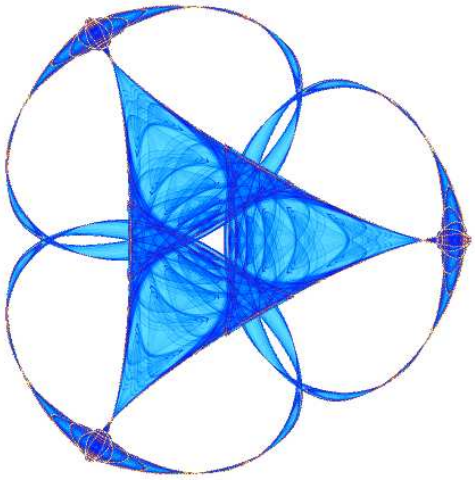
LEARNING THE STRUCTURE OF TEXTURES

By

Diego Rother
and
Guillermo Sapiro

IMA Preprint Series # 2223

(September 2008)



INSTITUTE FOR MATHEMATICS AND ITS APPLICATIONS

UNIVERSITY OF MINNESOTA
400 Lind Hall
207 Church Street S.E.
Minneapolis, Minnesota 55455–0436

Phone: 612-624-6066 Fax: 612-626-7370
URL: http://www.ima.umn.edu

Learning the Structure of Textures

Diego Rother and Guillermo Sapiro

Abstract

A framework for learning the structure of textures is introduced in this work. A Bayesian network, whose structure and parameters are learned from a small texture sample, is used to characterize the high-dimensional probability density function (PDF) that models the given texture. The underlying network learning framework is based on the combination of tools such as kernel density estimation and quasi-greedy tree search, together with optimality criteria such as maximum likelihood and cross-validation. While learning the Bayesian network, in addition to the PDF, we learn critical structural relationships among the texture pixels. As examples of the proposed framework, we present two basic applications: First, the learned PDFs are used for texture classification, through a Bayes classifier; and second, the structural knowledge is used to improve state-of-the-art texture synthesis algorithms by customizing the context to each texture.

1 Introduction and motivation

It is often assumed in the texture analysis, synthesis, and classification literature that classes of equivalent textures can be characterized by a probability density function (PDF) [1-8].[1] In this work, in order to model textures, we investigate and develop state-of-the-art density estimation techniques based on *Bayesian Networks* and *Kernel Density Estimation*. These techniques lead to learning not only the PDF, but perhaps more importantly, also the basic underlying structure of the texture, providing texture models as well as insights leading to improvements in applications such as texture synthesis and classification.

In texture learning, as well as in most problems of interest in this field, the dimensionality of this unknown density is very high, thereby, hypotheses are often added to simplify and constrain the problem. A common assumption is that of locality: a pixel conditioned on its "closest neighbors" is independent of every other pixel. When the *perceived* (by human observers) quality of the texture is the main interest, these hypotheses are often related to (human) visual perception. The goal then is not to estimate the probability density that best characterizes the given data (texture) according to some formal statistical criterion, but rather to estimate the density that is *perceived* as generating the training data (see [9] for a review of the current texture perception understanding).

To understand basic intrinsic properties of texture PDFs, we set out to estimate this density and texture structure under very weak assumptions (to be described later). Although the mathematical and computational framework here introduced can be extended to address additional constraints (and might even benefit from them, further reducing the problem dimensionality and thereby simplifying the learning task), such generality also helps to understand the power of commonly used hypothesis and the limitations of not considering them. In addition, such generality makes the overall framework here introduced independent of assumptions that although are frequent in the texture modeling literature, are still to be broadly accepted and confirmed. Lastly, for applications such as texture classification, detection and coding, common perceptual hypothesis that are useful for synthesis, might not be relevant and thereby the generality here addressed is appropriate.

In Section 2 we present the (weak) modeling assumptions and introduce the underlying

[1] These equivalence criteria are not universal though, as they are defined in the context of each application. For synthesis, textures are usually assigned to the same class if they are *perceived* as the "same," while in classification they are assigned to the same class if they are *made* of the same material.

learning methodology applied to construct texture models. In Section 3 we exemplify the use of the learned model (PDF and texture structure) for texture classification and propose a simple improvement to state-of-the-art texture synthesis algorithms. Conclusions and suggested future work are presented in Section 4.

