ESANN'2001 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 25-27 April 2001, D-Facto public., ISBN 2-930307-01-3, pp. 211-216

Texture Analysis with the Volterra Model using Conjugate Gradient Optimisation

Adam I. Wilmer[†], Tania Stathaki[‡], Steve R. Gunn[†], R. I. Damper[†]

[†]Department of Electronics and Computer Science,
University of Southampton, Southampton SO17 1BJ, UK

[‡]Department of Electrical and Electronic Engineering,
Imperial College, London SW7 2BT, UK

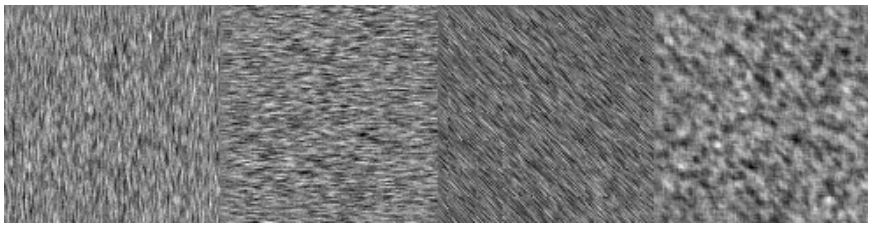
Abstract. Texture is an important characteristic differentiating objects or regions of interest in an image. A variety of approaches to texture analysis has previously been proposed; the approach in this paper is from the stochastic field, whereby a two-dimensional Volterra model is used to represent a texture field. Statistical moments are introduced as a means of determining the Volterra model generator for an unknown texture, with a view to using this model subsequently for recognition. However, an overdetermined set of equations results. These can be solved in the least squares sense using a conjugate gradient descent method with multiple restarts.

1. Introduction

Texture analysis is an important issue within the wider topic of image processing because of the variety of uses to which it can be applied e.g. content-based image recognition, remote sensing, medical diagnosis, quality control, advanced image compression and weather forecasting. It is a difficult task because of the non-uniformity that exists in typical images. Perspective, shade, orientation and scale can give otherwise identical textures different appearances and can pose problems for practical texture analysis. It has been suggested [2] that texture research can be segregated into five categories: structural, statistical, spectral, stochastic and morphological. This paper is concerned with the stochastic approach and the Volterra filter in particular.

The Volterra model and its application to texture analysis are described in Section 2. The model can be described as a power series with memory [3] and can account for nonlinearities in system identification tasks through the inclusion of higher-order terms. An added motive for a model-based approach is that texture synthesis becomes possible, so allowing 'seamless' texture mapping in object modelling. Previous work [5] using a Volterra model for texture analysis has shown promising results. Section 3 describes simulation results, giving an insight into the potential of this approach before Section 4 concludes.

ESANN'2001 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 25-27 April 2001, D-Facto public., ISBN 2-930307-01-3, pp. 211-216



(a) (b) (c) (d)

Figure 1: Examples of texture generated by the first-order Volterra model: (a) vertical texture generated by a (1×10) grid of coefficients; (b) horizontal texture generated by a (10×1) grid; (c) texture generated by a (10×10) diagonal grid; (d) texture generated by a (10×10) non-diagonal grid.

2. The Volterra Model and Texture Analysis

A texture image can be described by a 2-D pth order Volterra system [5],

$$y[m,n] = \sum_{i=1}^{p} H_i\{x[m,n]\},$$

where $x[m,n]$ is driving noise, $H_c\{...\}$ is the cth-order Volterra operator, and

$$H_c\{x[m,n]\} = \sum_{\substack{i_1,j_1,\ldots,\\ i_c,j_c}} h^c_{(i_1,j_1),\ldots,(i_c,j_c)} x[m-i_1,n-j_1]...x[m-i_c,n-j_c],$$

where, $h^c_{(i_1,j_1),\ldots,(i_c,j_c)}$ are the cth-order Volterra coefficients associated with the model. In the following, we restrict ourselves to second-order models ($c = 2$) because they are capable of handling nonlinearities without over-parameterising the problem. Also, this model only requires the use of moments up to third-order which is advantageous as moments become increasingly unreliable as their order increases. A second-order model is given by,

