Parametric Texture Model based on Joint Statistics

Gowtham Bellala, Kumar Sricharan, Jayanth Srinivasa

Department of Electrical Engineering, University of Michigan,
Ann Arbor

1. INTRODUCTION

Texture images are a special class of images that are spatially homogeneous and consist of repeated elements, often subject some randomization in their location, size, color, orientation, etc. Textures can be classified into different classes or groups based on their structure and origin. Figure 1 gives some example textures. Textures are widely used in varied fields ranging from bio medical imaging to computer graphics. The flexibility and advantage of texture images is that they can be statistically modeled.

The three basic reasons that lead to the statistical modeling of images are the Julesz conjecture, the fact that texture images can be treated as Markov Random fields and the use of linear kernels in multiple scales for image analysis and representation.

Julesz hypothesized that the statistics of a texture and its appearance are inter related. The Julesz hypothesis said that two images would be visually indistinguishable if their N^{th} order pixel statistics match. Therefore statistical modeling could be used to classify textures. The theory of Markov random fields, in which the full model is characterized by statistical interactions within local neighborhoods and the use of oriented linear kernels at multiple spatial scales for image analysis and representation are recent developments and have allowed newer ways of modeling textures.

In the paper[1] we have referred to ,the authors have used an over-complete wavelet representation and the Markov statistical descriptors are pairs of wavelet coefficients at adjacent spatial locations, orientations and scales. Later they have formulated an efficient algorithm for image synthesis based on these constraints using projections onto sets. In our studies and implementation, we have followed a very similar approach and have also successfully implemented various applications presented in the paper.

Figure 1:: *The left column has a periodic , artificial texture, the middle column shows a pseudo periodic texture and the right column shows a random texture.*

2. STATISTICAL FRAMEWORK

2.1. Interpreting the Julesz Hypothesis:

As mentioned before, Julesz hypothesis is a seminal work that binds statistical definition of textures to visual perception. In the Julesz hypothesis the definition of a texture images are treated as two dimensional homogeneous random field (RF), X (n, m) on a finite lattice (n, m) ε L $\subset Z^2$.In order to develop an implementation to the classification problem, the Julesz hypothesis itself has been reformulated: there exists a set of functions $\{\phi(X), k = 1,\ldots,N_c\}$, such that samples drawn from any two Random fields that are equal in expectation over this set are visually indistinguishable under some fixed comparison conditions. These set of functions are called the constraint functions.

2.2. The Constraint Functions:

In drawing up a set of constraint functions, a number of interpretations to the Julesz conjecture have been made. Some people have interpreted the Julesz hypothesis as both a necessary and sufficient for two textures to be perceptually similar. That is, if one were to find such a set of functions, one would be able to statistically and uniquely identify any class of texture, without any redundancy. Since the basic test for the hypothesis is human vision, such a set would also give some idea about perception of textures by the human eye.

Given all the assumptions it is not very easy to test a specific set of constraints. For example the term visually indistinguishable is a loose term and depends on the conditions in which the two textures are compared. Also ergodicity, which is defined in the limit of an infinite image is assumed. For the definition of visual perception, the authors [1] refer to preattentive judgments, in which a human subject has to make rapid decisions, without careful inspection. The authors [1] also define a stronger form of ergodicity, which they refer to as '*practical ergodicity*', according to which a homogeneous Random Field has '*practical ergodicity*' with respect to a function only if the spatial average over a sample image in that field is a good approximate to the expectation of the field itself.

In order to verify the constraint functions, we have to verify if two images with the same constraint values will be visually indistinguishable. This can be done using a set of sample images and comparing the images with same expected values of constraint functions. But it is unlikely for us to find two such images in a random sample set. The most efficient way to test the algorithm is by the 'synthesis by analysis' approach.

