A unifying framework for LBP and related methods

Francesco Bianconi and Antonio Fernández

Abstract In this chapter we describe a unifying framework for local binary patterns and variants which we refer to as histograms of equivalent patterns (HEP). In presenting this concept we discuss some basic issues in texture analysis: the problem of defining what texture is; the problem of classifying the many existing texture descriptors; the concept of bag-of-features and the design choices that one has to deal with when designing a texture descriptor. We show how this relates to local binary patterns and related methods and propose a unifying mathematical formalism to express them within the HEP. Finally, we give a geometrical interpretation of these methods as partitioning operators in a high-dimensional space, showing how this representation can propound possible directions for future research.

1 Introduction

It is somewhat surprising that, in spite of the wide popularity and the numerous applications of local binary patterns and variants (LBP&V, henceforth), little effort has been devoted to address their theoretical foundations. In this chapter we wish to provide some insight about the rationale behind these methods, hoping that this could help to fill this gap. The overall aim of this chapter is to put LBP&V in the context of related literature and show that such methods actually belong to a wider class of texture descriptors which we refer to as histograms of equivalent patterns (HEP).

Francesco Bianconi
Department of Industrial Engineering, Università degli Studi di Perugia, Via G. Duranti, 67 – 06125 Perugia (Italy), e-mail: bianco@ieee.org

Antonio Fernández
School of Industrial Engineering, Universidade de Vigo, Campus Universitario – 36310 Vigo (Spain) e-mail: antfdez@uvigo.es

Our journey starts with a discussion on some very fundamental issues in texture analysis, such as the definition of texture itself and the problem of establishing a clear and unequivocal classification of the existing texture descriptors. The lack of a formal definition of texture, as well as the uncertainty entailed within the classification frameworks proposed thus far, represent, in fact, serious problems when it comes to deciding whether LBP and related methods belong to one category or another. A closer look at LBP&V reveals that a trait common to them all is the idea of 'bag of features', a concept that proves fundamental to texture analysis in general. In the process of developing a texture descriptor, there are some fundamental issues one has inevitably to deal with: how should we probe images? What type of local features should we consider? How can we establish a dictionary of local features? In what way should we assign a local feature to an entry in the dictionary? What type of global statistical descriptors should we characterize an image with? Of these questions, and of the possible answers, we discuss in Sec. 2, with specific emphasis on the solutions provided by LBP&V. In doing so, we show how these methods can be easily expressed in a formal way within the wider class of histograms of equivalent patterns. We show that many texture descriptors such as local binary patterns, local ternary patterns, texture spectrum, coordinated clusters representation and many others are all instances of the HEP, which can be considered a generalization of LBP&V. Finally we present a geometrical interpretation of these methods as partitioning operators in a high-dimensional space. Since they are expressible as systems of linear equalities and inequalities in the grey-scale intensities, we see that they represent convex polytopes: convex figures bounded by hyperplanes which are analogue of polygons in two dimensions and polyhedra in three dimensions.

2 Fundamental issues in texture analysis

Texture analysis is an area of intense research activity and vast literature. In spite of this, there are two fundamental matters which have not been solved yet: 1) the definition of the concept of texture; and 2) the establishment of a meaningful and unambiguous taxonomy of the existing texture descriptors. The lack of satisfactory solutions to these two key issues is a serious point of weakness that limits the progress of the discipline. We strongly believe that significant advances in the field could be achieved, if these problems could be sorted. In the following subsections we discuss the two matters in detail.

2.1 Defining texture

Texture is a widely used term in computer vision, and it is rather surprising that such a ubiquitous concept has not found a general consensus regarding an explicit definition. Ahonen and Pietikäinen [2] correctly noted that this is perhaps one of the

reasons why neither a unifying theory nor a framework of texture descriptors has been proposed so far.

The root of the word (from Latin `texere` = *to weave*) suggests that texture is somewhat related to the interaction, combination and intertwinement of elements into a complex whole. The concept of texture as the visual property of a surface, however, is rather subjective and imprecise. We can recognize texture when we see it, but defining it in a formal way is much more difficult. Certainly there are some attributes of texture which are largely agreed upon: that texture is the property of an area (and not of a point), that it is related to variation in appearance and that it strongly depends on the scale of an image and that it is perceived as the combination of some basic patterns. Davies, for instance, states that most people would probably call texture a pattern with both randomness and regularity [13]. Petrou and García-Sevilla [65] call texture the 'variation of data at scales smaller than the scale of interest'. Many are, in fact, the definitions that have been proposed in literature: the reader may find a small compendium in Ref. [77]. Unfortunately none of such definition has elicited general consensus, mainly because there is no formal, mathematical model from which we can infer a quantitative general definition.

2.2 Categorizing texture descriptors

The second critical point – which is actually a direct consequence of the first – concerns the development of a taxonomy of texture descriptors. Several attempts to classify texture descriptors have been made so far.

To the best of our knowledge the first attempt dates back to the late 70's and was proposed by Haralick [26]. He divided texture descriptors into *statistical* and *structural*, though it was soon recognised that it was quite difficult to draw a sharp border between the two classes [86, 23]. Such a division was inspired on the pioneering work of Julesz, who conjectured that texture discrimination in the human visual system comes in two forms: perceptive and cognitive [36]. The former provides an immediate characterization of texture and is mostly statistical, the latter requires scrutiny and is mostly structural.

Wu *et al*. [87] refined this two-class taxonomy by splitting the class of statistical methods into five subclasses: spatial gray-level dependence methods, spatial frequency-based features, stochastic model-based features, filtering methods, and heuristic approaches. Then, in the late 90's Tuceryan and Jain proposed a classification into four categories (i.e.: *statistical*, *geometrical*, *model-based* and *signal processing* methods) which gathered a good number of followers and strongly influenced literature thereupon. Yet not even this classification has been exempt from criticism, due to the fact that some texture descriptors possess distinctive traits that belong to more than one class, and therefore a completely crisp separation does not hold in general. Recently, Xie and M. Mirmehdi [88] have suggested that the four classes proposed by Tuceryan and Jain should be rather considered as attributes that one specific method may possess or not. Such a categorization represents, in our

view, the best attempt to classify texture descriptors so far. Yet, any classification based on 'semantic' categories will never be completely satisfactory, because of its intuitive and informal nature. Rather, the correct approach should be based on formal mathematical definitions.

The above mentioned difficulties come up clearly when it comes to find the right placement to LBP and related methods. Though LBP was proposed as 'the unifying approach to the traditionally divergent statistical and structural models of texture analysis' [51], there is actually no consensus on this point, due to the lack of a universally accepted taxonomy. Different authors classify LBP in different ways: as purely statistical [68], purely structural [30], stochastic [72] or even model-based [74, 64].

In this contribution we show that LBP and variants that can be easily defined in a formal way within the HEP. This approach generates no doubt whether a method pertains to this class or not, and therefore is a step in the direction of defining a unifying taxonomy of texture descriptors. Before presenting the mathematical formalism of our approach we need to digress a bit on the concept of bag of features, which helps to clarify the ideas presented herein.

3 Bag of features

The orderless 'bag of features' model (BoF) derives from the 'bag of words' model, an approach to natural language processing in which a text is represented through the total number of occurrences of each word that appear in it, regardless of their ordering [53]. Thus, in this view, the two texts "Spain plays better soccer than Italy" and "Italy plays better soccer than Spain" are perfectly equivalent. Likewise, the BoF represents images through the probability of occurrence of certain local features ('textons'), regardless of their spatial distribution. Such a procedure combines the responses of some local feature detectors obtained at different image locations into convenient statistical descriptors (e.g.: histograms) that summarize the distribution of the features over the region of interest [9]. Representing images through the frequency of a discrete vocabulary of local features has proven widely effective for image classification tasks and object recognition [93, 56]. Fundamental to this approach are the five alternative options that we discuss in the next subsections. These can be viewed as dichotomous design choices and can be conveniently represented through a binary tree, as in Fig. 1.

3.1 Image sampling: dense *vs.* sparse

The first design choice is about the feature sampling mechanism. This can be either *dense* or *sparse*, depending on whether the image is probed exhaustively at

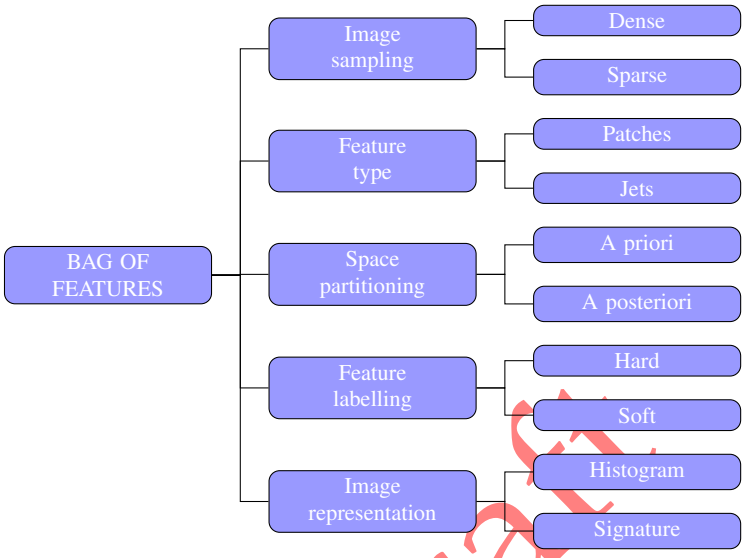

Fig. 1: Dichotomous design choices in the bag-of-features model.

each pixel or not. Though computationally intensive, dense sampling is used very commonly, mainly because it produces no information loss.

