white noise (see section 4). It is important to note that with such a projection, only the first statistics of the reference image is taken into account when generating new texture samples.

In [1], the authors improved this method by correcting some artifacts due to the assumption of periodicity of the reference image (when using the DFT in the pyramid) thanks to the Moisan decomposition of an image into its periodic and smooth parts. But their main contribution is that they introduced a way to treat RGB images. Indeed, performing Principal Component Analysis of the color space of the reference image enabled them to consider three decorrelated channels, that are treated as independent. Thus, the previous algorithm is performed on each channel before coming back to the original color space. Prior to that, a new projector operation is performed, based on a sliced Wasserstein distance in [6]. It lead to fast computation, even in a high dimensional setting, which was not possible through histogram matching. Therefore, it enables to take into account higher order statistics of the reference texture, which is believed to strengthen the *similarity* of the generated image.

2 Projection in Wasserstein space

In this section, we will describe a new tool that can be used to project one input image onto a set of reference images. It is an extension of histogram matching in the sense that histogram matching corresponds actually to the one dimensional case of this new projection. Let us denote by $X = (X_i)_{i\in I}$ and $Y = (Y_i)_{i\in I}$, with $|I| = N$, some discrete density distributions in $\mathbb{R}^d$.

2.1 Wasserstein distance

Definition 2.1. *The 2-Wasserstein distance between two point clouds X and Y is defined as :*

$$\begin{aligned}\mathcal{W}_2^2(X,Y) &:= \min_{\sigma\in\Sigma_N} W_\sigma(X,Y) = \min_{\sigma\in\Sigma_N} \sum_{i,j\in I} ||X_i - Y_{\sigma(i)}||^2 \\ &= \min_{P\in\mathcal{P}_N} \sum_{i,j\in I} P_{i,j}||X_i - Y_j||^2\end{aligned}$$

where $\mathcal{P}_N$ denotes the set of bistochastic matrices of order N.

This computation results in finding an optimal ordering, which is easily done in the 1D case with fast sorting algorithms $\mathcal{O}(N\log N)$. In the case of images, the dimension d can be large. So, an alternative is to compute 1D Wasserstein distances of projected point clouds. This way, one could use fast sorting algorithms to compute this approximate Wasserstein distance. It is called the sliced Wasserstein distance.

Definition 2.2. *The Sliced 2-Wasserstein distance between two point clouds X and Y is defined as :*

$$\begin{aligned}\tilde{\mathcal{W}}_2^2(X,Y) &:= \int_{\theta\in\Theta} \mathcal{W}(X_\theta, Y_\theta)^2 d\theta \qquad \text{where } X_\theta = \{< X_i, \theta >\}_{i\in I} \subset \mathbb{R}^N \\ &= \int_{\theta\in\Theta} \min_{\sigma_\theta\in\Sigma_N} \sum_{i,j\in I} |< X_i - Y_{\sigma_\theta(i)}, \theta >|^2 d\theta\end{aligned}$$

where Θ denotes the unit sphere of $\mathbb{R}^d$.

2.2 Wasserstein barycenter

Definition 2.3. *The Wasserstein Barycenter between a set of point clouds $(Y^j)_{j\in J}$ is defined as :*

$$Bar(\rho_j, Y^j)_{j\in J} \in \arg\min_X \mathbb{E}(X) = \arg\min_X \sum_{j\in J} \rho_j \mathcal{W}_2^2(X, Y^j)$$

$$where \sum_{j\in J} \rho_j = 1, \rho \succeq 0$$

The Wasserstein barycenter enables to mix different measures into a single one. It has many applications, one of them being the synthesis of textures as we will see in the next sections. For now, there is no closed form known of this barycenter, except in the special case of gaussian distributions. Carlier and Aguch derived an analysis of the Wasserstein Barycenter through its dual formulation in the case of continuous distributions. In this paper, an alternative is proposed : to compute the Sliced Wasserstein Barycenter. Instead of considering the Wasserstein distance in the computation, one uses the Sliced Wasserstein distance, which is a 1D approximation of it using projections.

Definition 2.4. *The Sliced Wasserstein Barycenter between a set of point clouds $(Y^j)_{j\in J}$ is defined as :*

$$\begin{aligned}\tilde{Bar}(\rho_j, Y^j)_{j\in J} \in \arg\min_X \tilde{\mathbb{E}}(X) &= \arg\min_X \sum_{j\in J} \rho_j \tilde{\mathcal{W}}_2^2(X, Y^j) \\ &= \arg\min_X \sum_{j\in J} \rho_j \int_{\theta\in\Theta} \mathcal{W}(X_\theta, Y_\theta^j)^2 d\theta \\ &= \arg\min_X \int_{\theta\in\Theta} \sum_{j\in J} \rho_j \mathcal{W}(X_\theta, Y_\theta^j)^2 d\theta\end{aligned}$$

$$where \sum_{j\in J} \rho_j = 1, \quad \rho \succeq 0$$

To minimize this integral, the authors propose to use stochastic gradient descent algorithm. The convergence to a local minimum is ensured through the choice of the step size's decay along the iterations. As the result $X^{(\infty)}$ is dependent on the initialization $X^{(0)}$, the algorithm is proceeded M times. The final solution is the one that minimizes all the results obtained among $(X_1^{(\infty)}, X_2^{(\infty)}, ..., X_M^{(\infty)})$.