Algorithm 1: Synthesis by Analysis

1 A library of texture images are used to get the constraint values.
2 An algorithm is used [1] to synthesize texture images that satisfy the constraint values.
3 Visual Perception is the tool used to distinguish between the original and the synthesized textures

3. TEXTURE MODEL

Since texture comparisons are to be done by human observers, one source of inspiration is our knowledge of the earliest stages of human visual processing. If the set of constraint functions can be chosen to emulate the transformations of early vision, then textures that are equivalent according to the constraint functions will be indistinguishable to human eye. As is common in signal processing, we proceed by first decomposing the signal using a linear basis. The constraint functions are then defined on the coefficients of this basis.

3.1 Local Linear Basis

We wish to choose a fixed multi scale linear decomposition whose basis functions are spatially localized, oriented and roughly one octave in bandwidth. In addition, we should be able to invert this linear transformation for synthesizing back the texture. In this regard, a bank of Gabor filters at suitable orientations and scales would be inconvenient. At the same time, we would not like our transformation to be translation variant, as this could cause artifacts in an application like texture synthesis. Thus, we choose to use a "steerable pyramid", since this transformation has nice reconstruction properties like tight frame (i.e. the matrix corresponding to the inverse transformation is equal to the transpose of the forward transformation matrix) in addition to being translation and rotation invariant.

3.2 Steerable Pyramid Decomposition

Similar to conventional orthogonal wavelet decompositions, the steerable pyramid is implemented by recursively splitting an image into a set of oriented subbands and a lowpass residual band. The system diagram for the transform is shown in the figure below.

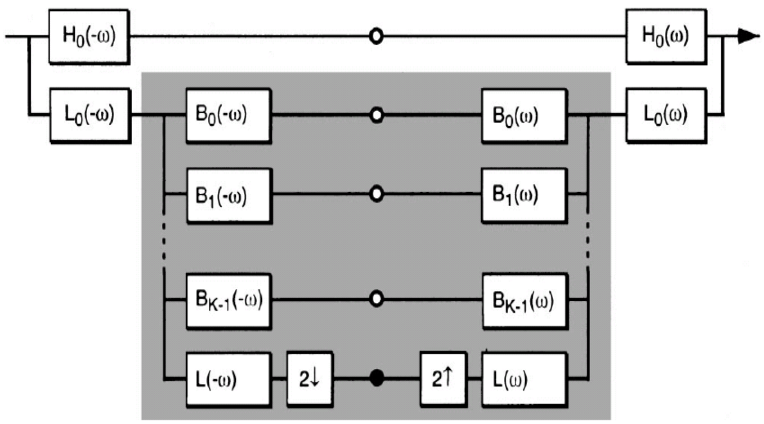

Figure 2. *System diagram for the steerable pyramid. The input image is initially split into high and low pass bands, which is then further split into a lower frequency band and a set of oriented bands (Source : Simoncelli and Portilla [1])*

The filters used in this transformation are polar separable in the Fourier domain. They are given by:

$$L(r,\theta) = \begin{cases} 2\cos\left(\frac{\pi}{2}\log_2\left(\frac{4r}{\pi}\right)\right), & \frac{\pi}{4} < r < \frac{\pi}{2} \\ 2, & r \le \frac{\pi}{4} \\ 0, & r \ge \frac{\pi}{2} \end{cases}$$

$$B_k(r,\theta) = H(r)G_k(\theta), \quad k \in [0, K-1],$$

with radial and angular parts

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

$$G_k(\theta) = \begin{cases} \alpha_K \left[\cos\left(\theta - \frac{\pi k}{K}\right)\right]^{K-1}, & \left|\theta - \frac{\pi k}{K}\right| < \frac{\pi}{2} \\ 0, & \text{otherwise,} \end{cases}$$

(Source : Simoncelli and Freeman [1])

where r, Θ are polar frequency coordinates.

Unlike conventional orthogonal wavelet decompositions, the subsampling does not produce aliasing artifacts, as the support of the low pass filter *L(r,Θ)* obeys the Nyquist sampling criterion. For all examples in the report, we have used K = 4 orientation bands, and N = 4 pyramid levels(scales).

