Analysis of Sampling-based Texture Synthesis as a Generalized EM Algorithm

Liu-yuan Lai[†], Wen-Liang Hwang[†], Silong Peng[‡]

Institute of Information Science, Academia Sinica, Taiwan[†]

Institute of Automation, The Chinese Academy of Sciences, China[‡]

Abstract

Research on texture synthesis has made substantial progress in recent years, and many patch-based sampling algorithms now produce quality results in an acceptable computation time. However, when such algorithms are applied to textures, whether they provide good results for specific textures, and why they do so, are questions that have yet to be fully resolved. In this paper, we deal specifically with the second question by modeling the synthesis problem as learning from incomplete data. We propose an algorithm and show that the solution of many sampling-based algorithms is an approximation of finding the maximum-likelihood optimum by the generalized expectation and maximization (EM) algorithm.

1 Introduction

Texture synthesis has long been an important topic in the fields of computer graphics, computer vision, and image processing because of its practical and theoretical value. The objective of texture synthesis is to create new images from a given sample texture image such that, to human observers, the new images appear to be generated by the same underlying process as the sample texture. The technique of texture synthesis has improved significantly in recent years so that it is now possible to design algorithms that synthesize some textures efficiently and yield high quality results.

We are especially interested in texture synthesis approaches that utilize sampling techniques because they are usually simple and can be implemented easily. Sampling-based methods model textures as homogeneous Markov random fields, so the value of a pixel depends on the distribution of its neighboring pixels' values. Therefore, by sampling according to this distribution of local dependency, these methods generate new textures; [3, 12, 1] are some examples that fall into this category. Surprisingly, these simple methods produce good results for a broad range of textures. Many researchers have addressed the speed issue and developed efficient algorithms to reduce the complexity of the sampling procedure [13, 9, 15]. It has been reported that the methods based on sampling a patch usually achieve both speed and quality improvements over those based on sampling a pixel. The improved speed results from pasting one block of pixels, instead of a single pixel, as well as searching for that block in a small pool of candidates. Many new patch-based algorithms attempt to further improve the visual quality of the synthesized textures without sacrificing the speed advantage. An important aspect of improving the visual quality is how to eliminate artifacts along the boundaries of patches; another is how to prevent artifacts caused by errors accumulating over a large range [4, 10, 8]. Meanwhile, the work in [7] addresses the issue of quantifying the global qualtity of a synthesized texture as a way of overcoming the limitation imposed by the Markov random field property.

Unlike sampling-based approaches, some synthesizing algorithms model textures as two-dimensional homogeneous random fields that can be characterized by a few statistical descriptions. Thus, the approach is feature-based. The hypothesis for texture synthesis based on this approach is that: two textures with similar statistical descriptions are likely to be visually indistinguishable. Thus, a new texture is generated so that its statistical descriptions closely match those of the sample texture [6, 16, 2, 11]. As noted in [11], the primary problem of the feature-based approach is that there is no formal way to verify the sufficiency or necessity of the statistical descriptions used to synthesize textures. It is therefore possible that a generated texture with the same statistical descriptions as those of the input texture could be perceived as a different texture. Also, the formulation of this approach is complex, so finding the optimal solution is difficult.

Motivation and Contribution

Our study is motivated by the success of sampling-based algorithms in reproducing a broad range of texture images. These algorithms search a pool of candidate patches, and sample the set of blocks that matches the target boundary region well. The sampled block is then pasted onto the target image, and the search for the next block continues until either the whole image is covered [9], or a stopping criterion is reached [8].

Though surprised that the simple algorithms can achieve such excellent results, we found that it cannot resolve two questions crucial to the understanding of the texture synthesis problem. First, for what kind of textures do the algorithms yield perceptually acceptable results? Second, is there an optimization procedure underlying the algorithms? The first question asks what kind of process can be accurately modeled by sampling-based procedures, while the second asks if the solutions can be quantified. It is clear that modeling textures as Markov random fields is too general to provide further analysis of either topic.

In this manuscript, we address the second question. We present the texture synthesis task as a problem of obtaining a maximum-likelihood estimation from incomplete data. The input texture image is the incomplete data that we observe, while the texture image to be synthesized is the unobserved data. After making assumptions required to solve the estimation problem efficiently, our proposed algorithm performs texture synthesis as a generalized EM algorithm. The synthesized textures are approximations of the maximum-likelihood solutions estimated from the input textures. We then relate some sampling-based algorithms to our approach and demonstrate that they can be viewed as generalized EM algorithms [5].