2.3 Wasserstein Projection

Once the Wasserstein distance is computed, many applications require to know the transport plan that has lead to such minimal distance. It is called the optimal transport plan or optimal permutation. Let us denote σ_* such a permutation for two point clouds X and Y :

$$\sigma_* = \arg\min_{\sigma\in\Sigma_N} \mathcal{W}_\sigma^2(X, Y)$$

It might be interesting in some applications to look at the point cloud X_{σ_*} on which we have applied the optimal permutation to minimize the Wasserstein distance.

Definition 2.5. *The Wasserstein Projection of a point cloud X onto another point cloud Y is defined as :*

$$X_{\sigma_*} = Proj_{[Y]}(X)$$

where $Proj_{[Y]}$ denotes the orthogonal projection onto the space $[Y]$.

Remark 2.1. *In the 1D case, the optimal permutation corresponds exactly to what is done when performing histogram matching between two images. This explains why related works have always limited themselves to histogram matching when a projection between two images was needed. This is especially the case in each iteration of the Steerable Pyramid as we will see.*

Note the intractable computation of the optimal permutation minimizing the standard Wasserstein distance, compared to the fast one when solving this problem in 1D. It explains why previous works have until now limited their studies to optimal 1D histograms matching. One key contribution of this paper is to open up new avenues of solving this problem even in multidimensional cases by using the Sliced Wasserstein distance instead of the standard one. Indeed, we have seen it comes with a lower computational cost. Computing this approximation with stochastic gradient descent ensures convergence into a local minimum only $X^{(\infty)}$. Therefore, its associated Sliced Wasserstein is not necessarily the global minimum one, and thus, the projection leading to such a point cloud is not guaranteed to be orthogonal. Note in practice, both projections are close.

Definition 2.6. *The Sliced Wasserstein Projection of a point cloud X onto another point cloud Y is defined as :*

$$X^{\infty} = \tilde{Proj}_{[Y]}(X)$$

where $\tilde{Proj}_{[Y]}$ denotes the non orthogonal projection onto the space $[Y]$.

The aim of this paper is to present a method of texture mixing. We recall the principle of texture mixing which is to generate a new texture based on a collection of examples.

3 Mathematical insights of texture modeling

3.1 Julesz conjecture

In a manner that is consistent with numerical implementations and following the definition of [5] we will take the following definition of textures :

Definition 3.1. *A texture is a real two dimensional homogeneous random field X taking values over the discrete lattice $\mathbb{Z}^2$*

Given a reference texture, that is a realization of the random field X, the work presented in [6] aims at developing numerical methods to synthesize new textures that look like the first one but are fundamentally different. That said, questions arise :

- ◇ What does "look like" mean for a texture ? One needs a referent to tell apart successes from fails. Since most of applications are human related, one can suppose that this referent is an human
- ◇ What does it mean for human that two textures look like one another ? Is their a fundamental (possibly numerical) criterion that allows to classify textures in the sense of a generic human.

The visual neuroscientist Béla Julesz interested itself for this second question for formulated his conjecture as follows :

Theorem 3.1 (Julesz conjecture (1962))**.** *There exists a set of constraint functions $\{\phi_k\}_{1\leq k\leq N_c}$ such that for any two realizations of two random fields that are equal in expectations over these set of functions are visually indistinguishable for the human brain. In mathematical terms :*

$$\mathbb{E}\phi_k(X) = \mathbb{E}\phi_k(Y)\ \forall k \ \Rightarrow\ \textit{samples from } X \textit{ and } Y \textit{ are indistinguishable} \tag{1}$$

This in particular implies that the visual stimulus provoked by a texture can be completely determined by a finite number of statistical features.

Note that Julesz himself disprove this conjecture for 2nd order (variance) and 3rd order (skewness) statistics, exhibiting counter-examples. Thus one can only hope to characterize textures by their moments up to at least the 4th order (kurtosis).

3.2 Statistical projection

Assuming the Julesz conjecture holds for a set of constraints functions $\{\phi_k\}_{1\leq k\leq N_c}$ and given a reference texture realization X, texture modeling aims at sampling from a random field Y that fulfill condition 1. Since many different random fields may satisfy 1, two factors are to be taken into account :

- ◇ **Expressiveness** : the method should find an appropriate solution Y for any realization of the constraints,
- ◇ **Generalization** : among the set of solutions, the method should not pick a random field who is only supported on a small set around X. A criterion is generally to find the solution that has maximal entropy.

Of course in practice statistical features can only be computed empirically on the one reference X and one doesn't aim at finding an admissible solution but only at sampling from it, producing realizations of the numerically modeled texture. In an attempt to fulfill the generalization criteria, one approach consists in sampling from an initial random field with sufficient entropy (e.g. gaussian) and perform successive *statistical projections* onto the set of admissible solutions. Theoretical framework is described in [5] then used in [2][6] :

1. Consider a set of constraints functions $\{\phi_k : \Omega \to \mathbb{R}^{n_k \times d_k}\}_k$ where Ω is the domain of image X and n_k, d_k are the number of features and their dimensions for each constraint.
2. Suppose furthermore, that their is a way to inverse these projections. That is given a set of constraints realizations $\{\tilde{c}_k\}_k$, there exists a function F whose outputs are images such that :
$$\phi_k(F(\{\tilde{c}_k\}_k)) = \tilde{c}_k \ \forall 0 \leq k \leq N_c$$
3. Let $c_k = \phi_k(X_k)$ be the realization of these constraints.
4. Let Y_0 be an initial solution. For $1 \leq k \leq N_c$ store :
$$\tilde{c}_k = P_k(\phi_k(Y_0))$$
where P_k is a statistical projection onto c_k (e.g. histogram matching if $d_k = 1$ or SW projection if $d_k > 1$)
5. Compute $Y_1 = F(\{\tilde{c}_k\}_k)$. If needed restart at point 4).

