Texture synthesis through level sets

Yann Gousseau

Abstract— **We present a fast method for the synthesis of texture from an input sample. The method relies on the extraction of connected parts of level sets that are then sequentially superimposed one to another, and enables to synthesize texture images of any size, tiling seamlessly. A particularity of the method is its ability to reproduce rare events in textures.**

Keywords— **Texture synthesis, level sets, occlusion, shape extraction.**

Texture synthesis is a challenging problem, let it be for the understanding of our visual perception, for practical applications or from the mathematical point of vue. Generic methods usually consider textures as samples from a probability distribution on images. The best known one is Markov Random Field modeling. A texture is associated with a random field (in the discrete case, each pixel has a random gray level value), characterized by neighborhood interactions, the field being usually given a Gibbs formalism, see e.g. [6], [21], [22]. These model usually suffer from the difficulty to handle long distance dependences, from the low order of modeled interactions, as well as from the preference given to the pixel scale. Another class of methods deals with statistical properties of multiscale decompositions of images. The paper of Heeger and Bergen ([11]) made a breakthrough in the field, and proposed to synthesize a texture by adjusting its histogram responses to a set of oriented filters implementing a wavelet decomposition. This method was significantly improved by capturing statistical dependences within and across scales, [17], [5]. This enables to represent large scale elements and edges of the textures, and a very large range of textures may be successfully synthesized this way. However the method may fail in the case of abrupt edges, for which it is necessary to take into account long range statistics across wavelet scales ([17]) or may lead to almost periodicity when synthesizing samples larger than the original. Another approach, developed by Mumford, Wu, and Zhu ([23]), merges the filter bank histogram responses method with a Gibbs formalism.

In a different direction, a very efficient method has been recentley proposed, that consider the texture as the realization of a Markov field, but samples directly from an input image, [7]. Putting it simply, the input image itself replaces an explicit model. One drawback of this method is its computational complexity, and recent papers tackled this problem quite successfully, see [19]. Actually, this article from Efros and Leung initiated several related works on synthesis methods based on sampling from an input image: [19], [4], [8]. The method we will now present also consists in sampling directly from an input texture, but deals with more complex primitives.

One of the main drawbacks of most synthesis methods is the difficulty to maintain the presence of rare events in images, which can be essential ingredients of the realistic aspect of the synthesized samples. In fact, most textures are intermediate objects between homogeneous fields and complicated scenes, so that methods based on the analysis-synthesis of average behaviors fail to reproduce some features of textures. We attack this problem in the most straightforward way, by dealing directly with large parts of the image, and by mixing them to obtained the synthetic image. In the same spirit, several methods rely on some visual masking phenomenon to synthesize a new texture by pasting together patches of an input image, but rely either on rectangular blocks ([10], [8]), or on some user defined patches ([16]). Our method was, to our knowledge, among the first in this direction ([9]), and automatically extract components having a visual significance. More precisely, we first extract connected components of level sets, and then superimpose them at randomly chosen locations in the image to be synthesized. This way, we developed a fast algorithm that permits to reproduce quite a large range of textures, including the one made by occlusion (flowering tree, forest ground, stones on a beach...), but also textures which are mainly homogeneous, but present rare events more or less evenly distributed. The method renews the so-called structural approach to texture synthesis ([1], [18]) which tries both to detect salient features of textures and to organize them, that have been left aside because they did not focus on the right basic elements of texture, looking for shapes far too simple to have realistic results. We also believe that our experiments underline the pertinence of level set representation for image analysis and processing ([13], [3], [15], [9]). This paper is organized as follows: in Section I we define the notion of shape and explain how we extract them, and in Section II we give the general synthesis algorithm, display some successful examples, give refinements and examples of failures, and eventually show how our algorithm permits to create realistic abstract images.

I. Level sets and "shapes"

In this section we define the basic elements to be used for the synthesis, the "shapes". Those are connected components of level sets whose holes have been removed. We write I for a digital gray level image defined on a discrete domain D (for example a 512×512 square lattice) whose values belong, say, to the set $G = [1, .., 256]$. For each k in G, define the two (respectively inferior and superior) k-level sets of I:

$$\chi_k = \{i \in D \text{ such that } I(i) \geq k\},$$

$$\chi^k = \{i \in D \text{ such that } I(i) \leq k\}.$$

We then define, for each k, the sets $s_{j,k}$, j varying in a k-dependent subset of the integers, consisting of the con-

