# Block-Based Methods for Image Retrieval Using Local Binary Patterns

Valtteri Takala, Timo Ahonen, and Matti Pietikäinen

Machine Vision Group, Infotech Oulu, PO Box 4500,
FI-90014 University of Oulu, Finland
{vallu, tahonen, mkp}@ee.oulu.fi,
http://www.ee.oulu.fi/mvg/

**Abstract.** In this paper, two block-based texture methods are proposed for content-based image retrieval (CBIR). The approaches use the Local Binary Pattern (LBP) texture feature as the source of image description. The first method divides the query and database images into equally sized blocks from which LBP histograms are extracted. Then the block histograms are compared using a relative $L_1$ dissimilarity measure based on the Minkowski distances. The second approach uses the image division on database images and calculates a single feature histogram for the query. It sums up the database histograms according to the size of the query image and finds the best match by exploiting a sliding search window. The first method is evaluated against color correlogram and edge histogram based algorithms. The second, user interaction dependent approach is used to provide example queries. The experiments show the clear superiority of the new algorithms against their competitors.

## 1 Introduction

Content-based image retrieval (CBIR) has gained a reasonable amount of interest in recent years. The growing number of image and video databases in the Internet and other information sources has forced us to strive after better retrieval methods. There is certainly a continuous need for novel ideas in all areas of CBIR.

While choosing feature descriptors for image retrieval we have several choices to begin with. The most common categories of descriptors are based on color, texture, and shape, and there are many alternatives in each of these. The popular color features in today's content-based image retrieval applications include color histograms [1], color correlograms [2], color moments [3] and MPEG-7 color descriptors [4]. As for the texture feature extractors, that have been under research since the late 1960s, there exist numerous methods. Many approaches like two of the MPEG-7 texture descriptors [4] are based on Gabor filtering [5]. Others put their trust on algorithms that rely on DFT transformation [6]. There are also usable features like MR-SAR [7], Wold features [8] and, of course, the old but still popular Tamura approach [9]. In addition to the previous ones, simple algorithms based on statistical feature distribution have proved to be efficient

H. Kalviainen et al. (Eds.): SCIA 2005, LNCS 3540, pp. 882–891, 2005.

in texture classification. For example, the edge histogram [10], which is a block-based descriptor included in the MPEG-7 standard, LBP [11], and its derivative LEP [12] have been in successful use.

LBP is one of the best texture methods available today. It is invariant to monotonic changes in gray-scale and fast to calculate. Its efficiency originates from the detection of different micro patterns (edges, points, constant areas etc.). LBP has already proved its worth in many applications [13], [14], [15] in which texture plays an important role. There already exist some CBIR platforms with LBP features included, but the use of the operator has been limited to the original version [11] and it has been applied on full images only [16].

Most of the current CBIR texture descriptors used in commercial systems are calculated for full images. The full image approach is well justified as it usually keeps the size of the feature database reasonably low – depending on the used features and the amount of images, of course. Still there is a problem while considering only full images. The local image areas of interest are easily left unnoticed as the global features do not contain enough information for local discrimination. A way to pay attention to local properties is to use image segmentation. However, the segmentation is usually prone to errors so it is not very suitable for images with general – in other words unknown – content. Another way to enhance the retrieval results is to apply the image extractor to the subimage areas without using any type of segmentation and compare the obtained feature descriptors separately. For instance, in [17] five constant subimage zones were used with several different features. In this paper a similar kind of approach is used, but instead of constant areas it is extended to arbitrary-sized image blocks which can be overlapping.

## 2 Texture Descriptor

### 2.1 LBP

The original LBP method [11], shown in Fig. 1, was first introduced as a complementary measure for local image contrast. It operated with eight neighboring pixels using the center as a threshold. The final LBP code was then produced by multiplying the thresholded values by weights given by powers of two and adding the results in a way described by Fig. 1. By definition, LBP is invariant to any monotonic transformation of the gray scale and it is quick to compute.

The original LBP has been extended to a more generalized approach [18] in which the number of neighboring sample points is not limited. In Fig. 2 three different LBP operators are given. The predicate (radius, $R$) has no constraints like in the original version and the samples ($P$) that do not fall on exact pixel positions are interpolated by using bi-linear interpolation.