Next section is devoted to find the kind of constraints that are suitable in a texture synthesis framework.

4 Steerable pyramid decomposition

Multiscale decomposition is commonly used in image processing in order to compute translation invariant transformation at different scales. Its use being originally motivated by the structure of the neurons dedicated to vision in the human brain [4], its recursive structure also allows for computational efficiency. At each scale step, linear filter are applied to the image which is then subsampled. Reconstruction of the image then follows from recursive linear inversion of the filters, summation and upsampling. The whole process is referred as *pyramid* decomposition.

Classical approach consists in using a wavelet basis decomposition, taking advantage from fast algorithm as exposed in [3] to perform state of the art results in various tasks such as image compression (JPEG-2000 standard). However, in the case of image processing and texture synthesis, wavelet basis decomposition is known to produce aliasing artifact due to the presence of high frequency coefficients that are not filtered. Thus [7] proposed to use an other set of filters which most interesting properties are to be **aliasing-free** and **self-inverting** (it is a *tight-frame* as defined in [8]). The obtained filter basis also has the property to be *steerable* in the sense that :

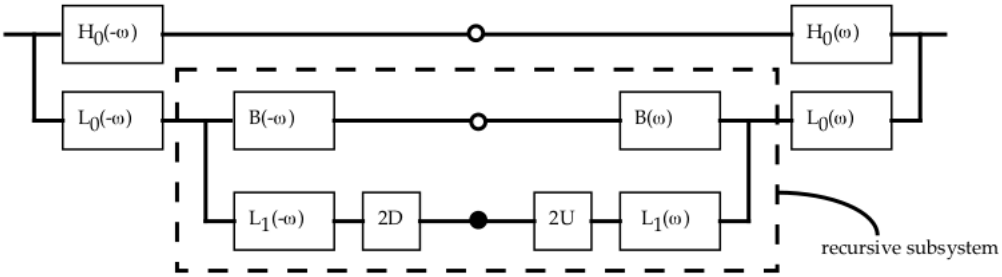

Figure 1: Steerable-pyramid decomposition process

- ◇ each filters are rotated copies of each other
- ◇ a copy of the filter at any orientation may be computed as a linear combination of the basis filters

To sum-up these ideas we can interpret the steerability property as ensuring the "expressiveness" of the filter decomposition. The obtained steerable-pyramid decomposition is illustrated by figure 1. Note that before performing the recursive part itself, the image is separated between a high-frequency and a low-frequency part thanks to filters H_0 and L_0.

Constraints on the used set of filters are expressed as follows :

- ◇ Bandlimiting, in order to prevent aliasing in the subsampling operation :

$$L_1(\omega) = 0, \ \forall \ |\omega| > \pi/2$$

- ◇ Self-inversion at the top of the pyramid :

$$|H_0(\omega)|^2 + |L_0(\omega)|^2 \left[|L_1(\omega)|^2 + |B(\omega)|^2\right] = 1$$

- ◇ Recursive self-inversion :

$$|L_1(\omega/2)|^2 = |L_1(\omega/2)|^2 \left[|L_1(\omega)|^2 + |B(\omega)|^2\right]$$

In order to fulfill these constraints, the filters proposed by [5] are expressed in polar coordinates in the Fourier domain as :

$$L_1(r,\theta) = \begin{cases} 2cos(\frac{\pi}{2}\log_2(\frac{4r}{\pi})) & , \frac{\pi}{4} < r < \frac{\pi}{2} \\ 2 & , \ r \leq \frac{\pi}{4} \\ 0 & , \ r \geq \frac{\pi}{2} \end{cases} \tag{2}$$

for the low-pass filter. The directional derivative filters are defined as $B_k(r,\theta) = H(r)G_k(\theta)$ with :

$$H(r) = \begin{cases} cos(\frac{\pi}{2}\log_2(\frac{2r}{\pi})) & , \frac{\pi}{4} < r < \frac{\pi}{2} \\ 0 & , \ r \leq \frac{\pi}{4} \\ 1 & , \ r \geq \frac{\pi}{2} \end{cases} \tag{3}$$

$$G_k(\theta) = \begin{cases} \alpha_k \cos(\pi - \frac{k\pi}{K})^{K-1} & , \ |\pi - \frac{k\pi}{K}| < \frac{\pi}{2} \\ 0 & , \ \text{otherwise} \end{cases} \tag{4}$$

for $0 \leq k \leq K-1$ directions with the normalizing constant α_K. The initial low-pass and high-pass filters applied at the top of the pyramid are respectively :

$$L_0(r,\theta) = \frac{1}{2}L_1(\frac{r}{2},\theta) \tag{5}$$

$$H_0(r,\theta) = H(\frac{r}{2},\theta) \tag{6}$$

Note that this decomposition basis indeed posses the steerable property as it consists of $(K-1)$-order directional derivatives in K distinct directions. On the other hand aliasing artifacts are prevented due to the fact that the support of L_1 respects the *Nyquist sampling criterion*, which is not the case for generic wavelet decomposition.