2 Learning the texture model and structure

Given an area of texture (to be learned/modeled) containing n pixels in locations $V=\{v_1,\ldots,v_n\}$, where $v_i \in Z^2$ is the position of the i^{th} pixel, the density to be learned, $p(\boldsymbol{X}_V)$, models the gray values, $\boldsymbol{X}_V=[x_{v1},\ldots,x_{vn}]$, corresponding to these pixels.[2] We assume, as it is frequently done in the literature, that this density is translation invariant (or stationary), i.e., $p(\boldsymbol{X}_W)=p(\boldsymbol{X}_{W+\boldsymbol{w}})$, $\forall \boldsymbol{w} \in Z^2$.[3]

Numerous estimation techniques for high dimensional PDFs are grouped under the general name of *Graphical Models* [10]. In particular, two different strategies stand out: undirected graphical models (known as *Markov Random Fields*, or MRF), and directed graphical models (known as *Bayesian Networks*, or BN). MRF have been extensively used to model textures (e.g., [2,6,11]). Here we pursue the BN approach, not as widely explored,[4] and are able to exploit an important feature that we describe in detail below: the intrinsic ancestral ordering of the texture pixels given by the scan order of the pixels in the "filling front." An additional advantage of BN as opposed to MRF is that samples can be generated from it in a very simple and efficient way [12], a useful characteristic for texture synthesis.

BN provide the necessary means to factorize a (high dimensional) joint probability density in terms of lower dimensional conditional probability density factors [10,13]. A BN is a very compact representation of the joint probability, completely specified by: 1) a directed acyclic graph (DAG) G, composed of a set of nodes N and a set of edges E, encoding the very important conditional independence relationships among the variables; and 2) the conditional probabilities of each variable given its "parents" (formally defined below). There exists a node $X_i \in N$ in the graph for each random variable, and for this reason we refer indistinctively to nodes and random variables. The set of parents $Pa(X_i)$ of a variable X_i, is defined as those nodes X_j having outgoing arcs ending in X_i. Symbolically, $X_j \in Pa(X_i) \Leftrightarrow (X_j,X_i) \in E$, where (X_j,X_i) is the directed edge from X_j to X_i. The Markov condition, stating that every variable is conditionally independent of all its non-descendents given its parents, allows for the density to be factorized [13] as:

$$p(X)=\prod_{i=1}^{n} p\left(X_i \middle| Pa(X_i)\right) \tag{1}$$

Learning a BN implies learning both the structure of the network (the edges), which encodes fundamental structure in the data, and the corresponding conditional probabilities in (1). The conditional probabilities $p(X_i|Pa(X_i))$, once the structure has been estimated, can be learned using traditional density estimation techniques, and in our case, also the assumption that the texture is translation invariant (or at least the assumption that many samples from the density exist if we want to consider mixtures). This last assumption allows us to construct a sample, $S=\{\boldsymbol{s}^1,\ldots,\boldsymbol{s}^m\}$, from the training texture, I, by translating the pixels involved in the density using every valid vector $\boldsymbol{u} \in Z^2$, ($\boldsymbol{s}^j=[I_{(i \cup Pa(X_i))+\boldsymbol{u}^j}]$).

To account for the fact that noise and illumination conditions can transform a pixel's gray value into a slightly different one, we model the pixels' gray values as continuous random variables, and use the *Kernel Density Estimation* (KDE) technique [14], a non parametric approach. Reasons for the choice of KDE include the relatively inexpensiveness of training samples, the unsuitability of parametric approaches to model the great diversity of textures, the existence of fast computational algorithms, and the success of other nonparametric approaches for texture modeling (e.g. [1]).

---

[2] Throughout this section we use uppercase italic letters (as in $\boldsymbol{X}$) and lowercase italic letters (as in $\boldsymbol{x}$) for random variables and the values those variables can take, respectively. Subscripts (as in X_i or $\boldsymbol{X}_{i \cup j}$) specify a subset of the variable's coordinates.

[3] This assumption can be relaxed and we can learn mixtures of distributions.

[4] Several recent state-of-the-art texture synthesis algorithms (e.g. [1,18-20]) have implicit BN flavor, although that connection is not made explicit in these papers and no attention is paid to optimizing the probability density. The outstanding performance of these algorithms is an additional reason to pursue the BN approach.

KDE uses the sample S to estimate the conditional probabilities according to

$$p\left(x_i \middle| Pa(X_i); w_i\right) = \sum_{j=1}^{m} K\left(\frac{\vec{x}_{i\cup Pa(X_i)} - \vec{s}^{\,j}_{i\cup Pa(X_i)}}{w_i}\right) \Bigg/ w_i \cdot \sum_{j=1}^{m} K\left(\frac{\vec{x}_{Pa(X_i)} - \vec{s}^{\,j}_{Pa(X_i)}}{w_i}\right) \quad (2)$$

where the kernel, $K(.)$, is here chosen to be the Gaussian kernel, $K(t)=(2\pi)^{-1/2}.e^{-t^2/2}$. The kernel's bandwidth, w_i, is automatically computed by maximizing the log-likelihood of (2). Leave-one-out crossvalidation is used to avoid overfitting [14].