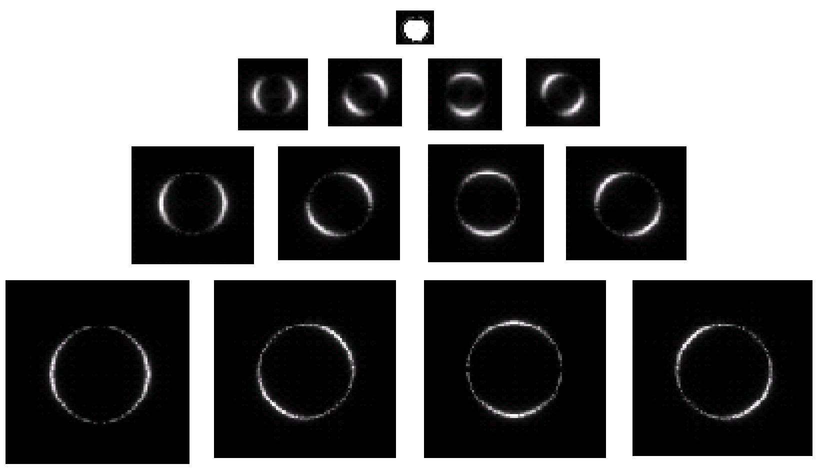

Figure 3: *A 3 scale, 4 orientation steerable pyramid representation of a disk image*

2.2 Statistical Constraints

Assuming one has decomposed an image using a set of linear filters, the constraint functions may be defined on the coefficients of this decomposition. The procedure for choosing the constraint functions is driven by perceptual criteria rather than information theoretic criteria. The algorithm for developing the constraint set is given below:

Algorithm 1: Expansion of Constraint Set

1 Initially choose a set of basic parameters and synthesize texture samples using a large library of examples.

2 Gather examples of synthesis failures, and classify them according to the visual features that distinguish them from their associated original texture examples.

3 Choose a new statistical constraint that captures the visual feature most noticeably missing from the failure group. Incorporate this constraint into the synthesis algorithm.

4 Verify that the new constraint achieves the desired effect of capturing that feature by re- synthesizing the textures in the failure group.

5 Finally verify all the constraints for redundancy.

Below, a description of the constraints obtained in the above way are given, along with a demonstration showing their necessity.

A. Marginal Statistics:

These consists of the image pixel statistics like mean, variance, skewness, kurtosis and range, along with the skewness and kurtosis of the partially reconstructed lowpass images computed at each level of the recursive pyramid decomposition. These statistics are important as they express the relative amount of each intensity in the texture. Fig 4A demonstrates the necessity of these pixel domain statistics.

B. Coefficient Correlation:

These comprise the central samples of the autocorrelation of the partially reconstructed low pass images including the lowpass band. These characterize the salient spatial frequencies and the periodic structure of the texture. Fig 4B demonstrates the necessity of these raw coefficient correlation functions.

C. Coefficient Magnitude Statistics:

These comprise the central samples of the autocorrelation of magnitude of each subband, cross correlation of each subband magnitudes with those of other orientations at the same scale and the cross correlation of subband magnitudes with all orientations at a coarser scale. These capture the important structural information about the textures. These correlations are often present despite the fact that the raw coefficients may be uncorrelated. This can occur due to variations in phase across the image leading to cancellation. Fig. 4C demonstrates the necessity of these parameters.

D. Cross Scale Phase Statistics:

These comprise the cross correlation of the real part of coefficients with both the real and imaginary parts of the phase doubled coefficients at all orientations at the next coarser scale. These coefficients help in distinguishing edges from lines, and help in representing gradients due to shading and lighting effects. For example, a dark line on a light background cannot be distinguished from a white line on a dark background without these coefficients. Fig 4D explains this with an example.

