Texture Segmentation by Evidence Gathering

Ben Waller
bmw306@ecs.soton.ac.uk
Mark Nixon
msn@ecs.soton.ac.uk
John Carter
jnc@ecs.soton.ac.uk

School of Electronics and Computer Science
University of Southampton
Southampton, UK

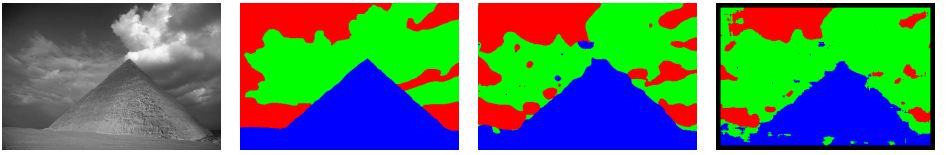
Abstract

A new approach to texture segmentation is presented which uses Local Binary Pattern data to provide evidence from which pixels can be classified into texture classes. The proposed algorithm, which we contend to be the first use of evidence gathering in the field of texture classification, uses Generalised Hough Transform style R-tables as unique descriptors for each texture class and an accumulator is used to store votes for each texture class. Tests on the Brodatz database and Berkeley Segmentation Dataset have shown that our algorithm provides excellent results; an average of 86.9% was achieved over 50 tests on 27 Brodatz textures compared with 80.3% achieved by segmentation by histogram comparison centred on each pixel. In addition, our results provide noticeably smoother texture boundaries and reduced noise within texture regions. The concept is also a "higher order" texture descriptor, whereby the arrangement of texture elements is used for classification as well as the frequency of occurrence that is featured in standard texture operators. This results in a unique descriptor for each texture class based on the structure of texture elements within the image, which leads to a homogeneous segmentation, in boundary and area, of texture by this new technique.

1 Introduction

Texture is an important property of images, representing the structural and statistical distribution of elements throughout the image. Images can contain a single texture, for example an image of a brick wall, or multiple textures of varying distribution throughout the image such as a satellite image containing textures representing urban areas, fields, forest and water. Image segmentation by texture has a wide range of applications, from analysis of medical images [7] to remote sensing [8]. Additionally there are industrial applications of texture analysis which include visual inspection and defect detection [11].

Texture descriptors can be divided into two types; structural and statistical. Structural approaches apply a transform, such as the Fourier transform, to the image and then obtain a set of measurements which describe the texture [13]. Statistical approaches classify textures by measuring a property of the image and comparing the rate of occurrence of this to that obtained from training images. A well-known example of this is the co-occurrence matrix



(a) (b) (c) 90.2% (d) 85.6%

Figure 1: BSDS Pyramid: a) original image; b) manual segmentation; c) segmentation using the EGTS algorithm; d) segmentation using histogram comparison.

[5], developed by Haralick *et al*. in 1973, where the number of pairs of pixels separated by a particular distance with a specific intensity are counted. The matrix of number of pairs is used as the texture descriptor for classification. Another popular and more modern operator is Local Binary Patterns (LBP) which uses the intensity at a point to threshold surrounding pixels to produce a code representing the texture pattern at that point [16]. A histogram of the texture codes is used as the texture descriptor. Both operators are well established and the LBP has continued to receive significant attention over the years with many published extensions and applications [2, 4].

Texture classification typically relies on using a measure of similarity between a texture sample and known texture classes to classify the sample. Segmentation is usually performed either by classification of each pixel separately via a windowing method [10] or by an iterative split and merge algorithm [14]. Both methods are computationally inefficient since they require the same information to be processed multiple times.

Unlike texture, the field of template matching in computer vision has benefited from the use of evidence gathering approaches, most notably present in the Hough Transform, which accumulates votes for each pixel every time evidence indicating the presence of a desired shape at that point is found [6]. The Hough Transform was extended further to accommodate many different shapes such as circles and ellipses, and a generalised form of the Hough Transform was developed to be able to search the image for any arbitrarily shaped object [1]. The advantages of this evidence gathering method include scale and rotation invariance and resilience to noise and occlusion.