To learn the structure of the network, the log likelihood of the joint probability in (1),

$$\sum_{j=1}^{m}\sum_{i=1}^{n} \log p\left(S_i^j \middle| S^j_{Pa(X_i)}\right) = -\sum_{i=1}^{n} H\left(X_i \middle| Pa(X_i)\right) \quad (3)$$

is maximized. The empirical conditional entropy, H, is introduced in (3) to simplify the notation (see [15] for details.)

The maximization of (3) is simplified by the fact that its individual summands were already computed when finding the optimal bandwidths w_i in (2). This then reduces the problem of learning the graph G to the nontrivial problem of choosing the parents of each node, subject to the graph's acyclicity constraint, that maximize (3). This combinatorial optimization problem has been shown to be NP-complete [16]. To simplify the problem, one can impose a predefined order among the variables, $\prec$ (known in the BN literature as topological order or ancestral ordering [13]), constraining the graph to be consistent with this ordering. Formally, if $X_j \in Pa(X_i)$ then $X_j \prec X_i$. Under this assumption, (3) can be optimized by independently maximizing each of its summands, reducing the problem to finding, for each node X_i, the parent set $Pa(X_i)$ (consistent with the ordering) that minimizes the conditional entropy $H(X_i|Pa(X_i))$. Furthermore, the parent set $Pa(X_i)$ of every node X_i (away from the texture borders) is the same, due to the translation invariance hypothesis and the fact that the *set of potential parents* (the set of pixels that can belong to the parent set, *SPP* in the following) is the same (relative to X_i). In the case of pixels close to the image borders, the borders impose an additional constraint on the SPP (apart from the ancestral ordering constraint).

An *equivalence* relationship between pixels can be defined as follows: two pixels are said to be equivalent if they share the same SPP (each one relatively to himself and ignoring pixels that are "far enough" as dictated by the dependencies of the texture). By definition then, we only need to find a parent set per equivalent class, in particular for our problem at hand, for the class of interior pixels, the largest one for sufficiently large textures. The total number of classes depends on the particular texture (through its dependencies), but not on the size of the image to model (for sufficiently large textures).

In this work, we define the ancestral ordering by restricting each pixel to have parents on the same half plane (which half plane is described below). An example of this can be seen in Figure 1a, where the SPP for Pixel_0 is highlighted in yellow. This condition on the SPP is not really limiting, since it is necessary and sufficient to guarantee that an arbitrary shaped, finite, empty patch can be filled using only one (the best) parent set.[5] Not knowing in advance the shape of the region to fill with texture, it is only natural to request that the model is able to fill any arbitrary shaped region. This argument does not depend on the application (filling-in was mentioned for illustrative purposes only), but is related to the causality imposed by (1). Among all possible half planes with their border on Pixel_0, we chose the half plane containing the maximum information (see the formal definition below of the *half plane of maximum information*).

This choice of the SPP, in turn, determines the ancestral ordering shown in Figure 1d. This ordering is therefore natural and is precisely the *filling* order adopted by recent state of the art texture synthesis algorithms (e.g. [1]), in which new pixels are placed at the boundary (or "filling front") between the already filled region and the region to be filled (the source and target regions respectively in the inpainting literature). The success of these approaches suggest that the "flow of information" from the pixels already filled to the pixels to be filled is sufficient to, at least, estimate a density capable of producing results *perceived* as "correct." Therefore, this ordering constraint is not expected to significantly deteriorate the density (graph) estimate, while significantly simplifying its computation. Figures 1b-c show

[5] The proof of this statement is included as supplemental material.

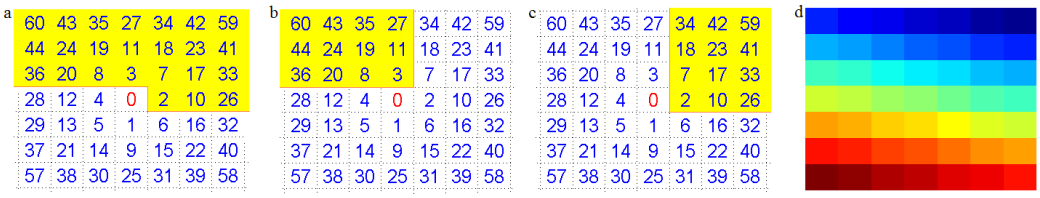


Figure 1: Defining the ancestral ordering. a) Set of possible parents for Pixel_0 (in yellow). b-c) Sets of possible parents for Pixel_0 when it lies on the border of the image. d) "Ancestral ordering" of pixels (blue < yellow < red) imposed by the choice of the yellow region in a).

two different SPPs for the class of pixels in the image borders.