Alternatively one can adopt sparse sampling, where the original image is probed at a subset of pixels, which are selected according to some convenient criteria. This has some potential advantages in that it reduces complexity and computational time, but, on the other hand, may induce significant loss of discrimination information. The last point, however, is controversial. Jurie and Triggs [37] showed that sparse sampling often loses a significant amount of information and is outperformed by dense sampling, particularly with inhomogeneous images. In contrast Jiang *et al*. [34] found comparable results with dense and sparse sampling. As for the approaches to sparse sampling, many have been proposed in literature. There are different strategies: they may be image-dependent or image-independent. In the first case one may either subsample the image to a grid of equally-spaced pixels [14] or even perform completely random sampling [57]. In the second case one probes the image at a set of keypoints or 'salient points' detected by some feature detectors, such as the Harris-affine region detector, the Laplacian blob detector [43, 93] or the scale invariant feature transform (SIFT) [49, 92].

3.2 Feature type: patches *vs.* jets

The second design choice is related to the type of local data from which features are extracted. These may be either image filter responses – usually referred to as *jets* – or image *patches*.

In the first case features are extracted from the output of oriented linear filters [52, 44, 83, 69]. The use of filters has a long history in texture analysis and has been justified on the basis of both theoretical and psychophysical considerations. For a long time it was believed that any transform of the original image patches satisfying some optimality criteria should always improve the efficiency, when compared with the original representation [79]. The relation between some classes of filters (e.g.: Gabor filters) and the human vision system [12] has also been frequently argued in favour of their use. Theories of sparse representation and compressed sensing have inspired filtering-based methods too, such as the recently proposed random projections [47]. Yet filtering is not exempt of problems: the design (and implementation) of filter banks is not trivial and is likely to be application-dependent. It is known that parameter tuning is critical and may have significant effects on the results [5]. Moreover the large support that some filters require may be, in some applications, incompatible with the size of the images.

The supremacy of filter bank-based descriptors for texture analysis has been questioned by several authors, most notably Varma and Zissermann [81, 84], who affirmed that filter banks are not strictly necessary, and that using directly grey-scale intensities in local patches with support as small as 3×3 may result in comparable or even superior accuracy. This leads to the second approach, that of image patches. Using image patches is generally faster than using filter responses, and free the user from having to design complex filter banks. Conversely, they tend to be more sensitive to noise [94], since the change in the value of a single pixel can dramatically affect the response of the feature detector. In a comparison between the two methods, Ghita et al. [22] showed that the performances in texture classification offered by LBP/C and multi-channel Gabor filtering are comparable.

As for LBP and related methods, these have been traditionally considered as based on image patches. Recently, Ahonen and Pietikäinen, however, correctly noted that LBP can be viewed as a filter operator based on local derivative filters and a threshold-based vector quantization function [2]. This interpretation in some sense smooths the traditional distinction between patch-based and jet-based methods – at least from a theoretical standpoint. From a practical one, however, we believe that such a distinction remains meaningful and useful, especially when we consider the significant differences that exist in the design and implementation of the two approaches.

3.3 Partitioning the feature space: a priori *vs.* a posteriori

The third design choice involves the definition of convenient rules to partition the feature spaces that arise from either filter responses (jets) or image patches. To this end there are two alternative strategies, which we refer to as *a priori* and *a posteriori*. The difference between the two is that rules are defined independently of data in the former, whereas they are learnt from data in the latter. This design choice is very intriguing since it reflects, in our view, the ancient philosophical debate as to whether knowledge is attained apart from experience or arises from it [38].

A priori partitioning has been the basis of many approaches proposed in literature. The simplest of such methods is perhaps regular subdivision of the feature space into equally-spaced regions (regular binning). The main inconvenience with this method is that the number of required bins grows exponentially as a function of space dimensionality and soon far outweighs the number of datapoints available in a single image with which to populate the histogram, with potential overfitting problems (in an experiment Varma and Zissermann clearly demonstrated that increasing the number of bins decreases the performance [82]). Hash-codes have been proposed as a partial workaround to this problem [78]. More commonly a priori partitioning of the feature space is defined through a function that operates on the grey-scale values of the pixels in a neighbourhood. Typically such a function is based on operators as simple as pairwise comparisons and thresholding [28, 61, 50, 10, 75, 54, 15, 48], though other operators such as ranking [32] have been proposed too. This is the idea LBP&V are based upon, as we shall discuss in detail in the forthcoming section. Alternatively one can define partitioning criteria that operate on filter responses instead of image patches. As for this approach, it is worth mentioning the basic image features (BIF) proposed by Crosier and Griffin [11], which define a mathematical quantisation of a filter response space into seven types of local image structure. Likewise the approach of Rouco *et al*. [69] is based on quantizing the response of some linear filters into three discrete values through an a priori-defined ternary thresholding function.

In the opposite strategy, a posteriori partitioning, the partitioning scheme is learnt from training data. The usual procedure involves the definition of sets of representative local image features usually referred to as 'codebooks'. Typical methods to generate codebooks are based on clustering, which has been used both with image patches [84] and filter responses [81, 47]. Due to its simplicity and good convergence properties, the iterative k-means is the most widely used algorithm in this context. Nonetheless, there are some drawbacks with this procedure: it is time consuming; it requires the number of clusters as input, and it produces non-deterministic results if random initialization is used (which is often the case). Furthermore, frequently appearing patterns are not necessarily the most discriminative [21]: indeed in text analysis, for instance, articles and prepositions are the most frequent words, but they are not particularly discriminative. Alternative approaches to k-means clustering include vector quantization through self-organising maps [80, 60], adaptive binning [45, 39] and sparse coding [19].

The a priori vs. a posteriori dilemma is a subject where scientific interest is currently high. General considerations suggest that a priori approaches are faster, since they do not require codebook generation. They also proved quite accurate in a wide range of practical applications, as demonstrate LBP&V. On the other hand one may argue that in some specific applications, where features tend to cluster in limited portions of the feature space, a priori partitioning schemes may be scarcely efficient, whereas a posteriori schemes may me more convenient, since they can be tuned to the application. Rouco *et al*. [69] recently suggested that, in principle, a priori partitioning is recommendable for broad-domain applications and large image databases, whereas a posteriori (data-driven) partitioning schemes suit better small databases containing few texture classes.

3.4 Feature labelling: hard *vs.* soft

Once the feature space has been partitioned, one has to label each feature of the image to process on the basis of the obtained partition. This is the fourth design choice, which gives rise to two alternative approaches, which we refer to as *hard* and *soft* labelling.

The *hard* approach consist in assigning an image feature to one single partition of the feature space [28, 52, 61, 81]. When codebooks are used, this is usually implemented through the nearest neighbour rule: a feature is assigned the label of the nearest element in the codebook. This results in a Voronoi tessellation of the feature space. Although computationally simple, hard labelling has some important drawbacks, such as sensitivity to noise and limited discriminant power.

The *soft* approach has emerged as a robust alternative to hard labelling. This is based on the consideration that, since verbal descriptions of visual characteristics like colour or a texture are often ambiguous, this ambiguity should be taken into account in the bag-of-features model [21]. The basic idea in soft assignment is that a local feature is assigned to more than one partition, or, equivalently, that partition's borders are not crisp, but fuzzy. Through convenient membership functions [76, 3, 1, 33] one can in fact consider a local feature as belonging to more than one partition. Another possible approach is kernel density estimation, the function of which is to smooth the local neighborhood of data samples. In the implementation proposed by Gemert *et al*. [21] the authors assume that the similarity between an image feature and a codeword is described by a normal function of the distance. Recently, the theory and algorithms of sparse coding and sparse representation have also been proposed to represent a local feature by a linear combination over a codebook. The 'weight' of each element of the codebook on the feature is estimated through suitable minimization procedures [89, 46].

3.5 *Image representation:* histogram *vs.* signature

The last design choice is about image representation. This can be based on *histogram* or *signature*. A histogram of a set with respect to a measurement is a fixed-size vector which reports the frequency of quantified values of that measurement among the samples [73]. The elements of the vector are usually referred to as bins. In the context of this paper histograms report how many times each partition into which the feature space is divided is represented in the image to analyse. In contrast, signatures are variable-size structures which report only the dominant clusters that are extracted from the original data [70]. Each cluster (or node) is represented by its center and a weight that denotes the size of the cluster. Signatures of different images may be therefore different in length, and the order in which clusters are listed does not matter [43]. Consequently, similarity between histograms and signatures are measured differently: histogram similarity is usually evaluated through standard distance functions such as Manhattan, Euclidean or χ^2, whereas signatures are compared through the earth movers' distance.

4 Histograms of equivalent patterns

Different combinations of the five design options discussed in the preceding section have been proposed in literature. We coined the term histograms of equivalent patterns (HEP) to refer to those BoF descriptors that adopt the following design choices: a) dense image sampling; b) image patches as input data; c) a priori partitioning of the features space; d) hard label assignment and e) histogram-based image representation. Here below we show that these concepts can be easily expressed in a formal way. The use of a mathematical formalism makes it possible to determine, unambiguously, whether a texture descriptor belongs to the HEP or not, and therefore removes the inherent uncertainty of semantic taxonomies of which we discussed in Sec. 2.2. We also show how LBP and variants can be regarded as instances of the HEP.

First of all let us introduce the notation to be used henceforth. Let $\mathbf{I}$ be an $M \times N$ matrix representing the raw pixel intensities of an image quantized to G grey levels ranging from 0 to $G-1$, and $I_{m,n}$ the grey-scale intensity at pixel (m,n).

Definition 1. A texture descriptor is a function F that receives an image $\mathbf{I}$ as input and returns a vector $\mathbf{h}$:

$$\mathbf{h} = F(\mathbf{I}) \tag{1}$$

where $\mathbf{h}$ is usually referred to as the *feature vector*.