The remainder of the paper is organized as follows. In Section 2, we describe how we formulate the texture synthesis problem as a process of maximum likelihood estimation from incomplete data, and analyze the problem using the generalized EM method. In Section 3, we discuss methods for generating hidden structures and textures. Finally, we present our conclusions in Section 4.

2 Formulation and Analysis

In this section, we formulate our algorithm as a process of parameter estimation in a missing data problem. For convenience, we introduce a parameterized hidden structure and, based on the structure, the synthesized image is constructed.

2.1 Generalized EM Analysis

Let D be the data that includes an input texture, I, and a synthesized texture, $\hat{I}$; and let R be the parameters of the auxiliary hidden structure C that generates $\hat{I}$. Also, let $P(D|R) = P(I, \hat{I}|R)$. The objective of our texture synthesis algorithm is to maximize $\log P(I, \hat{I}|R)$ given the input texture I. For any probability $T(C|I, \hat{I}^t, R^t)$, the log probability is

$$\begin{aligned}
&\log P(I, \hat{I}|R) \\
&= \log \sum_C P(I, \hat{I}, C|R) \\
&= \log \sum_C T(C|I, \hat{I}^t, R^t) \frac{P(I, \hat{I}, C|R)}{T(C|I, \hat{I}^t, R^t)} \\
&\geq \sum_C T(C|I, \hat{I}^t, R^t) \log \frac{P(I, \hat{I}, C|R)}{T(C|I, \hat{I}^t, R^t)}. \quad (1)
\end{aligned}$$

The bound used in the above takes the form of Jensen's inequality:

$$\log \sum_i t_i a_i \geq \sum_i t_i \log a_i,$$

where $\sum_i t_i = 1$. The goal of the generalized EM algorithm is to estimate $T(C|I, \hat{I}^t, \hat{R}^t)$ and $P(I, \hat{I}, C|R)$ jointly, so as to maximize this bound. Note that the solution of the generalized EM algorithm depends on $T(C|I, \hat{I}^t, R^t)$. We use the generalized EM because the EM solution of our application, where $T(C|I, \hat{I}^t, R^t) = P(C|I, \hat{I}^t, R)$, is difficult to estimate.

Let f be the method that obtains the parameters R from structure C; i.e. $R = f(C)$. We define

$$T(C|I, \hat{I}^t, R^t) \propto \exp^{-\frac{1}{\sigma^2}\, d(R^t \| f(C))}, \qquad (2)$$

where $d(R^t \| f(C))$ measures the discrepancy between R^t and $f(C)$; and σ^2 determines the sharpness of the distribution over the optimal class map C^{t+1}, where $d(R^t \| f(C^{t+1}))$ is a minimum. If σ^2 is a very small value, (2) can be approximated as

$$T(C|I, \hat{I}^t, R^t) \approx \begin{cases} 1 & \text{if } C = C^{t+1}, \\ 0 & \text{otherwise.} \end{cases} \qquad (3)$$

Then, because the entropy of a random variable with probability (3) is 0, we have the following approximation of (1):

$$\begin{aligned}
&\log P(I,\hat{I}|R)\\
&\geq \sum_C T(C|I,\hat{I}^t,R^t)\log\frac{P(I,\hat{I},C|R)}{T(C|I,\hat{I}^t,R^t)}\\
&\approx \max_C T(C|I,\hat{I}^t,R^t)\log P(I,\hat{I},C|R) \quad (4)\\
&\approx \log P(I,\hat{I},C^{t+1}|R). \quad (5)
\end{aligned}$$

To improve the bound in (4), we find

$$C^{t+1} = \arg\max_C T(C|I,\hat{I}^t,R^t), \quad (6)$$

and derive the next R^{t+1}:

$$R^{t+1} = f(C^{t+1}). \quad (7)$$

We then set $R = R^{t+1}$ in (5), and find the next $\hat{I}^{t+1}$:

$$\hat{I}^{t+1} = \arg\max_{\hat{I}} \log P(I,\hat{I},C^{t+1}|R^{t+1}). \quad (8)$$

The generalized EM algorithm improves the lower bound of $\log P(D|R)$ by iteratively applying the following two-phase algorithm.

1. **Structure Generation:** According to (6), from I, I^t and R^t, we find C^{t+1}, which minimizes $d(R^t||R^{t+1})$, where $R^{t+1} = f(C^{t+1})$ and f is a method used to obtain parameters from a hidden structure.

2. **Synthesizing Images:** Based on C^{t+1}, we sample patches from image I and paste them to obtain the image I^{t+1} according to (8).

The solution found by the algorithm is an approximation of the maximum-likelihood solution. The performance of our approach depends on the methods used to obtain (6) and (8), which correspond to how we generate the auxiliary hidden structure and the synthesized texture, respectively.