5 Numerical implementation and results

We present here a few numerical results of texture modeling that we obtained implementing the method described above and presented in [2][6] and [1]. We used here already implemented routines available with the packages `PyWavelet` and `Pyrtools`. Note that we only try to perform the generative modeling of one texture (barycenter of 1 reference texture), however, generalization to mixing (barycenter of several textures) is straightforward.

We assume in the following that we are provided with `HistogramMatching` and `SWprojection` routines that perform histogram matching and Sliced-Wasserstein projection of section 2 onto 2-dimensional arrays (images).

5.1 Gray-scale images

The case of gray-scale images is straightforward considering the theory of the previous sections. Image is decomposed using either multi-scale wavelet or steerable-pyramid decomposition. Given the fact that the dimension of the obtained feature is 1 (scalar image), statistical projection is performed using histogram matching. Detailed procedure is describe in algorithm 5.1 and numerical results are presented in figure 2

Algorithm 1 Grayscale texture synthesis

- Choose reference texture image X of size (N, N), order and height K, L of the decomposition, number of iterations Nit, wavelet basis w
- Define `Decomposition`(.) as `SteerablePyramid`$(., K, L, axis = (0, 1))$
- or `WaveletDecomposition`$(., K, L, w, axis = (0, 1))$
- $\mathbf{X} \leftarrow$ `Decomposition`(X)
- $Y \leftarrow Gaussian((N, N))$
- **for** $0 \leq it < Nit$ **do**
  - $\mathbf{Y} \leftarrow$ `Decomposition`(Y)
  - **for** $0 \leq l < L$, $0 \leq k < K$ **do**
    - $\mathbf{Y}[l, k] \leftarrow$ `HistogramMatching`$(\mathbf{Y}[l, k], \mathbf{X}[k, l])$
  - $Y \leftarrow$ `DecompositionRec`$(\mathbf{Y})$
  - $Y \leftarrow$ `HistogramMatching`(Y, X)
- Output: Y

Note the last statistical projection step at the end of each iteration. Described in [6], this corresponds to histogram equalization and ensures that the global grayscale aspect of the output image is the same as the input.

One sees that a steerable-pyramid decomposition seems to lead to better results. On the other hand, we performed at each iteration decomposition on several wavelet basis, therefore augmenting the number of constraints whereas a fair comparison using only one wavelet basis leads to far worse results. In a general manner, one sees that the algorithm doesn't manage to produce sharp edges (e.g. for 'pop-corn' or 'banana peel' image).

5.2 Color images

The case of color images is more tricky as dimension of the features is 3 and channels or strongly correlated as pointed out in [1] and illustrated in figure 3 and 4 Therefore, performing direct SW