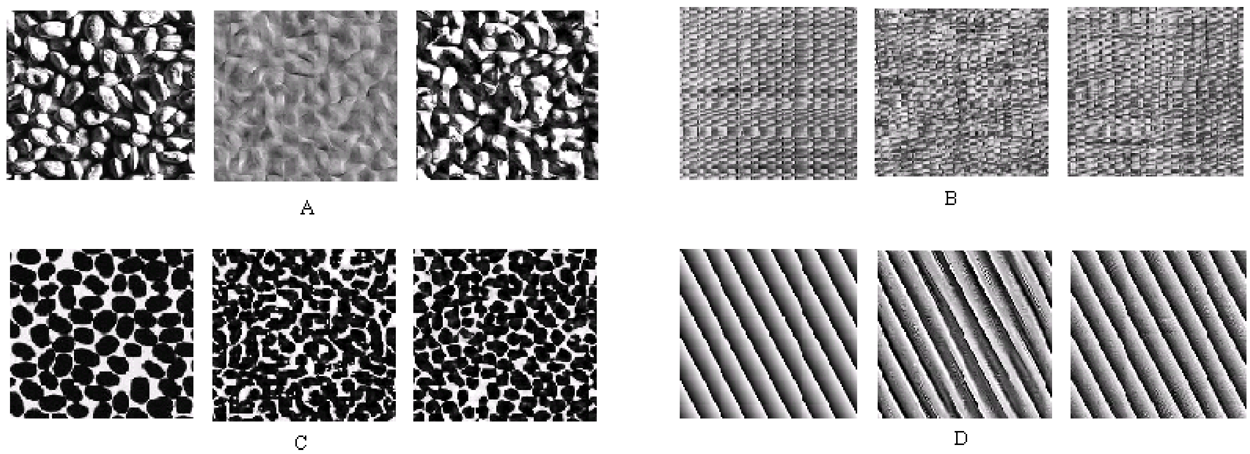

Figure 4. *Illustration of the necessity of each of the statistics discussed above*
Left column: Original Textures, Middle column: Images synthesized with partial set of constraints, Right column: Images synthesized with the full constraint set

4. TEXTURE SYNTHESIS

The need for a texture synthesis algorithm is to facilitate the checking of sufficiency of the statistics that are chosen. This is done using the synthesis by analysis algorithm described earlier.

The texture synthesis algorithm should take statistical constraints extracted during analysis as inputs and produce texture images with these statistics. These required statistics are otherwise known as constraints. This is because the synthesized image is *constrained* to have the required statistical properties.

4.1 Projection onto Constraint Surfaces

To synthesize the image with the required statistical property, we start with a noisy image which is generated from a Gaussian distribution. The mean and the variance of the generating Gaussian distribution is set to the mean and variance of the input texture which were extracted during analysis.

Now to synthesize the image, we essentially need to project this noisy image onto a subspace spanned by textures possessing the required statistical properties. However finding the projection operator becomes very difficult due to the large number of constraint functions and also the individual complexity of each of these functions.

Given this difficulty, an alternate approach is used. In this approach the constraints are imposed one after another in a sequential manner rather than simultaneously imposing them. This clearly reduces the problem complexity, because we now need to impose only one constraint at any given time.

However this leads to a new problem. Sequentially imposing the constraints does not guarantee convergence. To maximize chances of convergence, every time a constraint is imposed, we would like to do so while changing the image as little as possible. To do so, we use the method of gradient projections.

In this method, first the gradient of the particular statistical function is found. Then any changes to the image are made in the direction of the gradient. Since the gradient gives the direction of maximum change, by changing the image along the direction of the gradient, we ensure that the image is changed as little as possible while imposing the constraint. The exact algorithm is listed below.

Algorithm 3: Texture Synthesis

Input: *The set of statistical parameters extracted from the input image during analysis, Number of iterations.*

1 Begin with an image containing samples of white Gaussian noise.

2 Decompose the image using Steerable Pyramid decomposition.

3 Impose statistics extracted during analysis in a sequential manner as follows:

a Adjust the Auto-Correlation, Skewness, Kurtosis of the residual lowband.
b Impose Magnitude Correlation statistics on all the subbands sartting from the coarsest scale to the finest scale
c Build the partially reconstructed lowpass images and impose Coefficient Correlation statistics on them
d Impose variance constraint on the highband
e Reconstruct the image from these modified reconstructed lowpass images and the highpass image.
f Impose Image Pixel statistics on the reconstructed image

4 Decompose the reconstructed image once again repeat the above sequence of steps (a-f). Do this for the required number of iterations

Fig. 5 gives the high level block diagram for the texture synthesis algorithm.