$$\begin{aligned} y[m,n] = &\sum_{i_1,j_1} h^1_{(i_1,j_1)} x[m-i_1,n-j_1] + \\ &\sum_{i_1,j_1,i_2,j_2} h^2_{(i_1,j_1),(i_2,j_2)} x[m-i_1,n-j_1]x[m-i_2,n-j_2], \end{aligned} \tag{1}$$

where $h^1_{(i_1,j_1)}$, $h^2_{(i_1,j_1),(i_2,j_2)}$ are the linear and quadratic coefficients respectively. Zero mean images are used to simplify the moment expressions: Those with a mean value other than zero can be accommodated by subtracting this from every pixel. To constrain the Volterra model to be zero mean, the condition $\sum_{i,j} h^2_{(i,j),(i,j)} = 0$ must hold, which can be enforced by letting $h^2_{(i,j),(i,j)} = 0$ for all i, j. Figure 1 shows some examples of texture generated by the first-order Volterra model.

ESANN'2001 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 25-27 April 2001, D-Facto public., ISBN 2-930307-01-3, pp. 211-216

The coefficient values of (1) can be determined from a set of nonlinear equations [6], formed by equating the second- and third-order moments,

$$\begin{aligned} R[k,l] &= E\{y[m,n]y[m+k,n+l]\} \\ M[(k_1,l_1),(k_2,l_2)] &= E\{y[m,n]y[m+k_1,n+l_1]y[m+k_2,n+l_2]\}, \end{aligned} \tag{2}$$

with that of the Volterra model. The consideration of moments greater than third-order is an issue that will need to be addressed more fully in the future. Applying the second-order moment expression to the second-order Volterra expression of (1) results in an estimate for the second-order moment given by,

$$\begin{aligned} \hat{R}[k,l] = \sum_{i_1,j_1} h^1_{(i_1,j_1)} E\{x[m-i_1,n-j_1]y[m+k,n+l]\} + \\ \sum_{i_1,j_1,i_2,j_2} h^2_{(i_1,j_1),(i_2,j_2)} E\{x[m-i_1,n-j_1]x[m-i_2,n-j_2]y[m+k,n+l]\}. \end{aligned} \tag{3}$$

By constraining the driving noise, $x[m,n]$, to be zero-mean and Gaussian in equation (3) involves averaging over products of Gaussian random variables. It is well known that if $x_1,\ldots,x_4$ are zero-mean and jointly Gaussian then,

$$\begin{aligned} E\{x_1x_2x_3\} &= 0 \\ E\{x_1x_2x_3x_4\} &= E\{x_1x_2\}E\{x_3x_4\} + E\{x_1x_3\}E\{x_2x_4\} + E\{x_1x_4\}E\{x_2x_3\}. \end{aligned}$$

If these random variables are further constrained to be stationary and white, then averaging over the product of two of these random variables results in

$$E\{x[i_1,j_1]x[i_2,j_2]\} = \beta.\delta[i_1-i_2,j_1-j_2], \tag{4}$$

where β is the driving noise variance and $\delta[m,n]$ is the 2-D Kronecker delta function. This choice of zero mean Gaussian driving noise allows significant simplification of the second-order moment expression (3). By applying (4),

$$\begin{aligned} \hat{R}[k,l] = \beta \sum_{i_1,j_1} h^1_{(i_1,j_1)} h^1_{(i_1+k,j_1+l)} + \\ 2\beta^2 \sum_{i_1,j_1,i_2,j_2} h^2_{(i_1,j_1),(i_2,j_2)} h^2_{(i_1+k,j_1+l),(i_2+k,j_2+l)}. \end{aligned}$$

Similarly, estimates for the third-order moments of the system are given by,

$$\begin{aligned} \hat{M}[(k_1,l_1),(k_2,l_2)] = 2\beta^2(\phi_1[(k_2,l_2),(k_2-k_1,l_2-l_1)] + \\ \phi_1[(k_1,l_1),(k_1-k_2,l_1-l_2)] + \phi_1[(-k_1-k_2),(-l_1-l_2)]) + \\ 8\beta^3\phi_2[(k_2,l_2),(k_2-k_1,l_2-l_1),(k_1,l_1)], \end{aligned}$$

where,

$$\begin{aligned} \phi_1[(k_1,l_1),(k_2,l_2)] = \sum_{i_1,j_1,i_2,j_2} h^1_{(i_1+k_1,j_1+l_1)} h^1_{(i_2+k_2,j_2+l_2)} h^2_{(i_1,j_1),(i_2,j_2)} \\ \phi_2[(x_1,y_1),(x_2,y_2),(x_3,y_3)] = \\ \sum_{i_1,i_2,i_3,j_1,j_2,j_3} h^2_{(i_1,j_1),(i_2,j_2)} h^2_{(i_1+x_1,j_1+y_1),(i_3+x_2,j_3+y_2)} h^2_{(i_2+x_3,j_2+y_3),(i_3,j_3)}. \end{aligned}$$