Y. Gousseau is with the Département de Traitement du Signal et des Images, ENST, Paris, France. E-mail: gousseau@tsi.enst.fr

nected components of the previously defined level sets. That is to say, each $s_{j,k}$ is a group of connected pixels, all members of χ_k or all members of χ^k. Eventually, for each $s_{i,k}$, we define $S_{i,k}$ to be the set obtained from $s_{i,k}$ by "filling the holes". A hole in a set is defined as a connected component of its complementary set that does not meet the border, and an $S_{i,k}$ consists of all pixels from $s_{i,k}$, together with pixels which belong to one of its holes. In Figure 1, we give an example of χ, s, and S coming from a very simple binary image. We call any of the $S_{i,k}$, thus a group of pixels, a "shape" of the image. In brief, those consist of the connected components of the level sets together with their holes.

A very efficient extraction and organization of such shapes of an image, the fast level set transform (FLST), has been developed by Monasse and Guichard ([15]). The shapes are extracted and organized in a tree, using the fact that two shapes are either disjoint or included one in another. The descendants of a shape are the shapes included in it, the parent of a shape is the smallest shape that contains it, and the root of the tree is the complete image domain. A remarkable fact about this tree is that it is invariant by local contrast changes, whereas the level sets are only invariant by global contrast change. The important feature of a pixel is whether it is brighter or darker than its neighbors. This tree permits to reconstruct the image, and has been proved very useful for such tasks as simplification, comparison or registration of images, [14].

The "shapes" will be the basic material for our synthesis, as explained in the next section, and the FLST will be applied once to the image, providing us with a large dictionary of shapes to be used for the synthesis.

II. Texture synthesis

A. The general framework

The synthesis is achieved by randomly choosing shapes from the input image, and by sequentially dropping them on the output image. A shape that is dropped after another one will occlude it in case they fall at the same place. This procedure echoes the mathematical "dead leaves model" of G. Matheron, [12], as well as the spot noise method, [20], addition being replaced by occlusion.

In this section, we use the same symbol both for the image and its domain (a rectangular lattice). We start from an input image A, representing a sample of the image to be synthesized, and an output image B in which originally all pixels are of the same gray (or color) level. From A we extract all shapes, and we iterate the following procedure, starting from a uniform B:

- randomly pick a shape S from the initial image (according to some probability law P_1 to be specified);
- pick a random location x in B (according to some probability law P_2 depending on S and B);
- draw the chosen shape S in B, centered on x. That is to say, for each pixel i in S, if $i+x$ is in B, do $B(x+i) = A(i)$.

This procedure is iterated until B has been completely covered with shapes. Eventually, we perform a gray level histogram matching, that is to say that the histogram of B is matched to the one of the input texture A. Thus the synthesis relies on two probability laws P_1 and P_2, namely the choice of the shapes and the locations in the synthesized image. In the more general framework, P_1 and P_2 may depend on the iteration, for instance on previous choice of shape or location, or both if they depend on the current state of B.

The method is simply extended to color images by extracting shapes from the intensity component of A, and performing histogram equalization on this same component. Let us point out that this method permits the synthesis of samples that tile seamlessly, by drawing the shapes in B with torical limit conditions, that is to say that parts of the shapes that fall outside B are painted on the opposite side of B.

B. A first choice for the probability laws on the shapes

We choose the following version of P_1. To select a shape, first select a minimum and maximum size MIN and MAX, then pick a random location y in A (all locations being equally likely). Then pick a shape among all shapes containing y, whose area is between MIN and MAX and not meeting the border (shapes meeting the border are rejected to avoid visible horizontal and vertical lines), all these shapes being equally likely [1]. As a first choice for P_2, we considered a uniform draw among all locations in B, that is to say that each location is equally likely. In order to avoid edge effects, we pick locations in a slightly enlarged image. With those choices for P_1 and P_2, the only parameters are MAX and MIN, the maximal and minimal sizes for the shapes. Those very simple rules lead to convincing synthesis in the case of textures made by occlusion, as one may expect, but also for some microtextures. Other textures may also be mimicked, as is illustrated in the case of the wall. We display some results in Figures 2, 3, 5. Computing time for those figures were between 3 and 7 seconds on a Pentium 1 GHz. Note that once the shapes have been extracted, a new synthesis takes less than a second.

a b c d

Fig. 1. a: original image, b: χ^0 (where 0 is black), c and d: shapes from χ^0

[1] An alternative we first tried consisted in choosing the shape with maximal contrast on its boundary and containing y. Indeed, this seemed to be a better way to extract meaningful elements of the texture. However, those sets correspond to regions where level sets accumulate, so that uniform draws lead more or less to the same result.