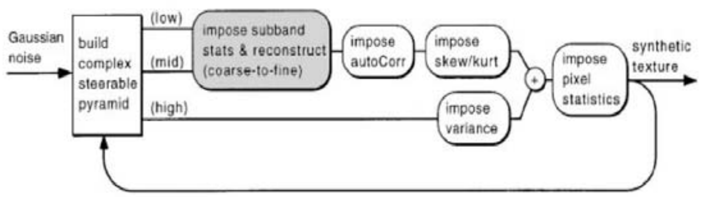

Figure 5. High level block diagram illustrating the testure synthesis algorithm.

(*Source: Simoncelli and Portilla [1]*)

5. RESULTS

5.1 Progress of Texture Synthesis over iterations

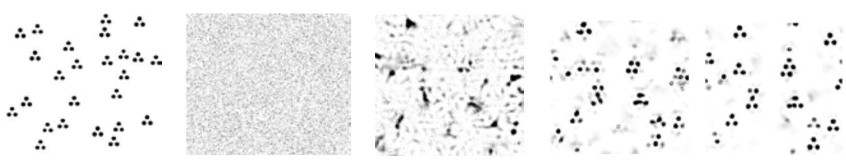

Figure 6. *Progress of texture synthesis algorithm over iterations for 0, 1 , 5, 25 iterations. The left image is the original texture.*

5.2 Examples of successful texture synthesis

Figure 7. *Examples of successful texture synthesis. The original and synthesized textures are shown for an artificial texture(Left) and natural texture(Right) .*

5.3 Examples of failed texture analysis

Figure 8. *Examples of failed texture synthesis. The original and synthesized textures are shown for an homogeneous texture(Left) and inhomogeneous texture(Right) .*

Fig. 6 shows the progress of the algorithm over iterations. The image at iteration 0 is the noisy Gaussian image. Fig. 7 illustrates successful synthesis of a natural and artificial texture. Fig. 8 illustrates two examples of failed synthesis. In the case of the homogeneous texture, the statistics fail to capture the fact that while there are lines of different orientations, each individual line is itself straight and not curved. This indicates that the statistics chosen are not sufficient. In the case of the inhomogeneous texture, the algorithm fails badly simply because the texture is inhomogeneous to begin with.

6. CONCLUSIONS

In this work, a set of statistical functions have been chosen to extract features from textures. A synthesis algorithm has been developed that synthesizes images with the required statistical properties. The algorithm has been successful in synthesizing a large number of a variety of textures – both natural and synthetic. Despite this, there are two obvious issues with the above algorithm that could be improved.

6.1 Choice of Constraint Functions

The set of statistical functions is not the complete set of constraint functions predicted by the Julezs conjecture. This is because the statistical functions used are inadequate for synthesizing certain kind of textures as seen in the examples.

Furthermore, the constraints used have been determined through observations and reverse-engineering as detailed in Sec.1 star. Hence there is no guarantee that the chosen statistical function set is unique. One can always come up with other alternatives that could perform equally as well or perhaps better.

However, the current set is good enough to distinguish and synthesize a large set of textures and thus is a good descriptor of textures.

6.2 Issues with Convergence

The synthesis algorithm described above is not guaranteed to converge. However, because gradient projections have been used, it has been found that the algorithm has almost always converged to an texture that is indistinguishable from the original texture.