Based on these concepts, and inspired by the work described in [17] for modeling protein ensembles, we propose a quasi-greedy approach to learn the parent set (the network structure) of each pixel equivalence class (recall that this is all what is left to complete the BN). First we learn the parent set for the class of interior pixels.

We start by computing the entropy $H(X_i)$ of a pixel X_i in this equivalence class. This entropy can also be regarded as the conditional entropy between X_i and the empty parent set, $Pa(X_i)=\emptyset$. This is the root node of a tree that is built toward finding the optimum parent set. We refer to this tree as the *entropy tree*. Each node Z in this tree corresponds to a tentative parent set, $Pa(X_i)$, and has an associated value equal to the conditional entropy improvement, $IG(X_i|Z) = H(X_i) - H(X_i|Z)$ (the *information gain* [15]), caused by conditioning on this set.

Then, we compute the information gain, caused by the inclusion of each possible pixel X_j in $Pa(X_i)$, one at a time. Only nodes X_j satisfying the ancestral ordering condition, $X_j \prec X_i$, can be included in $Pa(X_i)$ in order to guarantee that the graph does not have cycles. This reduces the number of terms of the form $H(X_i|X_j)$ that have to be computed. This number is further reduced by the assumption that pixels that are too far away ($|i - j| > r_{\max}$) do not (strongly) depend on each other (in our examples we have used $r_{\max} \approx 20$). For each conditional entropy term computed, a new node is added to the entropy tree at level 1 as the children of the empty set (root) node (see Figure 2d).

Figure 2b shows the entropy reduction (we refer to it as the "entropy map"), and hence the model improvement that can be obtained by including one pixel at a time in the parent set of (the central pixel) X_i. This graph contains the information in the first two levels of the entropy tree (levels 0 and 1). Note how, contrary to what is generally assumed by *patch based texture algorithms* (e.g., see [1,8,18-20]), for (nearly) regular textures the physically closest pixels (to a given pixel) do not always have the greatest influence. We will revisit this observation in Section 3.

By looking at Figure 2b, we define the *half plane of maximum information* (which we referred to earlier and was needed to define the ancestral ordering), as the half of the entropy map whose pixels' added conditional entropy is minimum (i.e., the half plane containing, on average, the best single pixel parents). As an example, in Figure 2b, first row, the half plane of maximum information corresponds to the upper left half. As was mentioned above, if X_i is the center pixel in this entropy map, only pixels in the half plane of maximum information can be part of its parent set.

We continue the construction of the entropy tree by progressively adding parent sets (nodes) to it according to a stochastic quasi-greedy rule:

1. Draw one node Z_1 from the entropy tree with probability proportional to some power b of its IG ($P_{Z1} \propto IG(Z_1)^b$), excluding those sets that do not reduce the uncertainty of the current pixel (their IG is negative). The constant b is selected such that the best node in the tree (the one with highest IG) has a 10% probability of being selected.

2. Choose, with probabilities computed in the same fashion, a child Z_2 of the root node of the entropy tree.

3. Create a new parent set, Z_{new}, by joining the selected sets ($Z_{new}=Z_1 \cup Z_2$). If Z_{new} is not already in the entropy tree, compute its information gain ($IG(X_i|Z_{new})$) and add it to the tree as a child of Z_1. Note that Z_{new} has one more element that Z_1.

4. If the maximum information gain of the entropy tree has not improved for a certain number of iterations (we used 250 in the experiments), exit the loop. Otherwise, return to step **1**.

This algorithm finds, with very high probability for the cases we tested, the same sets that a greedy algorithm would find, and also, those sets close to them. We experimentally