Definition 2. Histograms of equivalent patterns (HEP) is a class of texture descriptors for which the k-th element of $\mathbf{h}$ can be expressed in the following way:

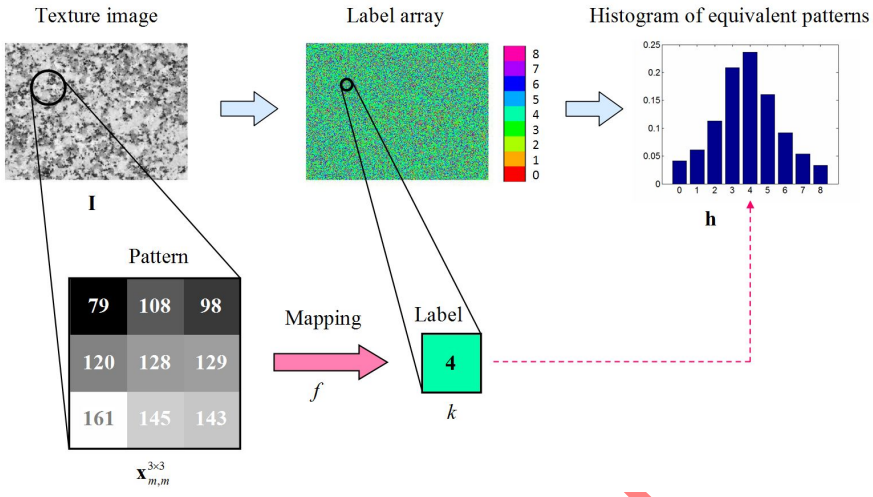

Fig. 2: Schematic representation of the HEP framework.

$$h_k = \frac{1}{D} \sum_{m=m_{min}}^{m_{max}} \sum_{n=n_{min}}^{n_{max}} \delta \left[f\left(\mathbf{x}_{m,n}^{\Omega}, \mathbf{T} \right) - k \right] \quad (2)$$

where m and n represent row- and column-wise pixel indices, $\mathbf{x}_{m,n}^{\Omega}$ the grey-scale values of a set of pixels defining a generic neighbourhood $\Omega_{m,n}$ around (m,n), $\mathbf{T} = [T_1, \cdots, T_p, \cdots, T_P]$ a vector of parameters, D a normalizing factor, δ the function defined in Eq. 7 and f a generic function that returns an integer between 0 and $K-1$. The limits of the sums in Equation 2 are intended to guarantee that for each (m,n) the neighbourhood $\Omega_{m,n}$ be fully contained in $\mathbf{I}$.

In plain words the definition is straightforward: a texture descriptor belonging to the HEP is based on a neighbourhood of predefined shape which scans the image by steps of one pixel. For each position, one among K predefined class labels is assigned to the neighbourhood, and the corresponding k-th component of $\mathbf{h}$ is incremented by $1/D$. Therefore the feature vector represents the probability of occurrence of each class (factor $1/D$ normalizes the feature vector to sum one). Typically $\Omega_{m,n}$ defines a square, rectangular or circular window, but other arrangements have been proposed too: ellipse, parabola, hyperbola, archimedean spiral and other settings can be found in Refs. [62, 63, 54, 27].

Eq. 2 represents the most general case. If we drop the dependance on $\mathbf{T}$ we obtain what we call a *non-parametric* descriptor. Otherwise, if $\mathbf{T}$ is present, we say that the method is *parametric*. In this case the p-th parameter can be either a function of the input image (i.e.: $T_p = T_p(\mathbf{I})$) or not. If at least one parameter is a function of $\mathbf{I}$ we call the method *global*, otherwise we call it *local*.

The definition of a texture descriptor belonging to the HEP is therefore a matter of determining a suitable function f. We refer to it as the *kernel function*, a concept we discuss in detail here below. For the sake of simplicity –but without loss of

generality– we restrict the discussion to 3×3 square neighbourhoods. Now consider a generic image $\mathbf{I}$ and let $\mathbf{x}_{m,n}^{3\times3}$ be the set of grey-scale values of a 3×3 square neighbourhood centred at (m,n):

$$\mathbf{x}_{m,n}^{3\times3} = \begin{bmatrix} I_{m-1,n-1} & I_{m-1,n} & I_{m-1,n+1} \\ I_{m,n-1} & I_{m,n} & I_{m,n+1} \\ I_{m+1,n-1} & I_{m+1,n} & I_{m+1,n+1} \end{bmatrix} \tag{3}$$

In this case the parameters in Eq. 2 take the following values: $m_{min} = n_{min} = 2$, $m_{max} = M-1$, $n_{max} = N-1$ and $D = (M-2)(N-2)$.

Now let $\mathscr{M}_{3\times3,G}$ be the set of all the possible instances defined by Eq. 3. This entity is the feature space we introduced in Sec. 3.2 and represents all the possible grey-scale patterns associated to a predefined neighbourhood (3×3 window, in this case). In the remainder of this section we use the symbol $\mathbf{x}$ to indicate a generic pattern of this type. The HEP partitions the feature space through the a priori-defined function f, which establishes an equivalence relation $\sim$ in $\mathscr{M}_{3\times3,G}$ that acts as follows:

$$\mathbf{x}_1 \sim \mathbf{x}_2 \Leftrightarrow f(\mathbf{x}_1) = f(\mathbf{x}_2) \quad \forall \mathbf{x}_1, \mathbf{x}_2 \in \mathscr{M}_{3\times3,G}. \tag{4}$$

This relation induces a partition in $\mathscr{M}_{3\times3,G}$ that can be expressed in the following way:

$$\mathscr{M}_{3\times3,G} = \bigcup_{0\le k\le K-1} \mathscr{M}_{f,k} \tag{5}$$

where the family of subsets $\{\mathscr{M}_{f,k} \mid 0 \le k \le K-1\}$ is pairwise disjoint, and each subset is defined by:

$$\mathscr{M}_{f,k} = \{\mathbf{x} \in \mathscr{M}_{3\times3,G} \mid f(\mathbf{x}) = k\} \tag{6}$$

If we consider a neighbourhood Ω of generic shape and size, the above reasoning equally holds: f still defines a partition of the pattern space $\mathscr{M}_{\Omega,G}$. In this case the number of possible patterns is G^{ω}, where ω is the number of pixels in Ω. In principle any function f defines a texture descriptor belonging to the HEP. In practice it is recommendable that f satisfy some reasonable constraints. A sensible criterion, for instance, could be that the induced equivalence relation be perceptually meaningful, or, in other words, that similar patterns be mapped into the same equivalence class. For example, in Ref. [20] the authors quantize 3×3 patches according to a modified order statistic, and define equivalence classes based on photometry, complexity and geometry in image space. Another important condition is that f provide effective dimensionality reduction, i.e. $K \ll G^9$.

In some cases a texture descriptor can be obtained combining two or more equivalence relations. The two combination approaches that we consider here are *concatenation* and *joint description*.

Let f_1 and f_2 be two mappings, and K_1 and K_2 the dimensions of the corresponding feature vectors. Concatenation generates a new feature vector that contains the

elements of both f_1 and f_2, therefore its dimension is K_1+K_2. We use the symbol $||$ to indicate this operation.

Joint description means that each class is uniquely identified by two labels, each one generated by a different mapping. Conceptually this operation is very similar to a Cartesian product, thus we indicate it with the symbol $\times$. The number of features is K_1K_2 in this case. In the implementation adopted here, this type of representation is serialized into a one-dimensional feature vector, with the convention that the $(k_1K_2+k_2)$-th element corresponds to class labels k_1 and k_2 of, respectively, f_1 and f_2.

5 LBP and variants within the HEP

In this section we review a selection of LBP variants and show that these apparently divergent texture descriptors are all instances of the HEP. To this end we present a set of LBP&V and provide, for each, the mathematical formulation inside the HEP. In presenting the methods we show how this formalization makes it possible to set into evidence similarities and dissimilarities between the texture descriptors that belong to this family. To keep things simple we limit the review to the $\mathscr{M}_{3\times3,G}$ pattern space, though the formulations presented henceforth can be effortlessly extended to neighbourhoods of different shape and size.

As a preliminary step we define four functions of the real variable x that are extensively used throughout the paper. These are:

- the δ function

$$\delta(x)=\begin{cases}1, & \text{if } x=0\\ 0, & \text{otherwise}\end{cases} \tag{7}$$

- the binary thresholding function

$$b(x)=\begin{cases}1, & \text{if } x\geq 0\\ 0, & \text{if } x<0\end{cases} \tag{8}$$

- the ternary thresholding function

$$t(x,T)=\begin{cases}0, & \text{if } x<-T\\ 1, & \text{if } -T\leq x\leq T\\ 2, & \text{if } x>T\end{cases} \tag{9}$$

- the quinary thresholding function

Table 1: Selection of LBP variants considered in this chapter

Method	Acronym(s)	Kernel function(s)	Year	Ref.
Texture spectrum	TS0	Eq. 12	1990	[28]
Texture spectrum	TS1	Eq. 13	1992	[29]
Rank transform	RT	Eq. 17	1994	[91]
Local binary patterns	LBP	Eq. 18	1996	[59]
Coordinated clusters representation	CCR	Eq. 24	1996	[40]
Modified texture spectrum	MTS	Eq. 19	2003	[90]
Simplified texture spectrum	STS	Eq. 14	2003	[90]
Simplified texture units	STU+,STU×	Eqs. 15,16	2003	[50]
Improved local binary patterns	ILBP	Eq. 20	2004	[35]
Center-symmetric local binary patterns	CS-LBP	Eq. 22	2006	[31]
Median binary patterns	MBP	Eq. 21	2007	[25]
Local ternary patterns	LTP	Eqs. 27–29	2007	[75]
Centralized binary patterns	CBP	Eq. 23	2008	[18]
Improved local ternary patterns	ILTP	Eqs. 30–32	2010	[55]
Completed local binary patterns	CLBP	Eqs. 33,35	2010	[24]
Local quinary patterns	LQP	Eqs. 38,39	2010	[54]
Binary gradient contours	BGC1,BGC2,BGC3	Eqs. 40–42	2011	[15]
Gradient-based local binary patterns	GLBP	Eq. 44	2011	[30]
Improved binary gradient contours	IBGC1	Eq. 43	2012	[17]

$$q(x,T,T') = \begin{cases} 2, & \text{if } x \geq T' \\ 1, & \text{if } T \leq x < T' \\ 0, & \text{if } -T \leq x < T \\ -1, & \text{if } -T' \leq x < -T \\ -2, & \text{if } x < -T' \end{cases} \tag{10}$$

where the parameters T and T' are positive real numbers satisfying $T < T'$.