7. EXTENSIONS

7.1 Extending a Texture

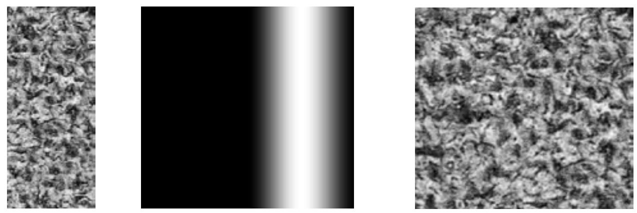

Figure 9. *Extension of texture: Left: Original Texture; Middle: Mask; Right: Extended texture*

Features are extracted from the given texture. Then the image is extended by using a binary mask. The mask should have the same size as the image to be synthesized. During synthesis, at the end of each iteration, the pixel locations of the synthesized image are reset to the original textures values at the pixel locations where the mask is white. Fig. 9 illustrates extension of a 128 x 256 pixel texture to 256 x 256 pixels.

7.2 Image Repair

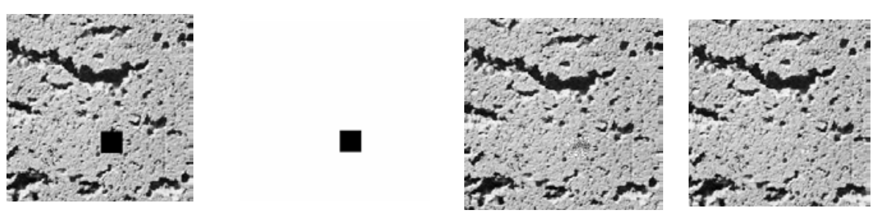

Figure 10. *Repairing a damaged texture. Left: Damaged Texture; Middle: Binary Mask; Right: Repaired images after the 5th and 25th iterations.*

Again a mask is used in this case. Here the mask has the same dimensions as the image to be repaired. The mask is made black in that portion of the image which has a defect, while the rest of the mask is made white. This ensures that a new texture is synthesized only in the region where there is a defect in the image. Fig. 10 illustrates repair of a texture of plaster.

7.3 Painting Texture onto Image

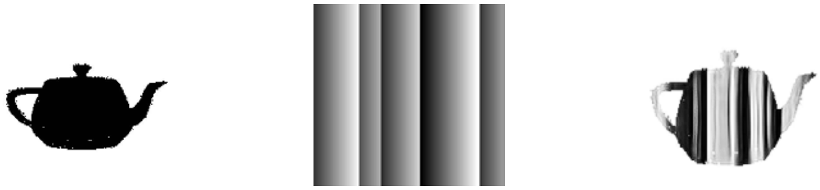

Figure 11. *Painting texture onto a teapot. Left: Binary Mask of teapot; Middle: Texture; Right: Painted teapot*

The idea used is similar to the one used in the previous example. In this case the required image is converted into a binary mask. In the mask, the required image is made black, and the rest of the mask is made white. This ensures that the texture is painted only on to the image. Fig. 11 illustrates this.

7.4 Combining Texture

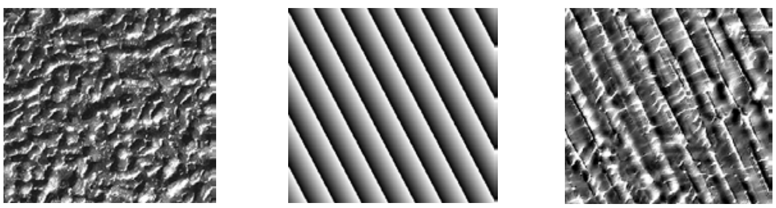

Figure 12. *Combining textures. Left: Metal Texture; Middle: Sawtooth Texture; Right: Combined Texture*

Features are extracted from two different textures and then averaged. These new set of statistical parameters are used to synthesize a texture that is a cross between the two input textures. Fig. 12 illustrates this for the case of a metal and sawtooth texture.

8. REFERENCES

[1] A parametric texture model based on Joint Statistics of Complex Wavelet Coefficients, Simoncelli and Portilla, *International Journal of Computer Vison 2000*

[2] Texture Characterization via Joint Statistics of Complex Wavelet Coefficient Magnitudes, Simoncelli and Portilla, *IEEE Workshop on Statistical and Computational Theories of Vision,1998*

[3] A Filter design techinque for Steerable Pyramid Image Transforms, Simoncelli et al. *IEEE Int'l Conf on Image Processing, 1997*