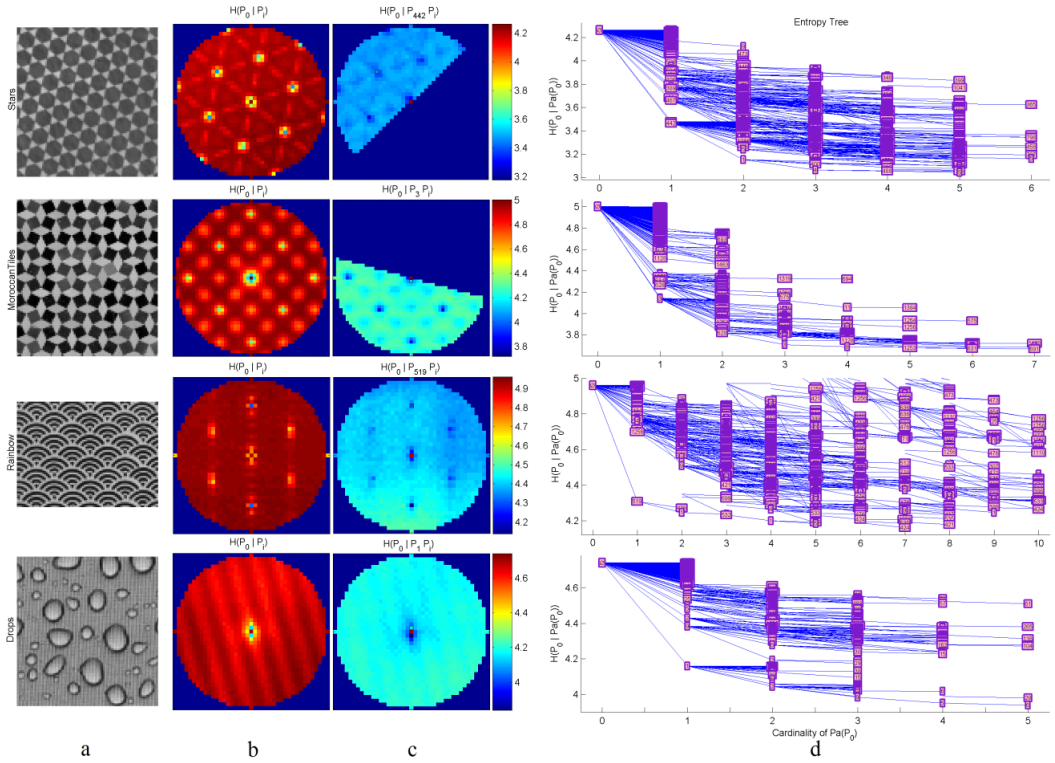


Figure 2: Analysis of four different textures, one per row. a) Original textures. b) Corresponding first order "entropy maps": the conditional entropy H(P_0 | P_i) between the center pixel P_0 and every other pixel P_i. c) Second order "entropy maps": the conditional entropy H(P_0 | P_b P_i), where P_b is the best conditioning single variable (marked with a black square). Computed only for the half plane of maximum information in the first two rows. d) "Entropy trees" for each texture. Each square represents a parent set, but only the last pixel included is shown in the square (see Figure 1 for the location corresponding to each pixel number). The lines connect each set with one of its subsets: the one having one fewer pixel and the lowest conditional entropy. The best parent set can be found very efficiently in this case (and in every case we tested) using the proposed quasi-greedy approach. Many more sets than necessary were processed just to illustrate the structure of the tree.

found through extended testing that these areas of the "entropy tree" consistently contain the best parent sets (as found via exhaustive search). Due to the particular structure of these trees, these are found in an extremely efficient and simple fashion. Figure 2d contains the trees obtained for four different textures. The conditional entropy of the children of the best (lowest) node at level one are better observed in the second order entropy maps in Figure 2c.

Another improvement that proved to have significant impact in performance, without degrading the quality of the parent sets found, consists of pruning the tree branches (corresponding to sets) having lower IG's than their parents (subsets).

Once the entropy tree has been constructed, the best (highest IG, lowest conditional entropy) parent set for the class of interior pixels can be selected. For pixels belonging to other equivalence classes (relatively few if the image is large), the entropy tree has to be searched again, this time including only the pixels in their respective SPP.

We should note that as a consequence of the learning procedure, we also obtain the model order (do not confuse with the ancestral ordering), that is, the number of pixel dependencies that can be learned from the available (finite) data set. This important information is directly obtained as the level of the entropy tree where the conditional entropy starts to increase (see Figure 2d). Having more data points might permit finding larger parent sets.

This concludes the description of the procedure for learning the texture model. We now proceed to present an application to texture classification and a simple improvement to a state-of-the-art texture synthesis algorithm.

3 Results and applications

In the previous section we presented a method to learn the probability density functions and basic underlying structures, encoded as Bayesian Networks, which model different textures. These PDFs can straightforwardly be integrated into a Bayes classifier [21]. Given an instance of a texture of an unknown class, the probability of this texture according to each learned model is computed, and the texture is assigned the label of the most probable class.

To test our texture learning framework with this application, we selected a subset of 66 images from the Brodatz database. Each of these images was split, using a 4 by 4 grid, into 16 non overlapping sub images. For each image, one sub image was committed for training and the other 15 to evaluate the classification performance (in total, 66 sub images were used for training and 66x15=990 for evaluation). Images from the Brodatz database were visually inspected and discarded if the stationarity requirement was not satisfied (i.e. their sub images did not produce similar statistics; see texture D43 in the Brodatz database for an example of this). No preprocessing was carried out on the images to compensate for variations (e.g. due to the illumination).