We describe a new approach to texture segmentation which determines texture class through evidence gathering. The Local Binary Pattern (LBP) operator is used as the mechanism to gather evidence and Generalised Hough Transform (GHT) style R-table and accumulators are employed to store this evidence and vote accordingly. This new approach is the first use of evidence gathering to determine texture and has been demonstrated to give very good results for texture segmentation, as illustrated by Figure 1, while maintaining smooth texture boundaries and minimising noise. The proposed algorithm will be referred to henceforth as the Evidence Gathering Texture Segmentation (EGTS) algorithm. To show the advantages of this new technique our evaluation compares results from EGTS with a pure LBP algorithm known as histogram comparison. This demonstrates that using evidence gathering in conjunction with an established texture descriptor can yield improved segmentation accuracy.

Section 2 summarises the existing work on texture classification and evidence gathering that is used as the basis for this new method and Section 3 describes the new EGTS algorithm. Section 4 provides experimental results and Section 5 concludes the paper.

2 Background

2.1 Local Binary Patterns

Local Binary Patterns are texture descriptors which label individual pixels in an image with a code corresponding to the local texture pattern surrounding the pixel. First introduced by Ojala *et al*. [15], the earliest form of the LBP used the centre pixel of a 3x3 grid, g_c, to threshold each of the eight neighbouring pixels g_0 to g_7. This produced an eight bit binary code which represents the texture element present at that point. The LBP was later extended to give the texture pattern for P points on a circle of radius R. It was observed that certain fundamental patterns make up the majority of all LBP patterns observed [16]. These were found to be the patterns which had at most two zero to one transitions ($U(LBP_{P,R}) \leq 2$) and are called *uniform* LBP patterns. All of the uniform patterns are labelled according to the number of '1' bits in the code. When P is equal to eight, there will be ten different patterns: the uniform patterns from '0' to '8' and the pattern '9' which is the agglomeration of all other patterns. Since the sampling positions for the neighbouring points are arranged in a circle, the uniform LBP is, by nature, rotation invariant. This is because if the sample image texture is rotated, the LBP code produced will still have the same number of zero to one transitions and the same number of '1' bits, resulting in an identical uniform LBP code regardless of the order of the bits. The rotation invariant uniform LBP code for a point, $LBP^{riu2}_{P,R}$, is calculated by:

$$LBP^{riu2}_{P,R} = \begin{cases} \sum_{p=0}^{P-1} s(g_p - g_c) & \text{if } U(LBP_{P,R}) \leq 2 \\ P+1 & \text{otherwise} \end{cases} \tag{1}$$

where

$$U(LBP_{P,R}) = |s(g_{P-1} - g_c) - s(g_0 - g_c)| + \sum_{p=1}^{P-1} \left| s(g_p - g_c) - s(g_{p-1} - g_c) \right| \tag{2}$$

and

$$s(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

Textures can be classified using the LBP by obtaining a histogram of the uniform patterns and using a dissimilarity metric to compare this to histograms obtained from known texture classes. Segmentation can be performed by performing classification on a pixel by pixel basis by obtaining the histogram of a region centred on the pixel.

2.2 Generalised Hough Transform

The Generalised Hough Transform [1] uses an evidence gathering approach to determine the location of previously defined arbitrary shapes within an image. An arbitrary shape can be described by the following set of parameters:

$$\mathbf{a} = \{\mathbf{y}, s, \theta\} \tag{4}$$

where $\mathbf{y} = (x_r, y_r)$ is the reference origin for the shape, s is the scale of the shape and θ is the orientation of the shape. For each edge point $\mathbf{x}$ on the shape the gradient is calculated and the vector $\mathbf{r}$ between $\mathbf{x}$ and $\mathbf{y}$ is stored in a table called the *R-table*. The R-table contains

a series of bins, each representing a range of gradients. The **r** vector for each edge point is stored in the bin that matches the calculated gradient, resulting in the R-table containing a complete description of the shape. The algorithm for using the R-table to find shapes with in an image is described by Ballard [1] as:

"For each edge pixel **x** in the image, increment all the corresponding points **x** + **r** in the accumulator array A where **r** is a table entry indexed by θ, i.e., $\mathbf{r}(\theta)$. Maxima in A correspond to possible instances of the shape S."