Next, to avoid cluttered equations, let us take out the sub-indices m,n of Eq. 3 and indicate the set of grey-scale values of a 3×3 neighbourhood in a more manageable way:

$$\mathbf{x} = \begin{bmatrix} I_7 & I_6 & I_5 \\ I_0 & I_c & I_4 \\ I_1 & I_2 & I_3 \end{bmatrix} \tag{11}$$

In the above equation I_c is the grey level of the central pixel and I_j the grey levels of the peripheral pixels ($j \in \{0,1,\ldots,7\}$). We are now ready to start with the list of methods belonging to the HEP (Tab. 1). Of each selected technique we briefly recall the basics and provide the mathematical formulation within the HEP by defining f. A comprehensive compilation of HEP methods can be found in Ref. [17].

5.1 Texture spectrum

Texture spectrum, introduced by He and Wang [28], can be considered the precursor of LBP. In its original formulation it is based on the ternary thresholding function (Eq. 9) with $T = 0$. We refer to this method as TS0. In this model each peripheral pixel of the 3×3 neighbourhood is assigned a value 0, 1 or 2 when its grey-level intensity is less, equal or greater than the intensity of the central pixel, respectively. This defines a set of 3^8 possible ternary patterns. The corresponding kernel function is:

$$f_{\mathrm{TS0}}(\mathbf{x}) = \sum_{j=0}^{7} t\left(I_j - I_c, 0\right) 3^j \tag{12}$$

Later on the same authors proposed a variation of the method in which T takes a value different than zero [29]. This improvement should be potentially beneficial in presence of noise, since a grey-level variation below T does not change a ternary pattern into another. We indicate this method with the acronym TS1, and the corresponding kernel function is formally analogous to Eq. 12:

$$f_{\mathrm{TS1}}(\mathbf{x}, T) = \sum_{j=0}^{7} t\left(I_j - I_c, T\right) 3^j \tag{13}$$

5.2 Simplified texture spectrum

Texture spectrum has been the basis of a good number of variations, all with the aim of reducing the rather high dimensionality of the method. Xu *et al*. [90] proposed a simplified version which we indicate here as simplified texture spectrum (STS). The approach is based on the observation that, as the neighbourhood moves across the image at steps of one pixel, a generic couple of pixels (c, j) switches into the symmetric (j, c). Consequently each comparison between the central pixel and each pixel in the periphery is performed twice. In order to avoid this redundancy and reduce complexity, the authors consider the neighbourhood formed by the central pixel and the four adjacent peripheral pixels of one quadrant only, specifically those corresponding to indices $j \in \{4, \cdots, 7\}$ (see Eq. 11). This setting reduces the number of features from 3^8 to 3^4. The kernel function can be expressed as follows:

$$f_{\mathrm{STS}}(\mathbf{x}) = \sum_{j=4}^{7} t\left(I_c - I_j, 0\right) 3^{j-4} \tag{14}$$

5.3 Simplified texture units

Madrid-Cuevas *et al.* [50] proposed two simplified versions of texture spectrum referred to as simplified texture units. The first one, STU+, considers the neighbourhood composed of the central pixel and its vertically- and horizontally-connected peripheral pixels (i.e.: pixels 0, 2, 4 and 6 in Eq. 11). The second one, STU×, operates on the neighbourhood formed by the central pixel and its diagonally-connected peripheral pixels (i.e.: pixels 1, 3, 5 and 7 in Eq. 11). In both cases dimensionality is reduced from 3^8 to 3^4. The corresponding kernel functions are:

$$f_{\mathrm{STU}+}(\mathbf{x},T)=\sum_{j=0}^{3} t\left(I_{2j}-I_c,T\right)3^j \tag{15}$$

$$f_{\mathrm{STU}\times}(\mathbf{x},T)=\sum_{j=0}^{3} t\left(I_{2j+1}-I_c,T\right)3^j \tag{16}$$

5.4 Rank transform

The rank transform (RT) [91] takes into account the number of pixels in the periphery of the 3×3 region the intensity of which is less than the intensity of the central pixel. Since this number ranges from zero to eight, there are nine possible patterns. The kernel function is:

$$f_{\mathrm{RT}}(\mathbf{x})=\sum_{j=0}^{7} b\left(I_c-I_j-1\right) \tag{17}$$

5.5 Local binary patterns

Local binary patterns (LBP) [59] have received a great deal of attention in the pattern recognition community. In the 3×3 domain the LBP operator thresholds the eight peripheral pixels of the neighbourhood at the value of the central pixel, thus defining a set of 2^8 possible binary patterns. The kernel function is:

$$f_{\mathrm{LBP}}(\mathbf{x})=\sum_{j=0}^{7} b\left(I_j-I_c\right)2^j \tag{18}$$

The first appearance of the expression 'local binary patterns' dates back to a work published in 1994 [58]. Further studies, however, revealed that an embryonic idea of the method had appeared, under a different name, in earlier works too. The

interested reader may find further details about the origin of this very influential method in Refs. [66, 17].

5.6 Modified texture spectrum

Modified texture spectrum (MTS) can be considered as a simplified version of LBP, where only a subset of the peripheral pixels (i.e. pixels 4, 5, 6 and 7 in Eq. 11) is considered. To be precise the inequality in b is flipped in the original formulation of MTS [90], but this unimportant difference in no way alters the information that the method conveys. The kernel function of is:

$$f_{\mathrm{MTS}}(\mathbf{x}) = \sum_{j=4}^{7} b\,(I_c - I_j)\,2^{j-4} \tag{19}$$

5.7 Improved local binary patterns

Improved local binary patterns (ILBP) are based on an idea similar to LBP, the only difference is that the whole 3×3 neighbourhood is thresholded by its average grey-scale value [35]. This gives $(2^9 - 1)$ possible binary patterns (the all 0s pattern is not possible by definition, hence the subtractive term -1 in the equation below). The kernel function is:

$$f_{\mathrm{ILBP}}(\mathbf{x}) = b\,(I_c - T_{\mathrm{mean}})\,2^8 + \sum_{j=0}^{7} b\,(I_j - T_{\mathrm{mean}})\,2^j - 1 \tag{20}$$

where T_{mean} is the average grey-scale value over the whole neighbourhood.

5.8 Median binary patterns

Median binary patterns (MBP) [25] have much in common with ILBP, the only difference is that MBP thresholds the grey-scale values of the 3×3 neighbourhood at their median value, instead of their average value. The kernel function can be expressed as follows:

$$f_{\mathrm{MBP}}(\mathbf{x}) = b\,(I_c - T_{\mathrm{median}})\,2^8 + \sum_{j=0}^{7} b\,(I_j - T_{\mathrm{median}})\,2^j - 1 \tag{21}$$

5.9 Center-symmetric local binary patterns

Center-symmetric local binary patterns (CS-LBP) [31] are similar to LBP, but employ a different scheme to compare the pixel in the neighbourhood. Whereas the central pixel plays a pivotal role in LBP, CS-LBP discards it altogether and considers the following centre-symmetric couples of pixel values (Eq. 11): (I_0, I_4), (I_1, I_5), (I_2, I_6) and (I_3, I_7). Robustness on flat image regions is obtained by thresholding the gray level differences with a parameter T. This generates a set of 2^4 possible patterns. The kernel function can be expressed in the following way:

$$f_{\text{CS-LBP}}(\mathbf{x}, T) = \sum_{j=0}^{3} b\left(I_j - I_{j+4} - T - 1\right) 2^j \tag{22}$$

5.10 Centralized binary patterns

Centralized binary patterns (CBP) [18] consider the same couples of centre-symmetric pixels used by CS-LBP plus the central pixel. Relative comparison is based on the absolute difference of grey-scale values, which is thresholded at a predefined small positive value T. The kernel function can be formalized as follows:

$$f_{\text{CBP}}(\mathbf{x}, T) = b\left(|I_c - T_{\text{mean}}| - T\right) 2^4 + \sum_{j=0}^{3} b\left(\left|I_j - I_{j+4}\right| - T\right) 2^j \tag{23}$$

where T_{mean} is defined as in Sec. 5.7.

5.11 Coordinated clusters representation

The coordinated clusters representation (CCR) was originally intended as a texture descriptor for binary images [40]. It was later on extended to grey-scale images through a preliminary thresholding step [71], and, recently, to colour images too [7]. The method is based on the probability of occurrence of the 2^9 possible binary instances of a 3×3 window. It is similar to LBP and ILBP, though threshold is global in this case. The global threshold can be computed in various ways. A possible approach to estimate it is through isentropic partition [6, 16]. In this case threshold T_{isoentr} is the value that divides the grey-scale histogram into two parts of equal entropy. The kernel function can be expressed as follows:

$$f_{\text{CCR}}(\mathbf{x}, T_{\text{isoentr}}) = b\left(I_c - T_{\text{isoentr}}\right) 2^8 + \sum_{j=0}^{7} b\left(I_j - T_{\text{isoentr}}\right) 2^j \tag{24}$$

5.12 Local ternary patterns

Local ternary patterns [75] can be considered a hybrid between texture spectrum and local binary patterns. Similarily to texture spectrum, in fact, they make use of the ternary thresholding function to obtain ternary binary patterns. Each ternary pattern is split in two binary patterns (lower and upper) through the following rules:

$$b_{j,\mathrm{LOWER}} = \begin{cases} 1, & \text{if } t_j = 0 \\ 0, & \text{otherwise} \end{cases} \tag{25}$$

$$b_{j,\mathrm{UPPER}} = \begin{cases} 1, & \text{if } t_j = 2 \\ 0, & \text{otherwise} \end{cases} \tag{26}$$

where t_j and b_j represent, respectively, the ternary and binary value corresponding to pixel j. With this convention the kernel functions that define the distributions of lower (LTPL) and upper (LTPU) local ternary patterns can be expressed in the following way:

$$f_{\mathrm{LTPL}}(\mathbf{x},T) = \sum_{j=0}^{7} b\,(I_c - I_j - T)\,2^j \tag{27}$$

$$f_{\mathrm{LTPU}}(\mathbf{x},T) = \sum_{j=0}^{7} b\,(I_j - I_c - T)\,2^j \tag{28}$$