Using this dataset we compared the classification performance of the models described in Section 2 (whose optimal parent sets have on average 2.6 pixels) against control models having a fixed parent set with the closest pixels to the origin (4 pixels in size; larger parent sets should perform worse due to severe overfitting). Our model performed slightly better (77.1% versus 75.9%), used a smaller context (2.6 average against 4), and thereby run in 35% less time (the computation is linear in the context size).

We move now into the application of texture synthesis. An immediate question is whether this analysis (modeling) is ready, as it is, to be used for synthesis (which is of course just one of the many important applications of good texture models). Using our proposed learning framework and manageable sample sizes ($m < 10000$, recall that m is the number of pixels in the training texture), the optimal parent sets found are often relatively small (between 4 and 7 pixels in Figure 2d) in order to directly synthesize new textures from the learned PDF, unless the texture is such that small contexts are sufficient to characterize it (see Figure S1 in the supplemental material for one such example). This is expected since, as mentioned before, the optimality criterion imposed to learn the PDF was not perceptually motivated (if it were so, we may give more weight to learning the edges correctly). We then proceed differently.

Looking at "entropy maps" (as in Figure 2b-c), we observe that the assumption of spatial locality and isotropy often made in state-of-the-art texture synthesis methods (e.g. [1,18-20]) is poorly satisfied in textures containing structure and some degree of periodicity. This is often the reason for the deficient performance attained by Efros and Leung's (E&L) algorithm [1] with smaller than recommended window sizes on these textures (see first two columns, 1st to 3rd rows in Figure 3). E&L recommend setting the window (context) size equal to the size of the biggest texture feature, which the user is asked to determine. We propose, motivated by our learning framework, an alternative approach that automatically selects the "context" or parent set (pixels that must be looked at when synthesizing a new pixel). Both close and far away pixels can be simultaneously used as part of the context if the structure of the texture, as automatically learned, dictates so.

We suggest choosing the context (and after that proceeding as in E&L) as the set of pixels in the allowed region (see discussion regarding the "ancestral ordering" in the previous section) that have the greatest information gain (when considered individually). Notice how this criterion selects pixels close to the pixel being generated to maintain the local appearance of the texture, but also pixels far away to respect its structure (Figure 4). The corresponding weights for the pixels are chosen proportionally to the information gain of each pixel. Results of this approach are shown in Figure 3, 3rd and 4th columns.

Our approach works significantly better for textures that violate the spatial locality or isotropy hypothesis (e.g., Figure 3, 1st to 3rd rows). Furthermore, our approach uses much smaller contexts and, therefore, is considerably less computationally expensive (at the synthesis stage), and does not penalize the synthesis of small scale texture embedded in bigger scale regularity. On the other hand, for textures where the locality is a proper assumption, we do not necessarily improve the results of E&L (see last row of Figure 3).