3 Hidden Structure and Texture Generation

We use the generalized EM algorithm to quantify the synthesized images of a sampling-based texture synthesizing approach. We quantify the synthesized images according to the methods of hidden structure and texture generation. The simplest structure is called *lazy* structure, where C has only one configuration and $C^t = C^{t+1}$; therefore, $R^t = R^{t+1}$ for all t. In this case, the first phase of our generalized EM algorithm produces the same structure in each iteration. Textures are synthesized based purely on the image synthesizing procedure of the second phase. Although many sampling-based texture synthesizing algorithms can be classified as having lazy structure case, not all structures are lazy. As proposed in [14], the hidden structure is an image of two symbols, where two one-dimensional probability context free grammar (PCFG) are used to represent the hidden structure. The empirical probabilities of the grammar of a given structure are the parameters, R. The Kullback-Leibler divergence is used to measure the distance of two PCFGs, and a new hidden structure is generated according to a random greedy algorithm. This approach shows that a grammar structure can be used in our generalized EM algorithm. Any other method representing a hidden structure can be incorporated into our generalized EM algorithm.

The second phase of our algorithm synthesizes images according to the supervision of the hidden structure. There are many outstanding image synthesizing procedures, based mainly on smart pasting and random sampling procedures. We analyze a few representative sampling-based synthesizing algorithms with our framework in the following.

3.1 Patching Images and Sampling

To obtain a synthesized image, (8) can be rewritten as

$$\hat{I}^{t+1} = \arg\max_{\hat{I}} \log\bigl(P(I, \hat{I}|C^{t+1}, R^{t+1})P(C^{t+1}|R^{t+1})\bigr). \tag{9}$$

The probability $P(C^{t+1}|R^{t+1})$ does not depend on $\hat{I}$; therefore, the maximization of (9) can be simplified as

$$\hat{I}^{t+1} = \arg\max_{\hat{I}} \log\bigl(P(I, \hat{I}|C^{t+1}, R^{t+1})\bigr). \tag{10}$$

We define $\log P(I, \hat{I}|C^{t+1}, R^{t+1})$ as negatively proportional to a cost function $K(\hat{I})$:

$$\log P(I, \hat{I}|C^{t+1}, R^{t+1}) \propto -K(\hat{I}). \tag{11}$$

Next, we discuss the corresponding cost functions of two major approaches: the patch-work approach in [4, 9] and the graph-cut approach in [8]. Note that many other sampling-based algorithms can be regarded as variations of [4, 9, 8] with different cost functions.

In the **patch-work** approach, a random procedure is adopted to avoid pasting repeated patterns. $\hat{I}^{t+1}$ can be found by using the random sampling procedure:

$$\hat{I}^{t+1} = \text{Random}\{\arg\min_{\hat{I}} K(\hat{I})\}. \tag{12}$$

Let

$$\hat{I}^{*} = \arg\min_{\hat{I}} ||\partial\hat{I}||,$$

where $\|\partial\hat{I}\|$ denotes the total pasting error of $\hat{I}$. The cost function K can be defined as

$$K(\hat{I}) = \begin{cases} \|\partial\hat{I}^{*}\| & \text{if } |\|\partial\hat{I}\| - \|\partial\hat{I}^{*}\|| < \tilde{\epsilon}, \\ \infty & \text{otherwise.} \end{cases} \tag{13}$$

For a sufficiently large $\tilde{\epsilon}$, there are many images that would minimize the K function. One of the images is chosen as $\hat{I}^{t+1}$. Because it is computationally infeasible to obtain $\hat{I}^{*}$, the patch-work approach and its variations use a local greedy algorithm to approximate the optimum.

In the **graph-cut** approach, a probability function is defined for selecting an image $\hat{I}$ at each step t:

$$P(\hat{I}) \propto e^{\frac{-\Delta(\hat{I}, \hat{I}^{t})}{k\sigma^2}}, \tag{14}$$

where k is a parameter controlling the randomness of patch selection, and σ is the standard deviation of the pixel values in the input image. $\Delta(\hat{I}, \hat{I}^{t})$ is a measurement of the discrepancy between $\hat{I}$ and $\hat{I}^{t}$. Depending on the placement and matching methods used, several different Δ measurements have been proposed in [8]. $\hat{I}^{t+1}$ is sampled from the probability function, as in (14).