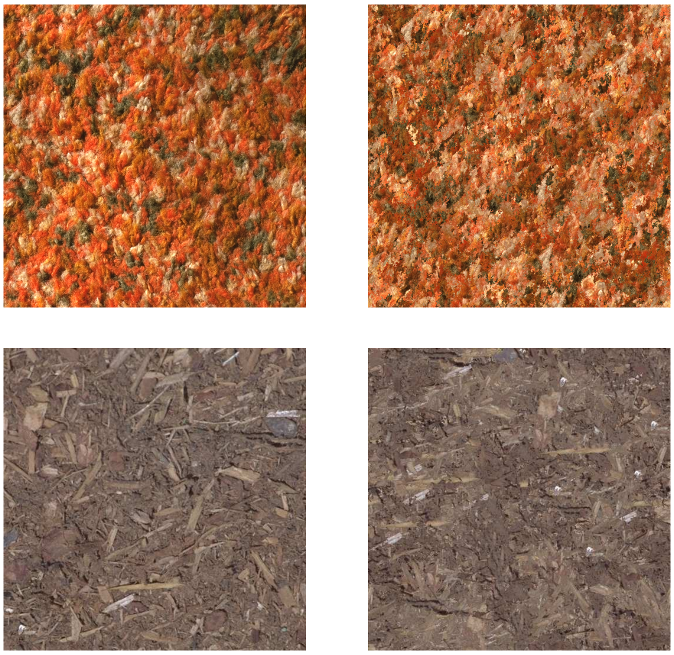

Fig. 2. left: original textures, right: synthesis.

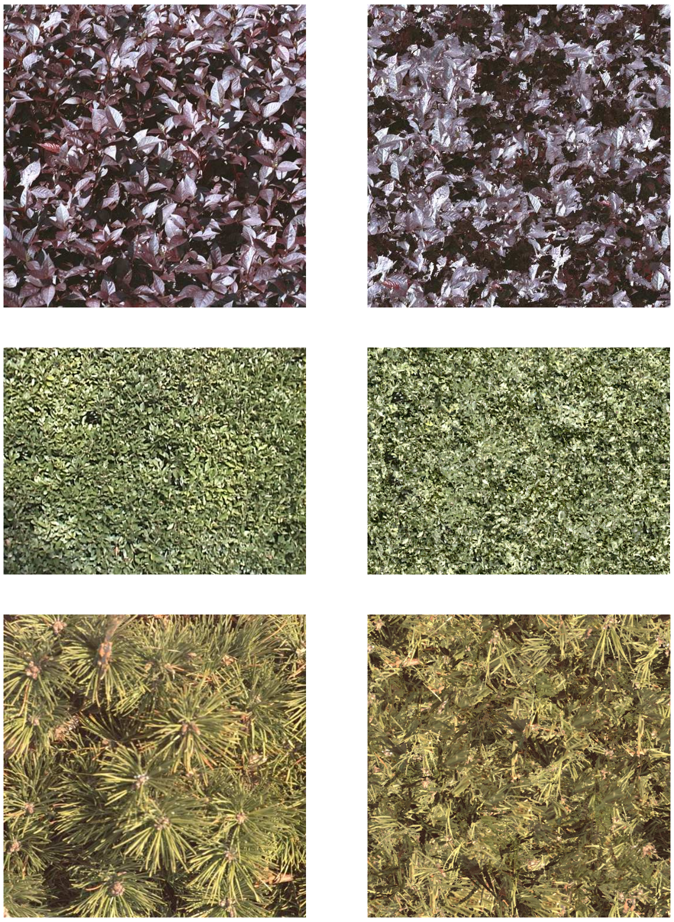

Fig. 3. left: original textures, right: synthesis. The first two experiments are particularly successful because of the occlusion structure of the original textures. The third one fails partially, because of the complicated macroscopic organization of petals

C. Refinements, failures and abstract synthesis

With the previously defined rules, we synthesize a texture B by organizing basic elements of A which correspond to low frequency features. High frequency features of the texture are reproduced within each of these elements, but relationships between them is lost. Therefore we tried to refine the probability law P_2, in order to respect other low frequencies statistics. A possibility is to choose a location for the shape in the synthesized texture that minimizes some cost function. For instance, we can minimize the difference between the contrast at the border of the shape in the original image, and in the synthesized one. Such a minimization would be far too long or impossible to achieve exactly, so that we define the following P_2:

- for each S chosen according to P_1 in A, for each point p of the complementary set of S which is neighbor of a point p' in S, compute $d = |A(p) - A(p)'|$. Define D to be the vector of those differences (the number of elements in D is the number of couples (p, p') at the border of S);
- for N choices of random location x in B, compute the new vector $\tilde{D}$, made of $\tilde{d} = |A(p') - B(p+x)|$. $\tilde{D}$ represents the contrast the shape S would have at its border if located at x in B;
- keep the random location x that minimizes the norm of $D - \tilde{D}$, and draw the shape at that location in B.