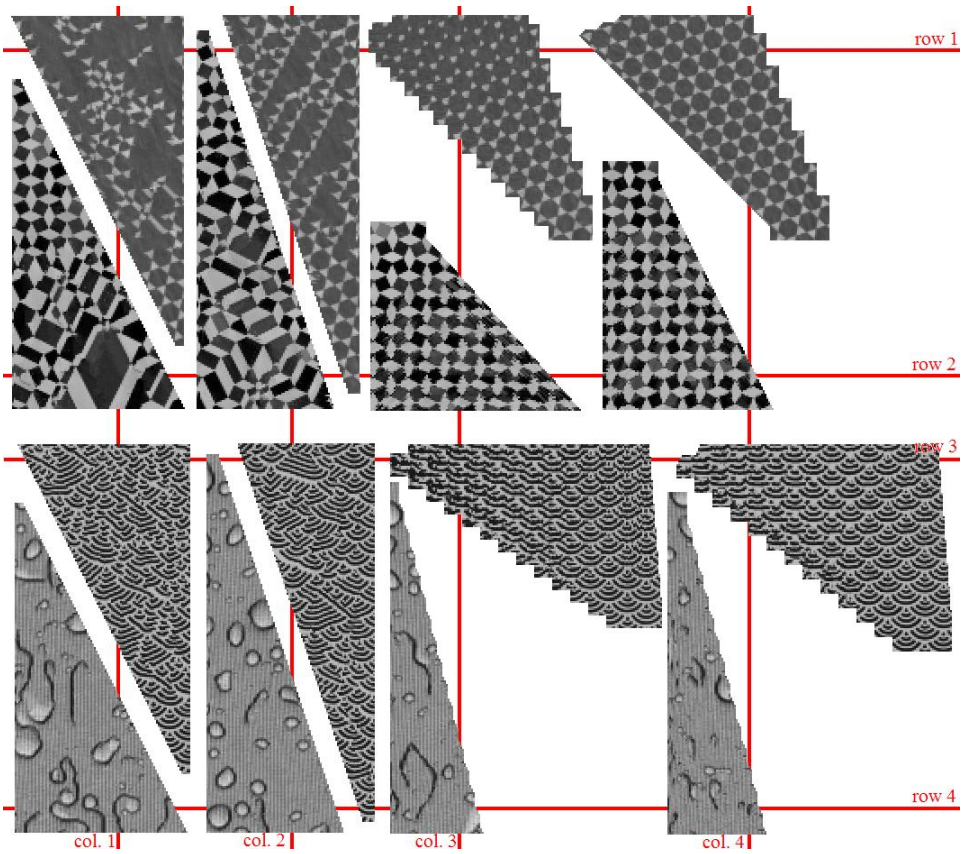


Figure 3: Comparison of the proposed algorithm against the algorithm proposed by E&L on the textures of the previous figure. 1st and 2nd columns: E&L results with contexts of size 12 and 24 pixels, respectively. 3rd and 4th columns: results of our proposed method with contexts of size 12 and 24 pixels, respectively. Synthesized images are not square because only the best parent sets for each of three pixel equivalence groups were used: one for the interior pixels and one for the pixels close to each of the borders. Synthesis starts in every case in the smaller side of the "quadrilateral" and proceeds to the opposite side. Note how for the first three textures, our learned context leads to better results. The fourth row shows a partial failure of the proposed approach (while also E&L produces poor results in this case).

4 Conclusions and future work

We presented a novel approach to learn and model textures with probability density functions by means of Bayesian Networks. We learned these BNs relying on very weak and general hypotheses. Along the way, we learned meaningful structural information about the texture in the form of what we call "entropy trees" and "entropy maps."

Inspired by the proposed texture modeling framework here introduced, a modification to the fundamental texture synthesis algorithm by E&L was proposed. This modification shows a clear improvement in textures with some degree of structure, suggesting that the choice of square patches (weighted by isotropic Gaussians) is not optimal for this kind of textures. Furthermore, other metrics consistent with the current knowledge on texture perception, if substituted by the current model selection criterion (maximum likelihood), could further improve the learned context. Similarly, we showed that texture classification can benefit from the learning of contexts, perhaps as part of current state-of-the-art methods (e.g. [8]). These improvements are based on the automatic selection of the context used by the synthesis/classification algorithms.

The study of information gains going beyond individual pixels might provide further insights into the automatic computation of the most relevant context (Figure 2c). The approach here developed for pixel gray values can be easily extended to other features, such as gradients, color, or filter responses. The use of our proposed approach for tasks such as texture detection, compression, and denoising, will be investigated in the future.

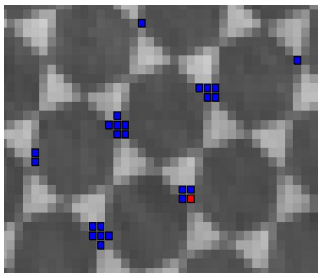


Figure 4: The parent set used to synthesize the texture in the 1st row, 4th column, of Figure 3. The red square is the pixel being synthesized and the blue squares are the pixels in the parent set, automatically learned with our proposed framework. Notice that the ancestral ordering constraint forces the parent set to lie only on one side of the pixel being synthesized. For each pixel in the parent set, a weight is automatically computed based on the information gain of the pixel by itself, as explained in the text. Note how both local and global structure is used.

References

[1] A. A. Efros and T. K. Leung, "Texture synthesis by non-parametric sampling." *ICCV*, vol. 2, pp. 1033-1038, 1999.

[2] R. Paget and I. D. Longstaff, "Texture synthesis via a noncausal nonparametric multiscale Markov Random Field." *IEEE Trans. Image Process.*, vol. 7, pp. 925-931, 1998.

[3] E. P. Simoncelli and J. Portilla, "Texture characterization via joint statistics of wavelet coefficient magnitudes." *ICIP 98.*, vol. 1, pp. 62-66, 1998.

[4] R. M. Haralick, "Statistical and structural approaches to texture." *Proc IEEE*, vol. 67, pp. 786-804, 1979.

[5] D. J. Heeger and J. R. Bergen, "Pyramid-based texture analysis/synthesis." *SIGGRAPH*, pp. 229-238, 1995.

[6] S. C. Zhu, Y. N. Wu and D. Mumford, "Minimax entropy principle and its application to texture modeling." *Neural Comput.*, vol. 9, pp. 1627-1660, 1997.