ESANN'2001 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 25-27 April 2001, D-Facto public., ISBN 2-930307-01-3, pp. 211-216

Coeff. Neighbourhood	No. of Coefficients	No. of Moment Equations
1×1	1	1
3×2	6	8
4×4	16	25
$i\times j$	$i\times j$	$2ij-i-j+1$

Table 1: A comparison of the number of Volterra coefficients with the number of corresponding non-zero moment expressions.

To recover the model parameters, the differences in the model and texture moments are minimised. The moment expressions form an overdetermined set, as illustrated by Table 1, which shows that the problem is overdetermined by a factor approaching two as the size of the model increases. A least squares approach is adopted to solve the set of overdetermined equations,

$$\begin{aligned}\min_{\beta,h} J(f) = &\sum_{i,j}(R[i,j]-\hat{R}[i,j])^2 + \\ &\sum_{i_1,j_1,i_2,j_2}(M[(i_1,j_1),(i_2,j_2)]-\hat{M}[(i_1,j_1),(i_2,j_2)])^2.\end{aligned} \tag{5}$$

This cost function is minimised in an effort to locate the global minimum corresponding to the Volterra model parameter values that most closely describe the unknown texture. The cost surface described by (5) will typically contain many local minima (especially for a texture described by many coefficients) and, consequently, a pure gradient descent technique will fail to locate the global solution. To achieve speed and accuracy in locating the global solution, a multiple start point, scaled conjugate gradient method is employed [4, 1]. Applied to a real texture, the parameters corresponding to the global solution may be used as features for recognition.

3. Optimisation and Simulation Results

Three types of synthetic texture image were generated using first-order Volterra models of the following forms,

$$\begin{aligned}
y_1[m,n] &= h^1_{(0,0)}x[m,n]+h^1_{(0,1)}x[m,n-1]+h^1_{(1,1)}x[m-1,n-1]\\
y_2[m,n] &= h^1_{(0,0)}x[m,n]+h^1_{(0,1)}x[m,n-1]+h^1_{(1,0)}x[m-1,n]+\\
&\quad h^1_{(1,1)}x[m-1,n-1]+h^1_{(0,2)}x[m,n-2]\\
y_3[m,n] &= h^1_{(0,0)}x[m,n]+h^1_{(0,1)}x[m,n-1]+h^1_{(1,0)}x[m-1,n]+\\
&\quad h^1_{(1,1)}x[m-1,n-1]+h^1_{(0,2)}x[m,n-2]+h^1_{(1,2)}x[m-1,n-2].
\end{aligned}$$

An ensemble of 10 textures is generated using each model and the relevant moments obtained. Conjugate gradient descent (using start points determined

ESANN'2001 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 25-27 April 2001, D-Facto public., ISBN 2-930307-01-3, pp. 211-216

Parameter	Actual Value	Mean Estimate	Variance of Estimate
$h^1_{(0,0)}$	1.5	1.4983	2.37×10^{-4}
$h^1_{(0,1)}$	0.8	0.8028	9.60×10^{-5}
$h^1_{(1,1)}$	2.1	2.1034	1.10×10^{-4}

Table 2: Conjugate gradients applied to the three coefficient texture, $y_1[m,n]$.

$h^1_{(0,0)}$	$h^1_{(0,1)}$	$h^1_{(1,0)}$	$h^1_{(1,1)}$	$h^1_{(0,2)}$	Cost	% Located
1.026	1.661	0.6261	2.057	0.436	5.54×10^{-3}	61
1.654	0.793	1.914	1.227	0.221	6.01×10^{-2}	39

Table 3: Conjugate gradients applied to the five coefficient texture, $y_2[m,n]$, with $\{h^1_{(0,0)}, h^1_{(0,1)}, h^1_{(1,0)}, h^1_{(1,1)}, h^1_{(0,2)}\} = \{1.0, 1.6, 0.7, 2.1, 0.4\}$.

from a uniform distribution between 0.0 and 5.0) was then applied to recover parameter estimates for each case and these are summarised in Tables 2, 3 and 4.