3 Evidence Gathering Texture Segmentation

A new evidence gathering approach to texture segmentation is proposed which uses the principles of template matching present in the Generalised Hough Transform (GHT) and modifies it to match texture instead of shape. The technique exploits a property of the Local Binary Pattern (LBP) texture descriptor which is that if there is structure in the image space, there must be structure in the LBP space. Mäenpää and Pietikäinen observed in [9] that each LBP code limits the set of possible codes adjacent to it. This implies that the arrangement of LBP codes within a texture element is not random and that taking a histogram of the codes reduces the available information further to that originally lost in the LBP process. It is possible for several textures to have the same histogram, rendering such methods incapable of distinguishing between them. By storing the LBP code along with its offset to the centre of the texture region for each pixel, this structural information is not lost and a unique descriptor is produced which can be used in the classification and segmentation of images. The descriptor is unique because it can be used to regenerate the array of LBP codes that represent the texture sample, unlike a histogram of LBP codes which cannot. The new algorithm is thought to be the first use of evidence gathering in texture segmentation and achieves high efficiency by transferring the principles of low computational complexity present in the GHT method to the field of texture analysis.

3.1 Method

As with the GHT, before sample images can be analysed, an R-table must be generated for each known texture class. This describes the structure and composition of a section of the texture and is used to classify the texture class of the sample images. Sub-images, or cells, are taken from the training images and the LBP code is calculated for each pixel within the cell. The R-table contains a number of bins equal to the number of different LBP codes that exist for the version of the LBP that is being used. For LBP P values of eight, the number of bins will be ten; one for each of the nine uniform LBP codes and a miscellaneous bin for all other codes which are not classified as one of the uniform patterns. For each pixel in the cell an entry is submitted to the bin corresponding to the LBP code for that pixel. The entry is a two dimensional vector $\mathbf{r}=(x_r,y_r)$ representing the translation from the pixel to the reference point of the cell. This reference point is usually chosen to be the centre. In Figure 2, the top left pixel (shown in red) in the cell has an LBP code of '1' and so an entry is made in the '1' bin with the vector (2,2) which maps the top left pixel to the centre. The size and number of cells taken from the training images are not fixed and these parameters can be tailored for different applications. The size of the cell should be large enough to contain at least one full example of the repeating pattern in the texture. Having multiple cells for each texture class will provide more evidence for classification during the segmentation process.

1	2	3	6	2
5	3	3	3	3
3	2	3	6	3
4	3	5	7	5
2	5	5	8	6

(a) Cell

Bin Number	Entries
1	(2,2)
2	(1,2)(-2,2)(1,0)(2,-2)
3	(0,2)(1,1)(0,1)(-1,1)(-2,1)(2,0)(0,0)(-2,0)(1,-1)
4	(2,-1)
5	(2,1)(0,-1)(-2,-1)(1,-2)(0,-2)
6	(-1,2)(-1,0)(-2,-2)
7	(-1,-1)
8	(-1,-2)
9	

(b) R-table

Figure 2: Example LBP values for a 5x5 pixel cell and corresponding R-table. The reference point for the cell is shaded in blue.

The following equation is used to calculate the R-table entry for each pixel $\mathbf{x} = (x,y)$ in a cell of centre $\mathbf{c} = (x_c, y_c)$:

$$\mathbf{r} = \mathbf{c} - \mathbf{x} \tag{5}$$

where the R-table index is the LBP code calculated by Equation 1 at the point $\mathbf{x} = (x,y)$.

As with the GHT, evidence is stored in an array called the accumulator, and a separate accumulator is maintained for each of the texture classes that are being searched for. In the segmentation of sample images, the LBP code for each pixel in the entire image is calculated. The entries in the R-tables represent the possible locations of the current pixel relative to the reference point of the cell. For the example in Figure 2, if a pixel in the sample image had an LBP code of '6', it could correspond equally to any of the three positions within the cell also with that LBP code. For each in turn, votes are made for the area that would cover the entire cell positioned on that pixel. Rephrasing Ballard, the algorithm becomes: For each pixel $\mathbf{x}$ in the image, increment all the corresponding points in a cell centred on the point $\mathbf{x} + \mathbf{r}$ in the accumulator array A where $\mathbf{r}$ is a table entry indexed by the LBP code at point $\mathbf{x}$. Maxima in A correspond to possible instances of the texture T.