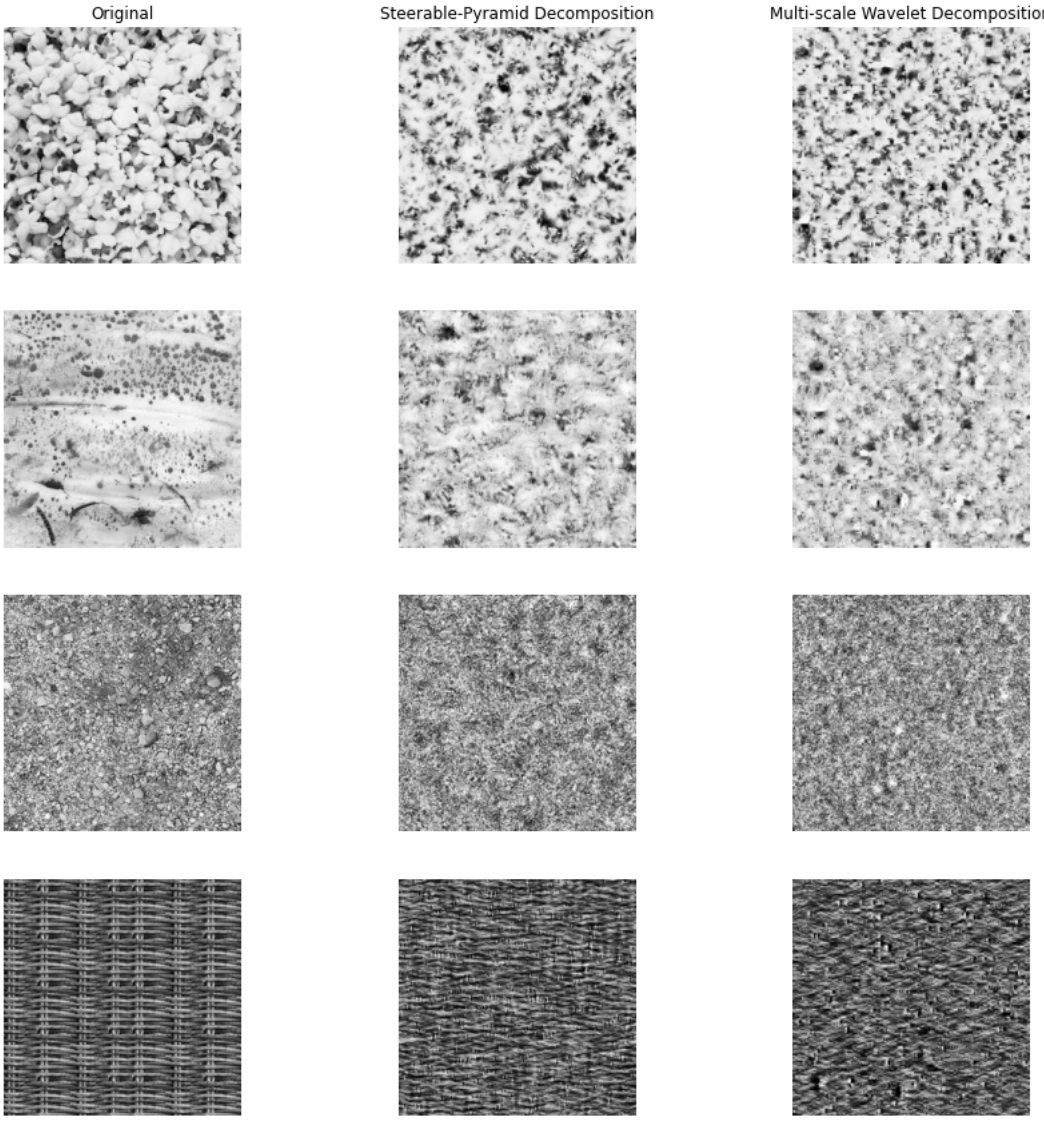

Figure 2: Reference grayscale textures (left) and images generated by algorithm 5.1 using steerable-pyramid decomposition (center) or multi-scale wavelet decomposition (right). Parameters : $L = 4$, $K = 4$, $Nit = 10$, images are size 256×256. Used wavelets are *symlet3*, *daubechies3* and *biorthogonal3.1* (see `PyWavelet` package documentation) performing decompositions and projections for each wavelet at each iterations.

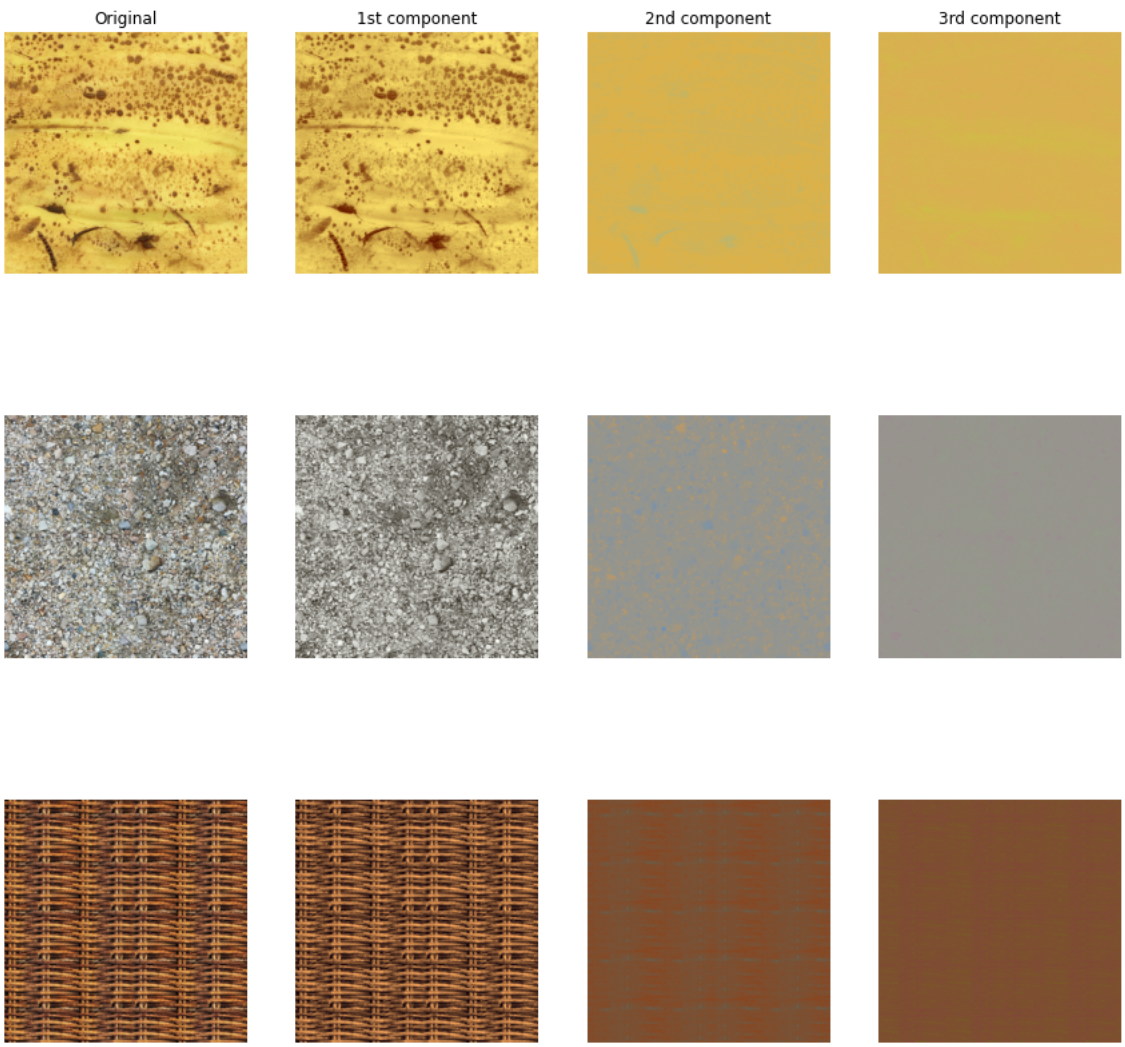

Figure 3: Principal components channel of several textures. One sees that almost all the information about the image is contained only in the first component, therefore justifying the use of histogram matching instead of SW projection

projection of totally uncorrelated features (initial image is gaussian random field) onto the original features of the image only led to poor results.