As before, this procedure is iterated. The idea behind P_2 is quite simple: the contrast at the border of the shape (before it is occluded by other ones) in B must be similar to the one the shape originally had in A. This choice of P_2 shares resemblances with the synthesis strategy adopted in [7]. This refined version of P_2 improves the results in some cases where the previous organization failed to reproduce low frequency features, even though the results are not yet perfect. In figure 4, bottom, we display an example, a piece of fabric, where the result is improved with the new version of P_2, but is still not satisfying. Probably more complicated relationships need to be respected between shapes, and this is the object of current work. We will also have to deal with the average reducing of shapes under occlusion, especially visible in the case of the wall, see Figure 4.

It was one of our first goal to develop a procedure to synthesize "abstract natural images". We call so images that do not reproduce real scenes immediately recognizable as photographs, but abstract images that respect some principles and/or statistics of such pictures. In a previous paper, [2], we explored the possibility to create a wide range of abstract images from simple geometric objects respecting some principles such as exclusion, inclusion, occlusion, self-occlusion. Here, we make use of the occlusion principle, replacing the geometrical objects by the shapes of the previous sections. In the framework of the synthesis described in Section II-C, we synthesized some abstract images, an example of which we display in Figure 7. We emphasize that we manage to create an image which is close to reproduce a realistic scene. More examples may be found in [9], and at http://www.tsi.enst.fr/~gousseau/Texture.

Fig. 4. Top row: due to occlusion, the average size of the stones is reduced. Middle row: synthesis failure, N=1. Bottom image: N=20 improves the result, but failure remains.

To conclude, we presented a new method for the synthesis of texture samples of any size from an input texture, based on the sequential superimposing of connected parts of level sets. The method is fast, enables to reproduce rare events in textures, and is especially efficient for texture made by occlusion, or for textures having extended elements whose positions are more or less independent. The method deals efficiently with sharp and intricate discontinuities that are an issue for other synthesis methods. However, in the case of textures having inter-dependent low frequency features, failures occur, and different choices for the probability law P_2 may improve the results for such textures.

Acknowledgments

We thank Jean-Michel Morel for useful advices and discussions, and Pascal Monasse for his help on the FLST.

Some of the texture images used in this section are from the VisTex database, a collection of textures images kindly provided by the Medial Lab at M.I.T., and available by anonymous ftp at whitechapel.media.mit.edu, directory pub/VisTex.

References

[1] N. Ahura and A. Rosenfeld. Mosaic models for textures. *IEEE Trans. On Pattern Analysis and Machine Intelligence*, 3(3):1–11, January 1981.

[2] L. Alvarez, Y. Gousseau, and J.-M. Morel. The size of objects in natural and artificial images. *Advances in Imaging and Electron Physics, Academic Press*, 111:167–242, 1999.

[3] L. Alvarez, F. Guichard, P.-L. Lions, and J.-M. Morel. Axioms and fundamental equations of image processing. *Archive for Rational Mechanics and Analysis*, 16(9):200–257, 1993.

[4] M. Ashikhmin. Synthesizing natural textures. In *2001 ACM Symposium on Interactive 3D Graphics*, pages 217–226, 2001.

[5] J. S. De Bonet and P. Viola. A non-parametric multiscale statistical model for natural images. *Advances in Neural Information Processing*, 10, 1997.

[6] A. Cross and A. K. Jain. Markov random field texture models. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 5(1):25–39, 1983.

[7] A. Efros and T. Leung. Texture synthesis by non-parametric sampling. In *IEEE International Conference on Computer Vision*, pages 1033–1038, Corfu, Greece, 1999.

[8] A. A. Efros and W. T. Freeman. Image quilting for texture synthesis and transfer. *Proceedings of SIGGRAPH 2001*, pages 341–346, August 2001.

[9] Y. Gousseau. *The distribution of shapes in natural images.* PhD thesis, CEREMADE, Université Paris IX, January 2000.

[10] B. Guo, H. Shum, and Y.-Q. Xu. Chaos mosaic: Fast and memory efficient texture synthesis. Technical report, Microsoft Research, 2000.