The two descriptors are finally concatenated to form the LTP model:

$$\mathbf{h}_{\mathrm{LTP}} = \mathbf{h}_{\mathrm{LTPU}} || \mathbf{h}_{\mathrm{LTPL}} \tag{29}$$

5.13 Improved local ternary patterns

Improved local ternary patterns (ILTP) [55] are an extension of LTP where each pixel in the neighbourhood is thresholded at the average grey-scale value. Similarly to LTP the representation is split into a lower and upper part:

$$f_{\mathrm{ILTPL}}(\mathbf{x},T) = b\,(T_{\mathrm{mean}} - I_c - T)\,2^8 + \sum_{j=0}^{7} b\,(T_{\mathrm{mean}} - I_j - T)\,2^j \tag{30}$$

$$f_{\mathrm{ILTPU}}(\mathbf{x},T) = b\,(I_c - T_{\mathrm{mean}} - T)\,2^8 + \sum_{j=0}^{7} b\,(I_j - T_{\mathrm{mean}} - T)\,2^j \tag{31}$$

where T_{mean} is defined as in Sec. 5.7. The two descriptors are concatenated to give the ILTP model:

$$\mathbf{h}_{\mathrm{ILTP}} = \mathbf{h}_{\mathrm{ILTPH}} || \mathbf{h}_{\mathrm{ILTPL}} \tag{32}$$

5.14 Completed local binary patterns

Completed Local Binary Patterns (CLBP) have been recently introduced by Guo *et al*. [24] as an extension of local binary patterns. The approach is based on different combinations of three basic descriptors: CLBP_C, CLBP_M and CLBP_S. The last is just an alias for standard LBP, already treated in Sec. 5.5.

CLBP_C thresholds the central pixel of the 3×3 neighbourhood at the average grey-scale value of the whole image, and therefore generates only two binary patterns. The kernel function is:

$$f_{\mathrm{CLBP\_C}}(\mathbf{x}, T_{\bar{I}}) = b\,(I_c - T_{\bar{I}}) \tag{33}$$

where

$$T_{\bar{I}} = \frac{\sum_{m=1}^{M}\sum_{n=1}^{N} I_{m,n}}{MN} \tag{34}$$

CLBP_M considers the possible binary patterns that are defined by the absolute difference between the grey-scale value of a pixel in the periphery and that of the central pixel when thresholded with a global parameter. In formulas:

$$f_{\mathrm{CLBP\_M}}(\mathbf{x}, T_{\Delta\bar{I}}) = \sum_{j=0}^{7} b\,(|I_j - I_c| - T_{\Delta\bar{I}})\,2^j \tag{35}$$

where $T_{\Delta\bar{I}}$ is the average value of the difference in grey value between a pixel in the periphery and the central pixel:

$$T_{\Delta\bar{I}} = \frac{\sum_{m=2}^{M-1}\sum_{n=2}^{N-1}\sum_{i=-1}^{1}\sum_{j=-1}^{1} |I_{m-i,n-j} - I_{m,n}|}{8(M-2)(N-2)} \tag{36}$$

In [24] the authors suggest that the three descriptors can be combined in different ways to give joint and concatenated descriptors, for example: CLBP_M $\times$ CLBP_C, CLBP_S $\times$ CLBP_M, CLBP_M $\times$ CLBP_S $\times$ CLBP_C and CLBP_S || CLBP_M $\times$ C.

5.15 Local quinary patterns

In local quinary patterns (LQP) [54] the grey level difference between the central pixel and the pixels of the periphery is encoded using five discrete levels (i.e.: -2, -1, 0, 1 and 2) which are computed using two thresholds: T and T'. LQP is therefore closely related to LTP, the only difference being that the number of encoding levels is five in LQP and three in LTP. The quinary pattern is split into four binary patterns through the following rule:

$$b_{j,i} = \begin{cases} 1, & \text{if } q_j = i \\ 0, & \text{otherwise} \end{cases} \tag{37}$$

where $b_{j,i}$ and q_j are the binary and quinary value corresponding to pixel j and level i; $i \in \{-2,-1,1,2\}$. The kernel function corresponding to each level is:

$$f_{\mathrm{LQP},i}(\mathbf{x},T,T') = \sum_{j=0}^{7} \delta\left[q\left(I_c - I_j,T,T'\right) - i\right] 2^j \tag{38}$$

The feature vector is obtained as follows:

$$\mathbf{h}_{\mathrm{LQP}} = \mathbf{h}_{\mathrm{LQP},-2} || \mathbf{h}_{\mathrm{LQP},-1} || \mathbf{h}_{\mathrm{LQP},1} || \mathbf{h}_{\mathrm{LQP},2} \tag{39}$$

5.16 Binary gradient contours

The recently introduced binary gradient contours (BGC) [15], are a family of descriptors based on pairwise comparison of adjacent pixels belonging to one or more closed paths traced along the periphery of the 3×3 neighbourhood (hence the name *contours*). For each closed path a binary pattern is obtained by assigning each pair of adjacent pixels (i,j) in the path the binary value $b(I_j - I_i)$. Since there are several paths that one can pick out from the 3×3 neighbourhood, different BGC operators exist. In Ref. [15] we proposed three different operators, which are referred to as BGC1, BGC2 and BGC3. Both BCG1 and BCG3 are based on one closed path. In the first case the pixels that define the path are: $\{0,1,2,3,4,5,6,7,0\}$ (Eq. 11), therefore the corresponding couples from which the binary values are extracted are: $\{(0,1),(1,2),\cdots,(7,0)\}$. In the second case the path is defined by the following sequence of pixels: $\{0,5,2,7,4,1,6,3,0\}$ (couples are defined in the same way). Both descriptors generate (2^8-1) possible different patterns, since, as it happens with ILBP, the all-0s pattern is, by definition, impossible. In contrast BGC2 employs two closed paths, which are: $\{1,7,5,3,1\}$ and $\{0,6,4,2,0\}$. In this case each path generates (2^4-1) possible patterns, therefore the joint combination of the two gives $(2^4-1)^2 = 225$ possible patterns. The kernel functions of the three models are reported here below:

$$f_{\mathrm{BGC1}}(\mathbf{x}) = \sum_{j=0}^{7} b\left(I_j - I_{(j+1) \bmod 8}\right) 2^j - 1 \tag{40}$$

$$f_{\mathrm{BGC2}}(\mathbf{x}) = (2^4 - 1) \sum_{j=0}^{3} b\left(I_{2j} - I_{2(j+1) \bmod 8}\right) 2^j + \sum_{j=0}^{3} b\left(I_{2j+1} - I_{(2j+3) \bmod 8}\right) 2^j - 2^4 \tag{41}$$

$$f_{\mathrm{BGC3}}(\mathbf{x}) = \sum_{j=0}^{7} b\left(I_{3j \bmod 8} - I_{3(j+1) \bmod 8}\right) 2^j - 1 \tag{42}$$

5.17 *Improved binary gradient contours*

An extension of BGC1 has been recently proposed [17]. The improved binary gradient contour (IBGC1) includes the central pixel and can be easily derived from the original formulation by comparing the central pixel value with the average grey-scale value T_{mean} over the 3×3 neighbourhood. The kernel function is:

$$f_{\mathrm{IBGC1}}(\mathbf{x}) = b\left(I_c - T_{\mathrm{mean}}\right)(2^8 - 1) + \sum_{j=0}^{7} b\left(I_i - I_{(j+1) \bmod 8}\right) 2^j - 1 \tag{43}$$

where T_{mean} is defined as in Sec. 5.7.

5.18 *Gradient-based local binary patterns*

Another LBP-related method is represented by gradient-based local binary patterns (GLBP) [30]. Here the absolute difference between the central pixel and each peripheral pixel is thresholded at the mean absolute difference between (I_0, I_4) and (I_2, I_6). In formulas:

$$f_{\mathrm{GLBP}}(\mathbf{x}) = \sum_{j=0}^{7} b\left(I_+ - \left|I_j - I_c\right|\right) 2^j \tag{44}$$

where:

$$I_+ = \frac{1}{2}\left(|I_0 - I_4| + |I_2 - I_6|\right) \tag{45}$$

6 Geometrical interpretation

In the preceding section we have provided an algebraic formulation of LBP and variants in the framework of the HEP. We show that these methods can be viewed as partitioning operators acting on a high-dimensional space. This is no surprise if we consider that, in the end, texture analysis is about modeling and understanding the distribution of a population in a high-dimensional space. As noted by Pothos *et al*. [67], many texture descriptors extract different features in number and quality by providing proper quantization and optimal partitioning of the high-dimensional space. So are, for instance, Crosier and Griffin's basic image features [11], through which the authors establish a direct link between detection of local features and space partitioning, and the local intensity order pattern (LIOP), recently proposed by Wang *et al*. [85]. Local binary patterns and variants are clearly based on this idea as well (as we show here below), though they have been seldom investigated under this perspective, and rarely studied under a theoretical viewpoint either. Actually, we are only aware of two references on this subject [41, 8]; in both the authors study the probability distribution of local binary patterns and show the high a-priori probability of uniform patterns.