The idea proposed by [1] to tackle this issue is to start by performing Principal Component Analysis (PCA) onto the color channels, de-correlating them and re-normalizing them. Thus, gaussian random field is already a better statistical approximation of the obtained image at start. Multi-scale wavelet or steerable-pyramid decomposition is then performed on each of the channels independently, as for gray-scale images.

For the projection step, several possibilities arise. While the whole idea behind [6] is perform a 3D statistical projection using a SW projection, [1] only proposes to perform histogram matching onto the three channels independently, therefore treating each channel as independent grayscale images during the whole process and not exploiting at any moment the 3D structure of the data. Surprisingly, that last approach better results, actually reproducing features from the original texture while the first one did not perform better than modeling a texture by a gaussian random field in the PCA space (which producing a white noise image but at the same hue of the original texture). Corresponding numerical method are exposed in algorithm 5.2 and visual comparison of the results

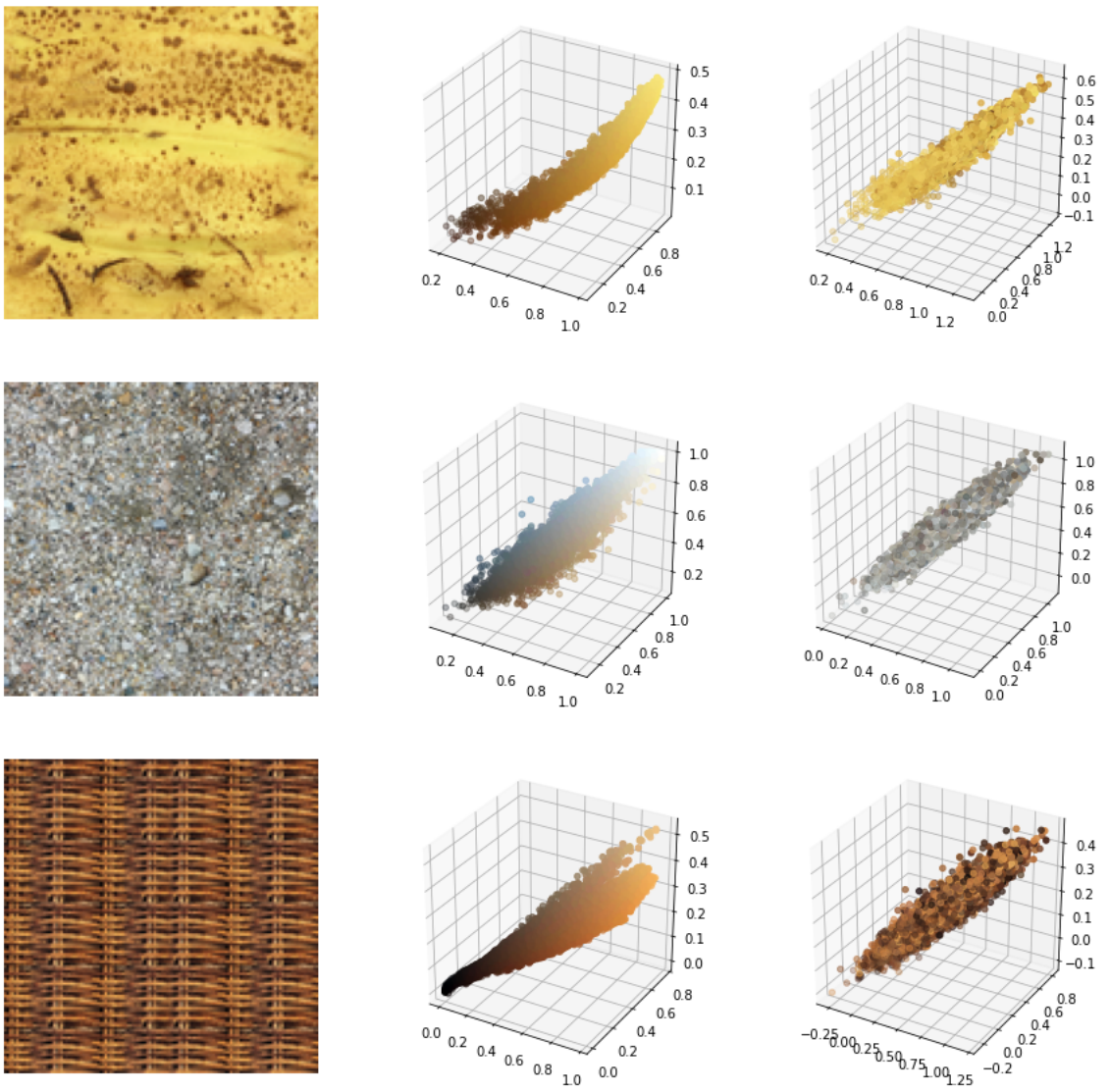

Figure 4: Original textures (left), their pixel distributions (center) and the pixel distributions of the generated texture (right), whose pixels colors are the same as the colors of the corresponding pixels in the original texture. One sees that color distributions are degenerate. Also the generated distributions are close to the original ones but colors are permuted, showing that the algorithm succeed in generating original textures.

are showed in figure 5

Algorithm 2 Color texture synthesis

- Choose reference texture image X of size $(N, N, 3)$, order and height K, L of the decomposition, number of iterations Nit, wavelet basis w
- Define `Decomposition`(.) as `SteerablePyramid`$(., K, L, axis = (0, 1))$
or `WaveletDecomposition`$(., K, L, w, axis = (0, 1))$
- Define `Projection`(.,.) as `HistogramMatching`$(., ., axis = (0, 1))$ or `SWprojection`(.,.)