With larger neighborhoods, the number of possible LBP codes increases exponentially. This can be avoided, to some extent, by considering only a subset of the codes. One approach is to use so called uniform patterns [18] representing the statistically most common LBP codes. With them the size of the feature

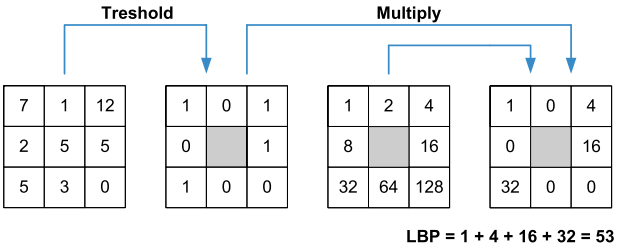


**Fig. 1.** The original LBP

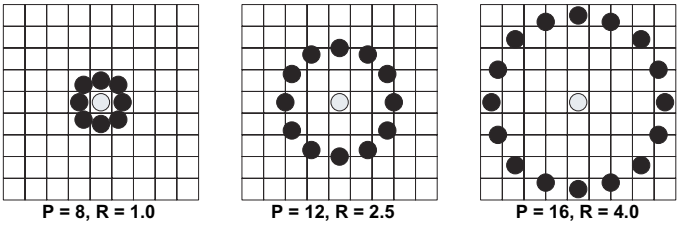


**Fig. 2.** The general $\mathrm{LBP}_{P,R}$ with three different circularly symmetric neighbor sets, where subscript $P$ stands for the number of samples and $R$ for the sampling radius

histogram generated by an LBP operator can be reduced without significant loss in its discrimination capability. For example, if we consider only those LBP codes that have U value of 2 (U refers to the measure of uniformity, that is the number of 0/1 and 1/0 transitions in the circular binary code pattern), in case of a $3 \times 3$ operator ($\mathrm{LBP}^{u2}_{8,1}$) we get a feature vector of 58 bins instead of original 256 bins. When the remaining patterns are accumulated to a single bin the histogram becomes 59. That is only a fraction (59/256) of the original.

The spatial support area of LBP feature extractor can be extended by using operators with different radii and sample counts and combining the results [18]. By utilizing N operators we get N different LBP codes which can be connected to form a single feature descriptor vector of N codes. While inserting the marginal distributions of feature extractors one after another, the distance between the sample and model is given by Eq. 1:

$$L_N = \sum_{n=1}^{N} L\left(S^n, M^n\right), \tag{1}$$

where $S^n$ and $M^n$ are the sample and model distributions extracted by the $n$th operator.

### 2.2 Nonparametric Dissimilarity Measure

A distance function is needed for comparing images through their LBP features. There are many different dissimilarity measures [19] to choose from. Most of the

LBP studies have favored a nonparametric log-likelihood statistic as suggested by Ojala et al. [18]. In this study, however, a relative $L_1$ measure similar to the one proposed by Huang et al. [2] was chosen due to its performance in terms of both speed and good retrieval rates when compared to the log-likelihood and other available statistics. In the initial tests the log-likelihood and relative $L_1$, which were clearly better than the rest, produced even results but the calculation of relative $L_1$ measure took only a third of the time required by log-likelihood. The dissimilarity measure is given in Eq. 2, where $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ represent the feature histograms to be compared and subscript $i$ is the corresponding bin.

$$L_1^{relative}(\boldsymbol{x}_1, \boldsymbol{x}_2) = \sum_i \frac{\left| x_{1,i} - x_{2,i} \right|}{x_{1,i} + x_{2,i}} \tag{2}$$

## 3 Block-Based CBIR

### 3.1 The Block Division Method

The block division method is a simple approach that relies on subimages to address the spatial properties of images. It can be used together with any histogram descriptors similar to LBP. The method works in the following way: First it divides the model images into square blocks that are arbitrary in size and overlap. Then the method calculates the LBP distributions for each of the blocks and combines the histograms into a single vector of sub-histograms representing the image. In the query phase the same is done for the query image(s) after which the query and model are compared by calculating the distance between each sub-histogram of the query and model. The final image dissimilarity $D$ for classification is the sum of minimum distances as presented by Eq. 3:

$$D = \sum_{i=0}^{N-1} min_j(D_{i,j}), \tag{3}$$

where $N$ is the total amount of query image blocks and $D_{i,j}$ the distance (relative $L_1$) between the $i$th query and $j$th model block histograms. An example of the approach in operation is shown in Fig. 3. Note, that in this figure the shown histograms are only examples and not the actual LBP distributions of corresponding image blocks.

### 3.2 The Primitive Blocks

Another way to utilize image blocks is to use small constant-sized elements referred here as primitive blocks. Instead of larger and less adaptive equivalents, the primitive blocks can be combined to match the size of the query image with reasonable accuracy and speed as there is no heavy processing involved like in the pixel-by-pixel sliding window methods. In this approach the model images are handled as in the previous method but the query images are left untouched