[7] A. Zalesny and L. Van Gool, "A Compact Model for Viewpoint Dependent Texture Synthesis." *Lecture Notes in Computer Science*, vol. 2018, pp. 124-141, 2001.

[8] M. Varma and A. Zisserman, "A Statistical Approach to Texture Classification from Single Images." *IJCV*, vol. 62, pp. 61-81, 2005.

[9] M. S. Landy and N. Graham, "Visual perception of texture." in *The Visual Neurosciences*. L. M. Chalupa and J. S. Werner, Eds. Cambridge, MA: MIT Press, pp. 1106-1118, 2004.

[10] M. I. Jordan, *Learning in Graphical Models*. Boston, MA.: Kluwer Academic Publishers, 1998.

[11] G. Winkler, *Image Analysis, Random Fields and Markov Chain Monte Carlo Methods: A Mathematical Introduction*. Berlin; New York: Springer-Verlag, 1995.

[12] R. Cowell, "Advanced inference in bayesian networks." in *Learning in Graphical Models* M. I. Jordan, Ed. Boston: Kluwer Academic Publishers, pp. 27-49, 1998.

[13] R. E. Neapolitan, *Learning Bayesian Networks*. NJ, USA: Pearson, Prentice Hall, 2004.

[14] B. W. Silverman, *Density Estimation for Statistics and Data Analysis*. New York: Chapman and Hall, 1986.

[15] T. M. Cover and J. A. Thomas, Elements of Information Theory. John Wiley & Sons, Inc., 1991.

[16] D. M. Chickering, "Learning Bayesian networks is NP-complete." Proceedings of AI and Statistic., 1995.

[17] D. Rother, G. Sapiro, and V. Pande, "Statistical Characterization of Protein Ensembles." IEEE/ ACM Transactions on Computational Biology and Bioinformatics 5: pp. 42-55, 2008.

[18] M. Ashikhmin, "Synthesizing natural textures." in Proceedings of 2001 ACM Symposium on Interactive 3D Graphic, pp. 217-226, 2001.

[19] A. Criminisi, P. Perez and K. Toyama, "Object removal by exemplar-based inpainting." in CVPR, pp. 721-728, *2003*.

[20] L. Wei and M. Levoy, "Fast texture synthesis using tree-structured vector quantization." in *SIGGRAPH*, pp. 479-488, 2000.

[21] T. Hastie, R. Tibshirani and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York: Springer-Verlag, 2001.

Additional Figures:

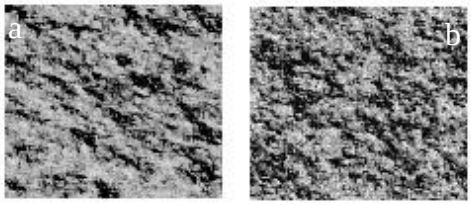


Figure S1: Direct synthesis example for a simple texture. a) The original texture. b) The texture synthesized directly using the learned PDF. See text and Figure 3 for an alternative technique derived from our approach and more difficult textures.

Filling-in property

In this section we address the following question: What is a necessary and sufficient condition on the Set of Potential Parents (SPP) to guarantee that a single parent set can be used to fill-in a finite arbitrarily shaped area?

Theorem:

$Pa(X_0) \subset HP \Leftrightarrow$ it can be used to fill-in a finite arbitrarily shaped area.

In words, the parent set of the current pixel is included in a half plane (HP) if and only if it can be used to fill-in a finite arbitrarily shaped area.

Proof:

($\Rightarrow$) Suppose, with no loss of generality, that the half plane is defined as in Figure 1a. Fill-in the pixels one at a time, starting from the rightmost pixel on the upper line, and continuing from right to left and from top to bottom until all the pixels have been filled.

($\Leftarrow$) The parent set can be used to fill-in any arbitrary area. Therefore, exists an ordering "$\prec$" in this area such that if $X_i \in Pa(X_j) \Rightarrow X_i \prec X_j$.

By absurd, suppose there exists $\vec{u} \in Z^2$ such that $\begin{cases} X_{\vec{u}} \in Pa(X_0) \\ X_{-\vec{u}} \in Pa(X_0) \end{cases}$ (there are opposing pixels in the parent set and therefore it can not be contained in a HP).

Then, $X_{\vec{u}}, X_{-\vec{u}} \prec X_0$. But $X_0 \in Pa(X_{-\vec{u}}) \Rightarrow X_0 \prec X_{-\vec{u}}$.

This is an absurd, therefore such $\vec{u} \in Z^2$ does not exist and $Pa(X_0) \subset HP$.

QED.