$M \leftarrow$ `Mean`$(X, axis = (0, 1))$
$X' \leftarrow$ `PCAtransform`$(X - M)$, $Var \leftarrow PCAvariance(X - M)$
$\mathbf{X} \leftarrow$ `Decomposition`(X')
$Y \leftarrow Gaussian((N, N, 3)) *$ `Sqrt`(Var)
for $0 \leq it < Nit$ **do**
  $\mathbf{Y} \leftarrow$ `Decomposition`(Y)
  for $0 \leq l < L$, $0 \leq k < K$ **do**
    $\mathbf{Y}[l, k] \leftarrow$ `Projection`$(\mathbf{Y}[l, k], \mathbf{X}[k, l])$
  $Y \leftarrow$ `DecompositionRec`$(\mathbf{Y})$
  $Y \leftarrow$ `SWprojection`(Y, X')
$Y \leftarrow$ `PCAinverse`$(Y) + M$
Output: Y

We can not provide a rigorous explanation of this phenomena : SW projection allows to recover histogram matching for 1D data, therefore, it should perform at least as good in 3D as histogram matching on each of the channels independently. Most probable explanation is that SW projection (and moreover a stochastic version of it) only performs an approximation of a rigorous optimal transport projection, which would be the right 3D equivalent of histogram matching. Therefore it is probable that lots of statistical approximations are made during the SW projection process whereas histogram matching performs "exact" statistical projection in 1D, which is sufficient because most (if not almost all) of the information about the texture is contained only in the first PCA channel.

Remark 5.1. *To test this hypothesis rigorously, one should modify algorithm 5.2 to replace* `SWprojection` *by a rigorous multidimensional optimal transport assignment obtain by linear programming. However, computing such an assignment would be very costly given the size of the data (typically of dimension* $128 \times 128 \times 3$*).*

5.3 Discussion

Throughout the experiments the reader saw that, at best, our implementations managed to generate "satisfying" results without achieving the same performances as [6] and [1]. Our method requires to set some parameters whose importance should be discussed on :

- ◇ **Number of orientations** K **:** we often set it to $K = 4$. While using less orientations could led to a lack of expressiveness, using more orientations does not lead to any significant improvement. This can be explained by the fact that features of the images are often visually well enough by their response to 4 different orientations.
- ◇ **Number of scales** L **:** same as for K. Reducing L to lower than 3 or 4 could led to lack of expressiveness but using more decomposition scales does not allow to generate better texture : important features of a textures are most of the time local features.
- ◇ **Number of iterations** Nit **:** this is probably the most interesting parameter to study. The method usually produces good results after only a small number of iterations (e.g. $Nit = 5$) and the image often don't evolve much after. Therefore, the question already asked by [1] is to know whether the iterative decomposition-projection-recomposition process described in section 3 converges towards a fix point. If it does converge anyway, this is probably not towards