Comparing the above equation to (11), the cost function K of the graph-cut algorithm is therefore

$$K(\hat{I}) = \frac{\Delta(\hat{I}, \hat{I}^{t})}{k\sigma^2}. \tag{15}$$

When k is set to a low value, then $P(\hat{I})$ peaks at the minimum of $\Delta(\hat{I}, \hat{I}^{t})$, and $\hat{I}^{t+1}$ would be close to

$$\hat{I}^{t+1} = \arg\min_{\hat{I}} K(\hat{I}). \tag{16}$$

When k is set to a larger value, the probability $P(\hat{I})$ is spread out; thus, the above approximation is not valid. In this case, the graph-cut algorithm does not fit into our proposed framework. The graph-cut algorithm iterates the two phases until a stopping criterion is reached. In Table 1, several texture synthesizing methods are listed and how they fit into our proposed framework are presented.

4 Conclusion

We model the sampling-based texture synthesizing approach as maximum-likelihood estimation from incomplete data. By some convenient assumptions that allows us to efficiently solve the generalized EM algorithm, we quantify the synthesized images as an approximation of the maximum-likelihood solution. We relate some sampling-based algorithms to our approach, and show that they can be regarded as generalized EM algorithms. They have different cost functions and use different methods to evaluate the cost for pasting image. In many sampling-based algorithms, the hidden structures are the same for each iteration. Consequently, the quality of the generated texture is based purely on the pasting algorithm. Whether modifying the hidden structure at each iteration would produce a better synthesized texture remains an open question.

References

[1] M. Ashikhmin. Synthesizing natural textures. In *The proceedings of 2001 ACM Symposium on Interactive 3D Graphics*, pages 217–226, March 2001.

[2] J. S. D. Bonet and P. Viola. A non-parametric multi-scale statistical model for natural images. In *Advances in Neural Information Processing Systems*, volume 10, 1998.

[3] A. Efros and T. Leung. Texture synthesis by non-parametric sampling. In *International Conference on Computer Vision*, volume 2, pages 1033–1038, September 1999.

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

[5] B. J. Frey. Graphical models for machine learning and digital communication. *The MIT Press, Cambridge, Massachusetts*, 1998.

[6] 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*, pages 229–238, August 1995.

[7] V. Kwatra, I. Essa, A. Bobick, and N. Kwatra. Texture optimization for example-based synthesis. *ACM Trans. Graph.*, 24(3):795–802, 2005.

[8] V. Kwatra, A. Schdl, I. Essa, G. Turk, and A. Bobick. Graphcut textures: Image and video synthesis using graph cuts. *ACM Transactions on Graphics, SIGGRAPH 2003*, 22(3):277–286, July 2003.

[9] L. Liang, C. Liu, Y. qing Xu, B. Guo, and H. yeung Shum. Real-time texture synthesis by patch-based sampling. *ACM Transactions on Graphics*, 20(3):127–150, July 2001.

	Structure	Pasting	Iterations	Stopping Cond.
Efros el al. [4]	lazy	Eqns (12)(13)	1	$R^{t+1} = R^t$
patchwork [9]	lazy	Eqns (12)(13)	1	$R^{t+1} = R^t$
graphcut [8]	lazy	Eqns (16)(15)	many	
Lai et al. [14]	PCFG	Eqns (12)(13)	many	$\lvert R^{t+1} - R^t \rvert < \epsilon_0$

Table 1: Comparison of synthesis methods under the proposed analysis.

[10] A. Nealen and M. Alexa. Hybrid texture synthesis. In *EGRW '03: Proceedings of the 13th Eurographics workshop on Rendering*, pages 97–105, 2003.

[11] J. Portilla and E. P. Simoncelli. A parametric texture model based on joint statistics of complex wavelet coefficients. *International Journal of Computer Vision*, 40(1):49–71, 2000.

[12] Y. W. S. Zhu and D. Mumford. Filters, random fields and maximum entropy (frame) - towards a unified theory for texture modeling. *International Journal of Computer Vision*, 27(2):107–126, 1998.

[13] L.-Y. Wei and M. Levoy. Fast texture synthesis using tree-structured vector quantization. In *SIGGRAPH*, pages 479–488, 2000.

[14] L. yuan Lai, W.-L. Hwang, and S. Peng. A sampling-based gem algorithm with classification for texture synthesis. In *International Conference on Acoustics, Speech, and Signal Processing*, May 2006.

[15] S. Zelinka and M. Garland. Towards real-time texture synthesis with the jump map. In *EGRW '02: Proceedings of the 13th Eurographics workshop on Rendering*, pages 99–104, 2002.

[16] S. C. Zhu, Y. N. Wu, and D. Mumford. Minimnax entropy principle and its application to texture modeling. *Neural Computation*, 9(8):1627–1660, 1997.