Now, we would like to give a geometrical interpretation of LBP&V. If we take a look at the kernel functions of the methods presented in Sec. 5, we soon recognise that these define sets of equalities and/or inequalities in the variables I_c and I_j, $j \in \{0,\cdots,7\}$ (see Eq. 11). Geometrically they represent polytopes in $\mathscr{M}_{3\times3,G}$, i.e.: regions bounded by hyperplanes which are equivalent to polygons in two dimensions. The good news is that many things about polytopes are known: efficient algorithms exist to calculate their exact volume [42] and to count how many discrete integer points fall inside them [4]. This makes it possible to investigate the theoretical properties of a texture descriptor through the volume distribution of its corresponding polytopes. In Ref. [15] we showed how this relates to the theoretical efficiency of a method: under the assumption of uniformly-distributed and stochastically-independent grey-scale values this is optimal when the feature space is partitioned into regions of equal volume.

In order to clarify these ideas we present a motivational example in which we reduce the dimensionality of the problem. We consider, to this end, an L-shaped 3-pixel neighbourhood like the one depicted in Fig. 3a. Pixel intensities are denoted by I_1, I_2 and I_3. We assume that grey-scale is continuous rather than discrete, and we also assume the simplifying hypotheses considered in Ref. [8], namely that pixel intensities are uniformly distributed in the range [0,1] and stochastically-independent. The feature space is therefore the unitary cube of uniform density represented in Fig. 3b.

This 3-dimensional model allows polytopes (polyhedra – in this case) to be easily visualized, a thing that would be impossible in a higher-dimensional space. We now wish to provide a geometrical representation of LBP&V in this model. For the sake of simplicity we limit the study to three representative examples: LBP, ILBP and BGC. Every part into which each method divides the feature space of Fig. 3b can be labelled through a binary string in the following way:

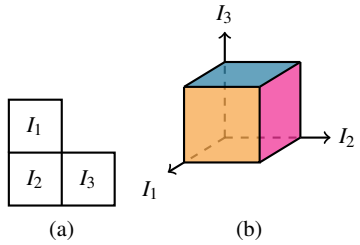

Fig. 3: L-shaped neighbourhood formed by 3 pixels: (a) layout of pixel intensities, and (b) corresponding pattern space.

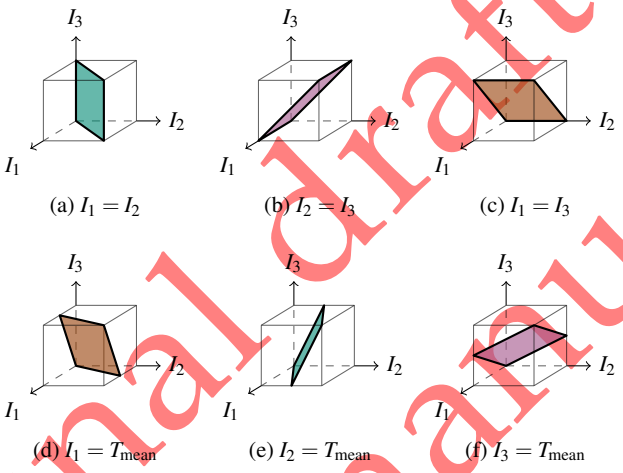

Fig. 4: Planes used to split the pattern space

$$\text{LBP}(I_1, I_2, I_3) = [b(I_1 - I_2) \quad b(I_3 - I_2)]_2 \tag{46}$$

$$\text{ILBP}(I_1, I_2, I_3) = [b(I_1 - T_{\text{mean}}) \quad b(I_2 - T_{\text{mean}}) \quad b(I_3 - T_{\text{mean}})]_2 \tag{47}$$

$$\text{BGC1}(I_1, I_2, I_3) = [b(I_1 - I_2) \quad b(I_2 - I_3) \quad b(I_3 - I_1)]_2 \tag{48}$$

The above equations can be easily interpreted, geometrically, if we consider that each of the following operations: $b(I_1 - I_2)$, $b(I_2 - I_3)$, $b(I_1 - I_3)$, $b(I_1 - T_{\text{mean}})$, $b(I_2 - T_{\text{mean}})$ and $b(I_3 - T_{\text{mean}})$, corresponds to a 'cut' in the pattern space through each of the planes represented in Figs. 4a-4d.

Now let's start to see what happens with LBP (Eq. 46). This operator divides the pattern space into the four polyhedra depicted in Fig. 5. Simple calculations show

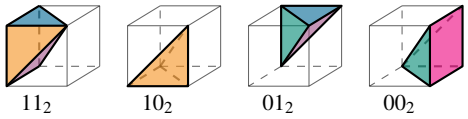

Fig. 5: Polyedra associated to LBP along with their corresponding binary codes

that the polyhedra are not the same volume: the volume of polyhedra 11_2 and 00_2 is $\frac{2}{6}$, while the volume of polyhedra 10_2 and 01_2 is $\frac{1}{6}$. Thus the partition induced by this operator is sub-optimal, therefore we expect a theoretical efficiency below 1. The exact value in fact is:

$$e_{\mathrm{LBP}} = \frac{-2 \times (\frac{2}{6} \log \frac{2}{6} + \frac{1}{6} \log \frac{1}{6})}{\log 4} = \frac{1.9183}{2} = 0.9591 \tag{49}$$

Likewise, we can repeat the same reasoning for ILBP (Eq. 47). In this case it is important to point out, preliminarily, the binary string 000_2 is impossible by definition, since the three conditions: $I_1 < T_{\mathrm{mean}}$, $I_2 < T_{\mathrm{mean}}$ and $I_3 < T_{\mathrm{mean}}$ cannot hold at once. The binary string 111_2 also deserves special attention: in this case the three conditions $b(I_1 - T_{\mathrm{mean}}) \geq 0$, $b(I_2 - T_{\mathrm{mean}}) \geq 0$ and $b(I_3 - T_{\mathrm{mean}}) \geq 0$ are satisfied simultaneously if $I_1 = I_2 = I_3$, which is the equation of the main diagonal of the pattern space (i.e.: unitary cube) passing through the origin of coordinates. This results in a degenerate polytope (i.e.: line segment) of dimension one and null volume. The polyhedra corresponding to the remaining six binary strings are depicted in Fig. 6. Note that these six polyhedra have the same volume ($\frac{1}{6}$), therefore the partition is optimal. The corresponding efficiency is in fact 1:

$$e_{\mathrm{ILBP}} = \frac{-6 \times \frac{1}{6} \log \frac{1}{6}}{\log 6} = 1 \tag{50}$$

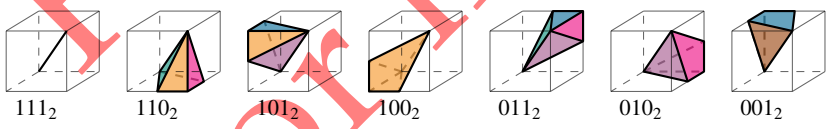

Fig. 6: Polyedra associated to ILBP along with their corresponding binary codes

Finally, we conclude the examples considering BGC1 (Eq. 48). In this case we have that the string 000_2 is impossible by definition, due to the fact that the three conditions $I_1 < I_2$, $I_2 < I_3$ and $I_3 < I_1$ cannot be satisfied simultaneously. The binary string 111_2 also deserves special attention. Similarly to ILBP, we note that the three following conditions $b(I_1 - I_2) \geq 0$, $b(I_2 - I_3) \geq 0$ and $b(I_3 - I_1) \geq 0$ are satisfied simultaneously if $I_1 = I_2 = I_3$, which again is the diagonal from the origin to the

point $(1,1,1)$. The polyhedra corresponding to the remaining six binary strings are represented in Fig. 7. Notice that these six polyhedra have the same volume ($\frac{1}{6}$). The entropy of the corresponding codebook can be calculated as:

$$e_{\mathrm{BGC1}} = \frac{-6 \times \frac{1}{6} \log \frac{1}{6}}{\log 6} = 1 \tag{51}$$

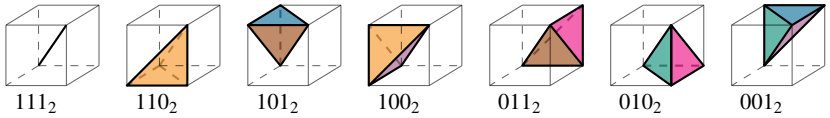

Fig. 7: Polyedra associated to BGC1 along with their corresponding binary codes

The examples provided above, though simplified, help us understand the rationale behind local binary patterns and related methods. In particular they show how the kernel function relates to the way the feature space is partitioned. All the patterns belonging to such partition are considered equivalent, and can be represented through a unique arbitrary symbol. The set of such symbols can be regarded as an alphabet to represent textures. Since we assumed, in this example, uniformly-distributed and stochastically independent grey-scale intensities, the volume of each subdivision represents the probability that a texture descriptor assigns, a priori, to that partition, and therefore to the corresponding symbol. From information theory we know that the efficiency of an alphabet is maximum when its symbols are equally likely, that is to say when the entropy of the histogram representing the probability distribution of each symbol is maximum. This clarifies the concept of theoretical efficiency introduced above. If we leave the simplified realm of 3-pixel, continous-valued, L-shaped neighbourhood (Fig. 3), and make for the standard domain of, for instance, 3×3 neighbourhoods of discrete grey-level intensities, these considerations equally hold, but with some differences. In Ref. [8] the curious reader can find an *in extenso* calculation of polytopes' volumes for $\mathrm{LBP}_{3\times3}$ and $\mathrm{LBP}_{8,1}$ both in the continuous and discrete domain.

7 Conclusions and open issues

In this paper we have proposed a unifying framework for local binary patterns and variants which we refer to as histograms of equivalent patterns (HEP). Starting from a discussion on some basic issues in texture analysis – such as the definition of texture itself and the classification of the multitude of texture descriptors – we have digressed about the concept of bag-of-features. Within this idea we have identified five dichotomous choices that are fundamental in designing texture descriptors: a)

image sampling (dense or sparse?); b) type of features (patches or jets?); c) space partitioning (a priori or a posteriori?); d) label assignment (hard or soft?) and e) image representation (histogram or signature?). We have shown which choices are adopted by the HEP and how these can be formalized mathematically. This enables local binary patterns and variants to be expressed through a common formalism and sets into evidence that they can all be viewed as instances of the HEP. In the last part of the chapter we have given a geometrical reading of these methods showing how they partition the feature space into polytopes. This interpretation makes it possible to study some intrinsic theoretical properties of the methods, and suggests possible directions for future research: this could be focused on studying partitioning schemes (i.e.: kernel functions) that maximize the theoretical amount of information conveyed by the descriptor, and, possess, at the same time, some type of invariance, such as robustness against against rotation, grey-scale transformations or viewpoint changes.