Voting is done in blocks rather than for individual pixels because texture covers an area and a single pixel on its own does not contain texture. The area covered by each block vote is equal to the area of the cell from which the evidence was gathered. The three block votes for an LBP code of '6' using the R-table in Figure 2(b) are shown in Figure 3. The algorithm is effectively searching the sample image for the texture structure observed in the training cell. In Figure 3, it can be seen that four of the pixels in the image were within all three possible cells for that R-table and hence these pixels have a higher probability of belonging to that texture class. The equations for calculating the coordinates of the four corners of the rectangle covering the voting block for each R-table entry, where the reference point is the centre of the cell, are as follows:

$$\text{Top left} = \mathbf{x} + \mathbf{r} + (-\frac{c_w}{2}, -\frac{c_h}{2}) \tag{6}$$

$$\text{Top right} = \mathbf{x} + \mathbf{r} + (\frac{c_w}{2}, -\frac{c_h}{2}) \tag{7}$$

$$\text{Bottom left} = \mathbf{x} + \mathbf{r} + (-\frac{c_w}{2}, \frac{c_h}{2}) \tag{8}$$

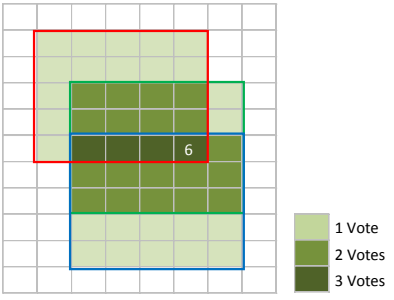


Figure 3: Accumulator showing block votes for three R-table entries, bordered by red, green and blue rectangles.

$$\text{Bottom right} = \mathbf{x} + \mathbf{r} + (\frac{c_w}{2}, \frac{c_h}{2}) \tag{9}$$

where c_w and c_h are the cell width and cell height respectively.

An accumulator for each texture class maintains the number of votes for each pixel for that texture. If there is more than one cell for a texture class, the votes of the subsequent cells are added to the accumulator for the first cell. When the voting process is finished, the higher the number of votes for each pixel, the higher the probability of the pixel belonging to that texture class. It is important to note that analysis of a single pixel yields evidence for many other pixels. This works because if there is structure in the texture, the LBP code at a point is related to those around it. Using a higher number of cells per texture class increases the amount of evidence used to classify pixels and leads to a higher accuracy. Segmentation is performed by filling an accumulator for each texture class and assigning each pixel to the texture class with the highest number of votes at that point.

3.2 Extensions

3.2.1 Multi-scale Support

Multi-scale versions of the LBP operator can be obtained from the individual histograms of the LBP at different scales by extending the measure of dissimilarity to compare over multiple histograms. The multi-scale LBP has been demonstrated to give better results than the single scale version [16]. The EGTS algorithm can be similarly extended to support multiple scales by calculating the votes for each pixel at each scale and then adding them together. In Figure 4(b) it can be seen that not all textures are identified correctly using an LBP radius of 1, however when these results are combined with those obtained from an LBP radius of 2, as seen in Figure 4(c), a vastly improved segmentation is obtained.

3.2.2 Matched Voting

An issue with the original form of the EGTS algorithm is overvoting. Since most modern LBP variants only have ten different codes many votes are made for the wrong texture since

(a) (b) 59.3% (c) 95.5%

Figure 4: Multiscale: a) original image; b) Segmentation results using LBP radius of 1 and nine cells of 32x32 pixels c) Segmentation results using LBP radius of 1 and 2 and nine cells of 32x32 pixels.

(a) (b) 93.8% (c) 95.5%

Figure 5: Matched Voting: a) original image; b) Results using radius of 1 and 2 and nine cells of size 32x32 pixels without using matched voting; c) Results under the same conditions using the matched voting extension.

there will always be an element of overlap in the code occurrence. The LBP methodology still works; there will always be more votes for a perfect sample than for a different texture, however the presence of noise or a slightly distorted texture sample can reduce the contrast of votes between texture classes. A solution to this problem is the matched voting extension. In the standard version of the algorithm the LBP code of the pixel being classified is matched to those of the training cells. However if we revisit the theory of structure present in the LBP space it is apparent that if there is also a match between the LBP codes of the neighbouring pixels in the sample image and the neighbouring pixels in the training cell there is a higher chance of the pixel belonging to that texture class. The matched voting extension awards one extra block vote per correctly matched neighbouring pixel. Tests have shown that allowing the neighbouring LBP codes to match any of the neighbouring codes in the R-table gives the best contrast increase while maintaining the rotation invariant properties and number of votes for correct textures. This means that in the example in Figure 3, the three entries in the R-table will not be treated equally and will be assigned votes dependant on how closely the structure matches. Each R-table entry is now required to contain the LBP codes for the neighbouring pixels as well as the vector from the pixel to the centre of the cell. Figure 5 shows the typical performance increase when matched voting is used instead of standard voting.