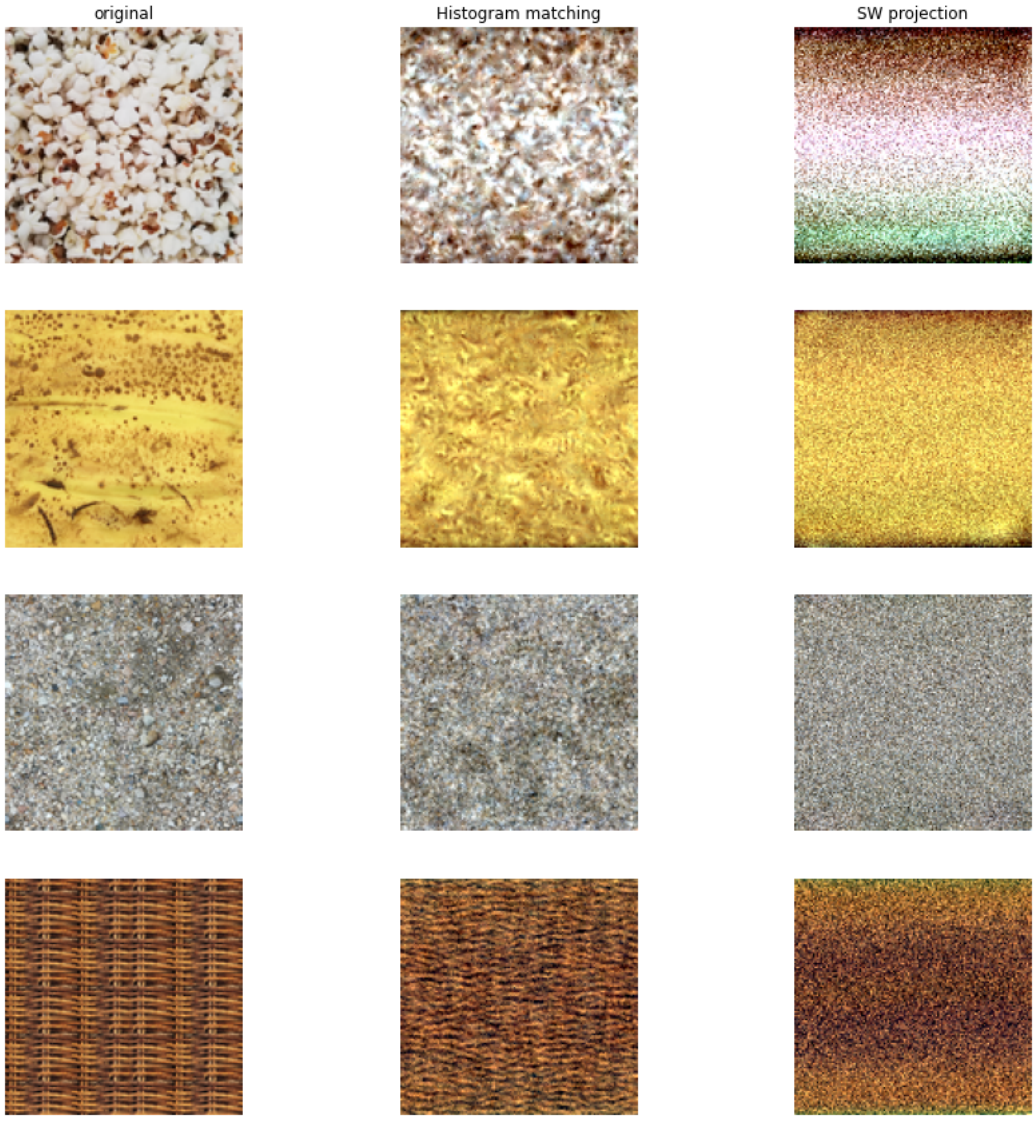

Figure 5: Reference color textures (left) and images generated by algorithm 5.2 using steerable-pyramid decomposition and histogram matching on each channel (center) or Sliced-Wasserstein projection (right). Parameters : $L = 4$, $K = 4$, $Nit = 5$, , images are size 128×128.

a minimizer of the projection distance to the reference image. Indeed, computing the projection distance at each iteration we often observed that this functional attained a minimum after a small number of iterations (typically $Nit = 5$ or 10), before increasing.

Finally, we also merely tried to modify our algorithm in order to include projections onto 2nd order statistical features, as advised in [6]. However it never led to significant improvements of the results.

6 Conclusion

Finally, we have covered some approaches that are mainly mathematically oriented and interpretable, based on the wavelet analysis theory. However, when recalling the structure of such a Steerable Pyramid and the arise of hundreds or thousands of filter parameters that are to be optimized when increasing the scale, one can find some substantial similarities with neural networks. This succession of convolutions through parametric filters all along the scale axis needs to bring both approaches closer. On the other hand, another alternative method was developed in [9]. Instead of wavelets, the decomposition is performed among a dictionary of sub-images that are learnt from the reference image. Imposing sparsity constraints in addition to the statistical constraints, an approximate solution of this problem can be obtained with alternate descent. It looks very similar to the dictionary learning that is performed on time series when trying to identify a pattern within the time serie. As a further investigation, it could be interesting to focus on the right choice of the statistics order that would yield better results and to try to explain why.

References

[1] T. Briand, J. Vacher, B. Galerne, and J. Rabin, *The Heeger & Bergen Pyramid Based Texture Synthesis Algorithm*, Image Processing On Line, 4 (2014), pp. 276–299.

[2] D. J. Heeger and J. R. Bergen, *Pyramid-based texture analysis/synthesis*, in Proceedings of the 22nd annual conference on Computer graphics and interactive techniques - SIGGRAPH '95, Not Known, 1995, ACM Press, pp. 229–238.

[3] S. Mallat, *A theory for multiresolution signal decomposition: the wavelet representation*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 11 (1989), pp. 674–693.

[4] D. Marr, *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*, The MIT Press, 2010.

[5] J. Portilla, *A Parametric Texture Model Based on Joint Statistics of Complex Wavelet Coefficients*, International Journal of Computer Vision, 40 (2000), pp. 49–70.

[6] J. Rabin, G. Peyré, J. Delon, and M. Bernot, *Wasserstein Barycenter and Its Application to Texture Mixing*, in Scale Space and Variational Methods in Computer Vision, A. M. Bruckstein, B. M. ter Haar Romeny, A. M. Bronstein, and M. M. Bronstein, eds., vol. 6667, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012, pp. 435–446. Series Title: Lecture Notes in Computer Science.

[7] E. Simoncelli, W. Freeman, E. Adelson, and D. Heeger, *Shiftable multiscale transforms*, IEEE Transactions on Information Theory, 38 (1992), pp. 587–607.

[8] E. P. Simoncelli and W. T. Freeman, *The steerable pyramid: a flexible architecture for multi-scale derivative computation*, in Proceedings., International Conference on Image Processing, vol. 3, 1995, pp. 444–447 vol.3.

[9] G. Tartavel, Y. Gousseau, and G. Peyré, *Variational texture synthesis with sparsity and spectrum constraints*, (2015).