Acknowledgements

This work was supported by the Spanish Government under projects no. TRA2011-29454-C03-01 and CTM2010-16573.

References

1. T. Ahonen and M. Pietikäinen. Soft histograms for local binary patterns. In *Proceedings of the Finnish Signal Processing Symposium (FINSIG 2007)*, Oulu, Finland, 2007.
2. T. Ahonen, J. Matas, C. He, and M. Pietikäinen. Rotation invariant image description with local binary pattern histogram Fourier features. In *Proceedings of the 16th Scandinavian Conference (SCIA 2009)*, volume 5575 of *Lecture Notes in Computer Science*, pages 61–70. Springer, 2009.
3. A. Barcelo, E. Montseny, and P. Sobrevilla. On fuzzy texture spectrum for natural microtextures characterization. In *Proceedings EUSFLAT-LFA 2005*, pages 685–690, 2005.
4. M. Beck and S. Robins. *Computing the Continuous Discretely. Integer-point Enumeration in Polyhedra*. Springer, New York, 2007.
5. F. Bianconi and A. Fernández. Evaluation of the effects of Gabor filter parameters on texture classification. *Pattern Recognition*, 40(12):3325–3335, 2007.
6. F. Bianconi, A. Fernández, E. González, and F. Ribas. Texture classification through combination of sequential colour texture classifiers. In Luis Rueda, Domingo Mery, and Josef Kittler, editors, *Progress in Pattern Recognition, Image Analysis and Applications. Proceedings of the 12th Iberoamerican Congress on Pattern Recognition (CIARP 2007)*, volume 4756 of *Lecture Notes in Computer Science*, pages 231–240. Springer, 2008.
7. F. Bianconi, A. Fernández, E. González, D. Caride, and A. Calviño. Rotation-invariant colour texture classification through multilayer CCR. *Pattern Recognition Letters*, 30(8):765–773, 2009.
8. F. Bianconi and A. Fernández. On the occurrence probability of local binary patterns: a theoretical study. *Journal of Mathematical Imaging and Vision*, 40(3):259–268, 2011.

9. Y.-L. Boureau, J. Ponce, and Y. LeCun. A theoretical analysis of feature pooling in visual recognition. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 111–118, Haifa, Israel, June 2010.
10. C.-I. Chang and Y. Chen. Gradient texture unit coding for texture analysis. *Optical Engineering*, 43(8):1891–1902, 2004.
11. M. Crosier and L. D. Griffin. Using basic image features for texture classification. *International Journal of Computer Vision*, 88:447–460, 2010.
12. J.G. Daugman. Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters. *Journal of the Optical Society of America A*, 2:1160–1169, 1985.
13. E. R. Davies. Introduction to texture analysis. In M. Mirmehdi, X. Xie, and J. Suri, editors, *Handbook of texture analysis*, pages 1–31. Imperial College Press, 2008.
14. L. Fei-Fei and P. Perona. A bayesian hierarchical model for learning natural scene categories. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2005)*, volume 2, pages 524–531, June 2005.
15. A. Fernández, M. X. Álvarez, and F. Bianconi. Image classification with binary gradient contours. *Optics and Lasers in Engineering*, 49(9-10):1177–1184, 2011.
16. A. Fernández, O. Ghita, E. González, F. Bianconi, and P. F. Whelan. Evaluation of robustness against rotation of LBP, CCR and ILBP features in granite texture classification. *Machine Vision and Applications*, 22(6):913–926, 2011.
17. A. Fernández, M. X. Álvarez, and F. Bianconi. Texture description through histograms of equivalent patterns. *Journal of Mathematical Imaging and Vision*, 2012. to appear.
18. X. Fu and W. Wei. Centralized binary patterns embedded with image euclidean distance for facial expression recognition. In *Proceedings of the Fourth International Conference on Natural Computation (ICNC'08)*, volume 4, pages 115–119, 2008.
19. M. J. Gangeh, A. Ghodsi, and M. Kamel. Dictionary learning in texture classification. In M. Kamel and A. Campilho, editors, *Proceedings of the 8th International Conference on Image Analysis and Recognition (ICIAR 2011)*, volume 6753 of *Lecture Notes in Computer Science*, pages 335–343, Burnaby, BC, Canada, June 2011. Springer.
20. D. Geman and A. Koloydenko. Invariant statistics and coding of natural microimages. In *Proceedings of the First International Workshop on Statistical and Computational Theories of Vision*, Fort Collins, CO, USA, June 1999. published on the web.
21. J. C. van Gemert, C. J. Veenman, A. W. M. Smeulders, and J.-M. Geusebroek. Visual word ambiguity. *IEEE Transactions on Patterns Analysis and Machine Intelligence*, 32(7):1271–1283, 2010.
22. O. Ghita, D. E. Ilea, A. Fernández, and P. F. Whelan. Local binary patterns versus signal processing texture analysis. A study from a performance evaluation perspective. *Sensor Review*, 32:149–162, 2012.
23. L. van Gool, P. Dewaele, and A. Oosterlinck. Texture analysis anno 1983. *Computer Vision, Graphics, and Image Processing*, 29(3):336–357, 1985.
24. Z. Guo, L. Zhang, and D. Zhang. A completed modeling of local binary pattern operator for texture classification. *IEEE Transactions on Image Processing*, 19(6):1657–1663, 2010.
25. A. Hafiane, G. Seetharaman, and B. Zavidovique. Median binary pattern for textures classification. In *Proceedings of the 4th International Conference on Image Analysis and Recognition (ICIAR 2007)*, volume 4633 of *Lecture Notes in Computer Science*, pages 387–398, Montreal, Canada, August 2007.
26. R. M. Haralick. Statistical and structural approaches to texture. *Proceedings of the IEEE*, 67(5):786–804, 1979.
27. Y. He, N. Sang, and C. Gao. Multi-structure local binary patterns for texture classification. *Pattern Analysis and Applications*, pages 1–13, 2012. Article in Press.
28. D.-C. He and L. Wang. Texture unit, texture spectrum, and texture analysis. *IEEE Transactions on Geoscience and Remote Sensing*, 28(4):509–512, 1990.
29. D.-C. He and L. Wang. Unsupervised textural classification of images using the texture spectrum. *Pattern Recognition*, 25(3):247–255, 1992.

30. Y. He and N. Sang. Robust illumination invariant texture classification using gradient local binary patterns. In *Proceedings of 2011 International Workshop on Multi-Platform/Multi-Sensor Remote Sensing and Mapping*, pages 1–6, Xiamen, China, January 2011.
31. M. Heikkilä, M. Pietikäinen, and C. Schmid. Description of interest regions with Local Binary Patterns. *Pattern Recognition*, 42:425–436, 2009.
32. L. Hepplewhite and T. J. Stonhamm. Texture classification using N-tuple pattern recognition. In *Proceedings of the 13th International Conference on Pattern Recognition (ICPR'96)*, volume 4, pages 159–163, August 1996.
33. D. K. Iakovidis, E. G. Keramidas, and D. Maroulis. Fuzzy local binary patterns for ultrasound texture characterization. In Aurélio Campilho and Mohamed Kamel, editors, *Proceedings of the 5th International Conference on Image Analysis and Recognition (ICIAR 2008)*, volume 5112 of *Lecture Notes in Computer Science*, pages 750–759, Póvoa de Varzim, Portugal, June 2008.
34. Y.-G. Jiang, C.-W. Ngo, and Jun Yang. Towards optimal bag-of-features for object categorization and semantic video retrieval. In *Proceedings of the 6th ACM international conference on Image and Video Retrieval (CIVR'07)*, pages 494–501, 2007.
35. H. Jin, Q. Liu, H. Lu, and X. Tong. Face detection using improved LBP under bayesian framework. In *Proceedings of the 3rd International Conference on Image and Graphics*, pages 306–309, 2004.
36. B. Julesz. Experiments in the visual perception of texture. *Scientific American*, 232(4):34–43, 1975.
37. F. Jurie and B. Triggs. Creating efficient codebooks for visual recognition. In *Proceedings of the Tenth IEEE International Conference on Computer Vision (ICCV'05)*, volume 1, pages 604–610, 2005.
38. I. Kant. *The Critique of Pure Reason*. Pennsylvania State Electronic Classics Series, 2010. Translated by J. M. D. Meiklejoh.
39. S. Konishi and A. L. Yuille. Statistical cues for domain specific image segmentation with performance analysis. In *Proceedings of the 2000 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'00)*, volume 1, pages 125–132, 2000.
40. E. V. Kurmyshev and M. Cervantes. A quasi-statistical approach to digital binary image representation. *Revista Mexicana de Física*, 42(1):104–116, 1996.
41. O. Lahdenoja. A statistical approach for characterising local binary patterns. Technical Report 795, Turku Centre for Computer Science, Finland, 2006.
42. J. Lawrence. Polytope volume computation. *Mathematics of Computation*, 57(195):259–271, 1991.
43. S. Lazebnik, C. Schmid, and J. Ponce. A sparse texture representation using local affine regions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1265–1278, 2005.
44. T. Leung and J. Malik. Representing and recognizing the visual appearance of materials using three-dimensional textons. *International Journal of Computer Vision*, 43(1):29–44, 2001.
45. X. Liu and D. Wang. Texture classification using spectral histograms. *IEEE Transactions on Image Processing*, 12(6):661–670, 2003.
46. L. Liu, L. Wang, and X. Liu. In defense of soft-assignment coding. In *Proceedings of the 13th IEEE International Conference on Computer Vision (ICCV 2011)*, pages 2486–2493, 2011.
47. L. Liu and P. W. Fieguth. Texture classification from random features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(3):574–586, 2012.
48. L. Liu, L. Zhao, Y. Long, G. Kuang, and P. W. Fieguth. Extended local binary patterns for texture classification. *Image and Vision Computing*, 30(2):86–99, 2012.
49. D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.
50. F. J. Madrid-Cuevas, R. Medina, M. Prieto, N. L. Fernández, and A. Carmona. Simplified texture unit: A new descriptor of the local texture in gray-level images. In Francisco J. Perales López, Aurélio C. Campilho, Nicolás Pérez de la Blanca, and Alberto Sanfeliu, editors, *Pattern Recognition and Image Analysis, Proceedings of the First Iberian Conference (IbPRIA 2003)*, volume 2652 of *Lecture Notes in Computer Science*, pages 470–477. Springer, 2003.