Second- and third-order moment values are determined from the synthetic image by applying expressions (2) at each pixel in the image and using a neighbourhood of surrounding pixels for the calculation. The texture images generated have dimensions of 256×256 pixels and a neighbourhood size of 32×32 was used to calculate moments. The means of the moment values across the image were then used as final estimates.

For $y_1[m,n]$, the estimated parameter values obtained (Table 2) are similar to the original values used in generating the texture. The conjugate gradient method consistently finds the correct parameters with a low variance, suggesting that the basin of attraction is large. This simple example validates the approach outlined in this paper.

Table 3 summarises the parameter estimates from applying the technique to $y_2[m,n]$, where two minima are found. The estimate corresponding to the parameter values used to synthesise the texture is located correctly the majority of the time and has a lower cost than the alternative solution.

The experiments were repeated for a more complex example, defined by $y_3[m,n]$, resulting in the estimates summarised in Table 4. Here, the correct estimate (row 2) is located with multiple start points. Our optimisation method

$h^1_{(0,0)}$	$h^1_{(0,1)}$	$h^1_{(1,0)}$	$h^1_{(1,1)}$	$h^1_{(0,2)}$	$h^1_{(1,2)}$	Cost	% Located
1.201	2.141	0.368	1.549	0.757	1.083	9.53×10^{-6}	41.7
0.998	1.601	0.697	2.098	0.400	1.305	2.67×10^{-10}	8.3
1.300	2.099	0.400	1.601	0.698	0.998	6.13×10^{-9}	27.8
1.083	1.549	0.757	2.141	0.368	1.200	9.54×10^{-6}	22.2

Table 4: Conjugate gradients applied to the six coefficient texture, $y_3[m,n]$, with $\{h^1_{(0,0)}, h^1_{(0,1)}, h^1_{(1,0)}, h^1_{(1,1)}, h^1_{(0,2)}, h^1_{(1,2)}\} = \{1.0, 1.6, 0.7, 2.1, 0.4, 1.3\}$.

ESANN'2001 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 25-27 April 2001, D-Facto public., ISBN 2-930307-01-3, pp. 211-216

can suffer from problems common to gradient descent methods; a more complex texture would pose a harsher test. As more coefficients (or a higher-order model) are used, the cost surface becomes more complex and a greater number of start points will typically be required.

4. Conclusions

This paper shows how the Volterra model can be used for texture analysis. The parameters of the model are recovered by minimising the differences between model and texture image moments. The resulting optimisation problem is typically non-convex and this paper proposes a scaled conjugate gradients method using multiple restart points. In this way, a hypothetical Volterra model generator for an unknown texture can be fitted and the parameter estimates used as features for recognition or fed back into the model for synthesis. Initial results show that the approach is promising for first-order models. Further investigations are required to validate the approach for higher-order models. As the work is in an investigatory phase, application to real texture has not yet taken place although this is a priority for the future.

At this stage of the work, convergence to a global minimum cannot be guaranteed. Future work will investigate alternative models to simplify the optimisation problem without sacrificing the virtues of the representation.

References

[1] C. M. Bishop. *Neural Networks for Pattern Recognition.* Clarendon Press, Oxford, UK, 1995.

[2] Y. Q. Chen, M. S. Nixon, and D. W. Thomas. On texture classification. *International Journal of Systems Science*, 28(7):669–682, 1997.

[3] M. Schetzen. *The Volterra and Wiener Theories of Nonlinear Systems.* Krieger, Malabar, FL, 1980.

[4] J. R. Shewchuk. An introduction to the conjugate gradient method without the agonizing pain. Technical report, School of Computer Science, Carnegie Mellon University, Pittsburg, PA, 1994.

[5] T. Stathaki and A. Constantinides. Blind adaptive Volterra system identification using barrier function methods for constrained optimisation. In *Proceedings of 31st Asilomar Conference on Signals, Systems and Computers*, pages 3–7, Pacific Grove, CA, 1998.

[6] T. Stathaki and A. Scohyers. A constrained optimisation approach to the blind estimation of Volterra kernels. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'97*, pages 2373–2376, Munich, Germany, 1997.