[11] D. J. Heeger and J. R. Bergen. Pyramid based texture analysis/synthesis. In *SIGGRAPH'95*, pages 229–238, 1995.

[12] G. Matheron. Modèle séquentiel de partition aléatoire. Technical report, CMM, 1968.

[13] G. Matheron. *Random Sets and Integral Geometry.* John Wiley & Sons, New York, 1975.

[14] P. Monasse. Contrast invariant image registration. In *Proc. of ICASSP*, volume 6, pages 3221–3224, 1999.

[15] P. Monasse and F. Guichard. Fast computation of a contrast-invariant image representation. *To be published in IEEE Trans. Image Proc.*, 1999.

[16] E. Praun, A. Finkelstein, and H. Hoppe. Lapped textures. In *SIGGRAPH'00*, pages 465–470, 2000.

[17] E. P. Simoncelli and J. Portilla. Texture characterisation via joint statistics of wavelet coefficient magnitudes. In *5th IEEE Int'l Conf. on Image Processing*, Chicago, IL, 1998.

[18] F. Vilrotner, R. Nevatia, and K. Price. Structural analysis of natural textures. *IEEE Trans. On Pattern Analysis and Machine Intelligence*, 8(1):76–89, 1986.

[19] L.-Y. Wei and M. Levoy. Fast texture synthesis using tree-structured vector quantization. In *SIGGRAPH'00*, 2000.

[20] J. J. Van Wijk. Spot noise: Texture synthesis for data visualisation. *Computer Graphics*, 25(4):309–317, 1991.

[21] G. Winkler. *Image Analysis, Random Fields and Dynamic Monte Carlo Methods.* Springer-Verlag, 1995.

[22] A. Zalesny and L. Van Gool. A compact model for viewpoint dependent texture synthesis. In *3D Structure from Images - SMILE 2000*, volume 2018 of *Lecture Notes in Computer Science*, pages 124–143. Springer, 2000.

[23] S. W. Zhu, Y. Wu, and D. Mumford. Filters, random fields and maximum entropy (FRAME). *Int'l Journal of Computer Vision*, 27:1–20, 1998.

Fig. 5. left: original texture, right: synthesis.

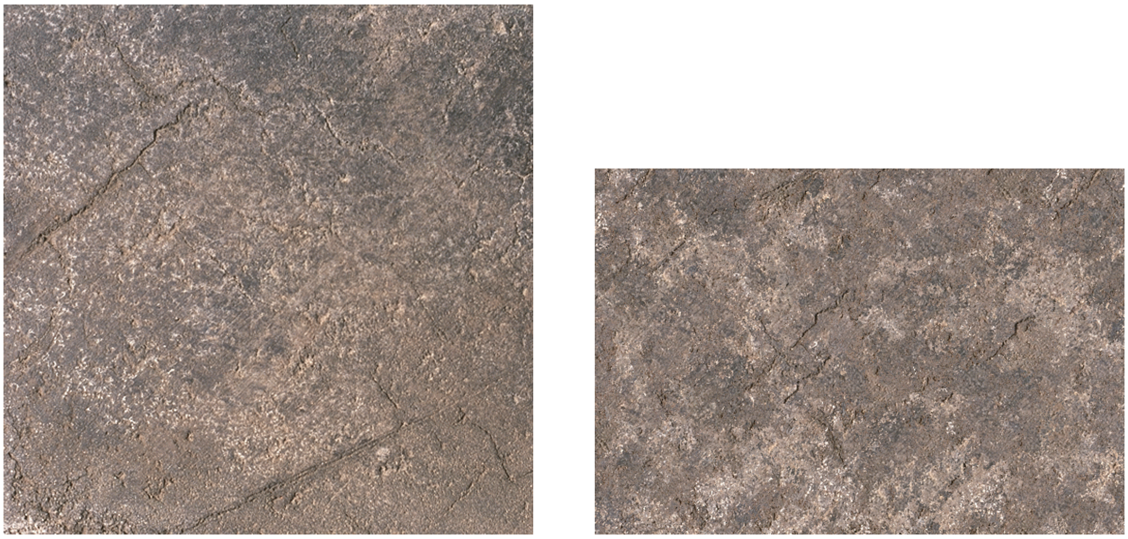

Fig. 6. left: original texture, right: synthesis.

Fig. 7. left: original picture, right: synthesis with $N = 20$. This image is no more a texture (has many single events), and we synthesize an abstract realistic version of it.