51. T. Mäenpää and M. Pietikäinen. Texture analysis with local binary patterns. In C. H. Chen and P. S. P. Wang, editors, *Handbook of Pattern Recognition and Computer Vision (3rd Edition)*, pages 197–216. World Scientific Publishing, 2005.
52. J. Malik, S. Belongie, J. Shi, and T. Leung. Textons, contours and regions: cue integration in image segmentation. In *The Proceedings of the Seventh IEEE International Conference on Computer Vision (ICCV'99)*, volume 2, pages 918–925, 1999.
53. C.D. Manning, P. Raghavan, and H. Schütze. *An introduction to information retrieval*. Cambridge University Press, Cambridge (UK), 2009.
54. L. Nanni, A. Lumini, and S. Brahnam. Local binary patterns variants as texture descriptors for medical image analysis. *Artificial Intelligence in Medicine*, 49(2):117–125, 2010.
55. L. Nanni, S. Brahnam, and A. Lumini. A local approach based on a Local Binary Patterns variant texture descriptor for classifying pain states. *Expert Systems with Applications*, 37(12):7888–7894, 2010.
56. L. Nanni, S. Brahnam, and A. Lumini. Random interest regions for object recognition based on texture descriptors and bag of features. *Expert Systems with Applications*, 39(1):973–977, 2012.
57. E. Nowak, F. Jurie, and B. Triggs. Sampling strategies for bag-of-features image classification. In *Proceedings of European Conference on Computer Vision 2006 (ECCV'06), Part IV*, volume 3954 of *Lecture Notes in Computer Science*, pages 490–503. Springer-Verlag, 2006.
58. T. Ojala, M. Pietikäinen, and D. Harwood. Performance evaluation of texture measures with classification based on Kullback discrimination of distributions. In *Proceedings of the 12th IAPR International Conference on Pattern Recognition*, volume 1, pages 582–585, 1994.
59. T. Ojala, M. Pietikäinen, and D. Harwood. A comparative study of texture measures with classification based on feature distributions. *Pattern Recognition*, 29(1):51–59, 1996.
60. T. Ojala, M. Pietikäinen, and J. Kyllönen. Gray level cooccurrence histograms via learning vector quantization. In *Proceedings of the 11th Scandinavian Conference on Image Analysis (SCIA 1999)*, pages 103–108, Kangerlussuaq, Greenland, 1999.
61. T. Ojala, M. Pietikäinen, and T. Mäenpää. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):971–987, 2002.
62. D. Patel and T. J. Stonham. A single layer neural network for texture discrimination. In *IEEE International Symposium on Circuits and Systems, 1991*, volume 5, pages 2656–2660, June 1991.
63. D. Patel and T. J. Stonham. Texture image classification and segmentation using rank-order clustering. In *Proceedings of the 11th International Conference on Pattern Recognition (ICPR'92)*, volume 3, pages 92–95. IEEE Computer Society, 1992.
64. O. A. B. Penatti, E. Valle, and R. da Silva Torres. Comparative study of global color and texture descriptors for web image retrieval. *Journal of Visual Communication andImage Representation*, 23:359–380, 2012.
65. M. Petrou and P. García Sevilla. *Image Processing. Dealing with Texture*. Wiley Interscience, 2006.
66. M. Pietikäinen, A. Hadid, G. Zhao, and T. Ahonen. *Computer Vision Using Local Binary Patterns*, volume 40 of *Computational Imaging and Vision*. Springer, London, 2011.
67. V. K. Pothos, C. Theoharatos, E. Zygouris, and G. Economu. Distributional-based texture classification using non-parametric statistics. *Pattern Analysis and Applications*, 11:117–129, 2008.
68. F. Tajeri pour, M. Rezaei, M. Saberi, and S. F. Ershad. Texture classification approach based on combination of random threshold vector technique and co-occurrence matrixes. In *Proceedings of the International Conference on Computer Science and Network Technology, 2011 (ICCSNT)*, volume 4, pages 2303–2306, Harbin, China, December 2011.
69. J. Rouco, A. Mosquera, M. G. Penedo, M. Ortega, and M. Penas. Texture description in local scale using texton histograms with quadrature filter universal dictionaries. *IET Computer Vision*, 5(4):211–221, 2011.

70. Y. Rubner, C. Tomasi, and L. J. Guibas. A metric for distributions with applications to image databases. In *Proceedings of the Sixth International Conference on Computer Vision (ICCV'98)*, pages 59–66, 1998.
71. R. E. Sánchez-Yáñez, E. V. Kurmyshev, and A. Fernández. One-class texture classifier in the CCR feature space. *Pattern Recognition Letters*, 24(9-10):1503–1511, 2003.
72. N. Sebe and M. S. Lew. Texture features for content-based retrieval. In M. S. Lew, editor, *Principles of visual information retrieval*, pages 51–85. Springer-Verlag, London, UK, 2001.
73. F. Serratosa and A. Sanfeliu. Signatures versus histograms: Definitions,distances and algorithms. *Pattern Recognition*, 39(5):921–934, 2006.
74. G. P. Stachowiak, P. Podsiadlo, and G. W. Stachowiak. A comparison of texture feature extraction methods for machine condition monitoring and failure analysis. *Tribology Letters*, 20(2):133–147, 2005.
75. X. Tan and B. Triggs. Enhanced local texture feature sets for face recognition under difficult lighting conditions. In *Analysis and Modelling of Faces and Gestures*, volume 4778 of *Lecture Notes in Computer Science*, pages 168–182. Springer, 2007.
76. J. S. Taur and C.-W. Tao. Texture classification using a fuzzy texture spectrum and neural networks. *Journal of Electronic Imaging*, 7(1):29–35, 1998.
77. M. Tuceryan and A. K. Jain. Texture analysis. In C. H. Chen, L. F. Pau, and P. S. P. Wang, editors, *Handbook of Pattern Recognition and Computer Vision (2nd Edition)*, pages 207–248. World Scientific Publishing, 1998.
78. T. Tuytelaars and C. Schmid. Vector quantizing feature space with a regular lattice. In *Proceedings of the IEEE 11th International Conference on Computer Vision (ICCV'07)*, pages 1–8, Rio de Janeiro, Brazil, October 2007. IEEE.
79. M. Unser. Local linear transforms for texture measurements. *Signal Processing*, 11(1):61–79, July 1986.
80. K. Valkealahti and E. Oja. Reduced multidimensional co-occurrence histograms in texture classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(1):90–94, 1998.
81. M. Varma and A. Zisserman. Texture classification: Are filter banks necessary? In *Proceedings of the 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'03)*, volume 2, pages 691–698, June 2003.
82. M. Varma and A. Zisserman. Unifying statistical texture classification frameworks. *Image and Vision Computing*, 22(14):1175–1183, 2004.
83. M. Varma and A. Zisserman. A statistical approach to texture classification from single images. *International Journal of Computer Vision*, 62(1-2):61–81, 2005.
84. M. Varma and A. Zisserman. A statistical approach to material classification using image patch exemplars. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(11):2032–2047, 2009.
85. Z. Wang, B. Fan, and F. Wu. Local intensity order pattern for feature description. In *Proceedings of the 13th International Conference on Computer Vision (ICCVï¿½2011)*, pages 603–610, Barcelona, Spain, November 2011.
86. H. Wechsler. Texture analysis – A survey. *Signal Processing*, 2(3):271–282, 1980.
87. C.-M. Wu and Y.-C. Chen. Statistical feature matrix for texture analysis. *CVGIP: Graphical Models and Image Processing*, 54(5):407–419, 1992.
88. X. Xie and M. Mirmehdi. A galaxy of texture features. In M. Mirmehdi, X. Xie, and J. Suri, editors, *Handbook of texture analysis*, pages 375–406. Imperial College Press, 2008.
89. J. Xie, L. Zhang, J. You, and D. Zhang. Texture classification via patch-based sparse texton learning. In *Proceedings of the International Conference on Image Processing*, pages 2737–2740, 2010.
90. B. Xu, P. Gong, E. Seto, and R. Spear. Comparison of gray-level reduction and different texture spectrum encoding methods for land-use classification using a panchromatic Ikonos image. *Photogrammetric Engineering and Remote Sensing*, 69(5):529–536, 2003.
91. R. Zabih and J. Woodfill. Non-parametric local transforms for computing visual correspondence. In *Proceedings of the 3rd European Conference on Computer Vision (ECCV 1994)*, pages 151–158, Stockholm, Sweden, May 1994. Springer-Verlag.

92. H. Zhang, R. Rahmani, S. R. Cholleti, and S. A. Goldman. Local image representations using pruned salient points with applications to CBIR. In *Proc. of the 14th Annual ACM International Conference on Multimedia*, Santa Barbara (USA), October 2006.
93. J. Zhang, M. Marszałek, S. Lazebnik, and C. Schmid. Local features and kernels for classification of texture and object categories: A comprehensive study. *International Journal of Computer Vision*, 73(2):213–238, 2007.
94. H. Zhou, R. Wang, and C. Wang. A novel extended local-binary-pattern operator for texture analysis. *Information Sciences*, 178(22):4314–4325, 2008.