3.2.3 Normalisation

It can be observed that different textures have different voting strengths. This means that some textures could give a larger number of votes for an incorrect texture than another texture

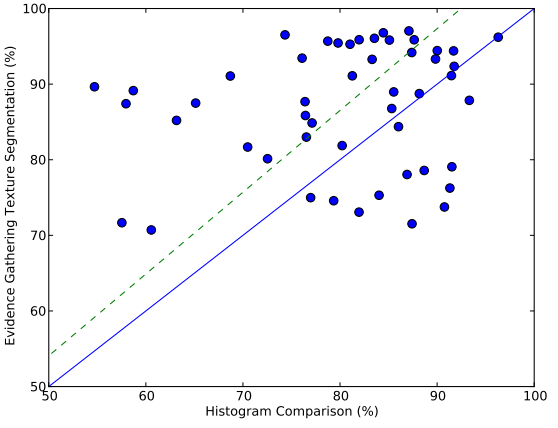


Figure 6: Segmentation accuracy of mosaics from the Brodatz subset using both the new evidence gathering algorithm and the histogram comparison algorithm. The solid blue line represents the line of equality and the dashed green line is the trend line.

could give for a correct match. This leads to cases where votes from one texture overpower those from another, distorting the segmentation results. A solution is to normalise the voting, whereby the votes from each texture are weighted according to their strength factor. One way of calculating the strength factor is to add up the total number of votes for the texture over the entire image and divide by the number of pixels. When all votes for a texture are divided by its strength factor the stronger textures will have their influence over the regions of other textures weakened, reducing the "overspill" effect. The equation for performing normalisation on an accumulator A of size w by h is:

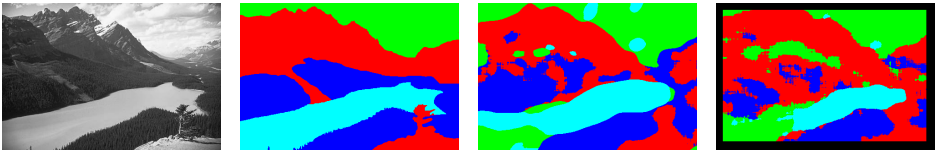
$$A_{norm}(x,y) = \frac{A(x,y) * w * h}{\sum_{a=0}^{w} \sum_{b=0}^{h} A(a,b)} \tag{10}$$

If normalisation is required where one texture is weaker than the others, its use can restore the texture boundaries to their correct locations. Better results can sometimes be obtained from manual assignment of the strength factors, leading us to believe that a machine learning approach is the best way of obtaining the optimum strength factor during the training stage.

4 Results

4.1 Texture Mosaics

A subset of 27 textures from the Brodatz album [3] was used to generate 50 mosaics containing four randomly selected textures. This subset is included in the supplementary material. For each texture in the subset, the bottom right quarter was used to generate the mosaics, and the top left quarter was used to provide training data for segmentation. Segmentation was performed using the EGTS algorithm, employing LBP radii of both 1 and 2 for multi-scale



(a) (b) (c) 76.7% (d) 67.8%

Figure 7: BSDS Mountain: a) original image; b) manual segmentation; c) segmentation using the EGTS algorithm; d) segmentation using the HC algorithm.

support and segmenting using 9 cells of 32x32 pixels each. The matched voting and automatic normalisation features were also enabled. The standard method of image segmentation using a texture classification algorithm classifies each pixel individually by taking a window centred on it and performing comparison against the training data [17]. For comparison, we have used the LBP segmentation from [10] which uses this method to segment each of the 50 texture mosaics. For simplicity, this algorithm will be referred to as Histogram Comparison (HC). Results from the 50 tests are shown in the graph in Figure 6, where the preponderance of results exceeding the line of equality shows the superiority of the new approach. Segmentation accuracy was calculated by comparing the results pixel-by-pixel against the ground truth. Our EGTS algorithm achieved an average segmentation accuracy of 86.9% and standard deviation of 8.12 over the fifty tests compared with an average of 80.3% and standard deviation of 10.36 achieved by HC.

4.2 Real Images

Results from two real images from the Berkeley Segmentation Dataset [12] have been included. The first is an Egyptian pyramid shown in Figure 1. The results obtained from our EGTS algorithm and the standard HC algorithm are shown in Figures 1(c) and 1(d) respectively. A manual segmentation of the image is included in Figure 1(b) and the segmentations are compared to this ground truth to obtain a numerical indicator of their quality which is shown in the captions. Both algorithms provide a good segmentation of the image, however it is apparent that that the EGTS algorithm provides a much smoother boundary between the textures along with a higher segmentation accuracy. The second image is of a mountain scene and results are shown in Figure 7. Our algorithm provides a significantly better result than the HC algorithm and again features smoother boundaries between textures and lower noise within texture segments.

5 Conclusions

In this paper, we have presented a new method for image texture segmentation which we contend to be the first use of an evidence gathering approach in the field of texture analysis. In contrast to conventional methods which compare measurements from a sample of an image to training data to classify a single pixel, our approach compiles information gathered from each pixel into evidence to support the classification of nearby pixels into each known texture class. Each pixel is then classified into the class for which it has the most evidence. We have performed a statistical test using a subset of the Brodatz texture database and our EGTS algorithm gives a higher average performance and lower standard deviation

than the HC algorithm under the same conditions. The lower standard deviation implies that in addition to performing better on average, our algorithm is also more robust. Two tests on real images from the Berkeley Segmentation Dataset show significantly higher segmentation accuracies are obtained from the EGTS algorithm. Our results also provide noticeably smoother texture boundaries and reduced noise within texture regions. The proposed EGTS algorithm is an implementation of a higher order texture descriptor; classifying texture based on the structure of the individual elements which make up the texture. Existing "low order" descriptors use the rate of occurrence of the texture elements to classify the textures, providing a descriptor which is not necessarily unique to a single texture class. By contrast, our EGTS algorithm generates a unique R-table for each texture which not only supplies information on the occurrence of texture elements, but also their structure. Further work will focus on parameter optimisation and developing a colour version of the algorithm.

References

[1] D.H. Ballard. Generalizing the Hough transform to detect arbitrary shapes. *Pattern recognition*, 13(2):111–122, 1981.

[2] HS. Bhatt, S. Bharadwaj, R. Singh, and M. Vatsa. On matching sketches with digital face images. In *Proc. BTAS*, 2010.

[3] P. Brodatz. *Textures: A photographic album for artists and designers*. Dover Publications New York, 1966.

[4] Z. Guo, L. Zhang, D. Zhang, and S. Zhang. Rotation invariant texture classification using adaptive lbp with directional statistical features. In *Proc. ICIP*, 2010.

[5] R.M. Haralick, K. Shanmugam, and I.H. Dinstein. Textural features for image classification. *IEEE TSMC*, 3(6):610–621, 1973.

[6] J. Illingworth and J. Kittler. A survey of the Hough transform. *CVGIP*, 44(1):87–116, 1988.

[7] J. Kontinen, J. Röning, and R. MacKie. Texture features in the classification of melanocytic lesions. In *Proc. ICIAP*, pages 453–460, 1997.

[8] A. Lucieer, A. Stein, and P. Fisher. Texture-based segmentation of high-resolution remotely sensed imagery for identification of fuzzy objects. In *Proc. GeoComputation*, 2003.

[9] T. Mäenpää and M. Pietikäinen. Texture analysis with local binary patterns. *Handbook of Pattern Recognition and Computer Vision*, pages 197–216, 2005.

[10] T. Mäenpää, M. Pietikäinen, and T. Ojala. Texture classification by multi-predicate local binary pattern operators. In *ICPR*, volume 15, pages 939–942, 2000.

[11] T. Mäenpää, M. Turtinen, and M. Pietikäinen. Real-time surface inspection by texture. *Real-Time Imaging*, 9(5):289–296, 2003.

[12] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proc. ICCV*, volume 2, pages 416–423, 2001.

[13] M.S. Nixon and A.S. Aguado. *Feature extraction and image processing*. Academic Press, 2008.

[14] T. Ojala and M. Pietikäinen. Unsupervised texture segmentation using feature distributions. *Pattern Recognition*, 32(3):477–486, 1999.

[15] T. Ojala, M. Pietikäinen, and D. Harwood. A comparative study of texture measures with classification based on featured distributions. *Pattern Recognition*, 29(1):51–59, 1996.

[16] T. Ojala, M. Pietikäinen, and T. Mäenpää. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE TPAMI*, 24(7):971–987, 2002.

[17] M. Petrou and P.G. Sevilla. *Image processing: dealing with texture*. Wiley, 2006.