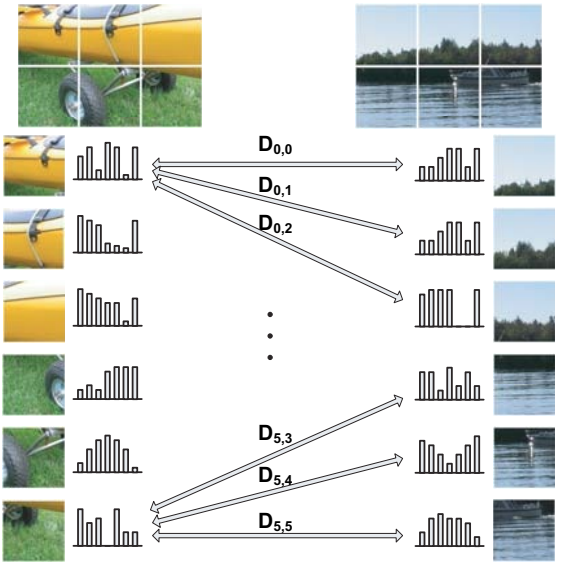


**Fig. 3.** The block division method

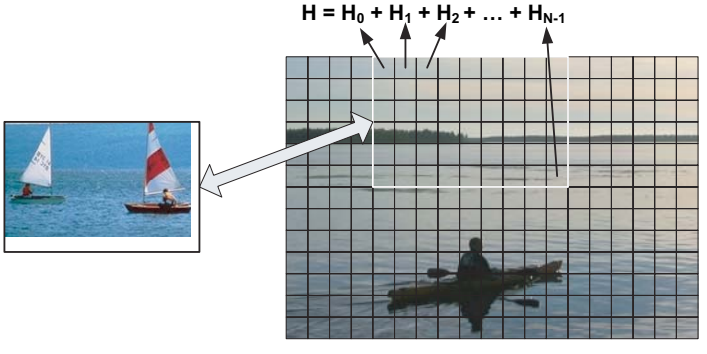


**Fig. 4.** The primitive blocks approach

and only a global LBP histogram is produced for each of them. The model's sub-histograms $\boldsymbol{H}_i$ are summed up to a single feature histogram according to

$$\boldsymbol{H} = \sum_{i=0}^{N-1} \boldsymbol{H}_i, \tag{4}$$

by first adapting to the size of the query image, and then they are normalized. The primitive blocks (actually the corresponding block histograms) are connected in the way depicted in Fig. 4, where the search window goes through the whole model image and does the matching by using the distance measure of

Eq. 2. The size of the search window is the same or a bit larger, depending on the chosen block size, than the area dictated by the query image dimensions.

While using primitive blocks, there exist two types of overlapping. The blocks themselves can overlap, as in the case of previous block division method, and then the measure of overlapping is determined in single pixels. The same applies to the search window areas consisting of primitive blocks but in their case the overlap is quantified to the dimensions of the used primitive blocks.

## 4 Experiments

### 4.1 Test Database and Configurations

Both image retrieval methods were tested on a database consisting of commercial Corel Image Gallery [20] images of sizes 384×256 and 256×384. The image categorization was set according to the original image database structure of the Corel set, so there were 27 categories of 50 images each making up 1350 images in total. No further categorization was utilized. This kind of categorization may sound rude but it was used to ensure the reproducibility of the tests. The following categories (physical database folder names) where chosen from the image gallery: Apes, Bears, Butterflies, Cards, Death Valley, Dogs, Elephants, Evening Skies, Fancy Flowers, Fireworks, Histology, Lighthouses, Marble Textures, Night Scenes, Owls, Rhinos and Hippos, Roads and Highways, Rome, Skies, Snakes Lizards and Salamanders, Space Voyage, Sunsets Around the World, Tigers, Tools, Waterscapes, Wildcats, and Winter. Some example images are shown in Fig. 5.

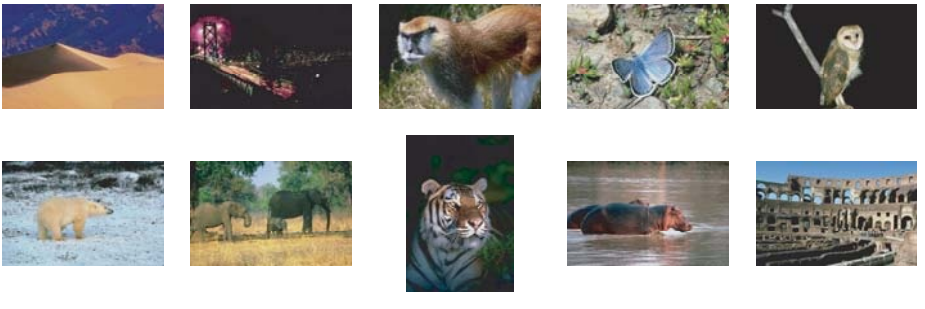

**Fig. 5.** Image examples from the Corel Gallery database

The category experiments were carried on five different image categories (Apes, Death Valley, Fireworks, Lighthouses, and Tigers), so there were 250 queries per experiment. Two different image feature descriptors, one based on color and the other one on texture, were chosen to be compared to the queries attained with LBP operators. The first one of them was the color correlogram [2], which is still one of the most powerful color descriptors, and the other one

was the edge histogram [10], that operates with image blocks and is included in the MPEG-7 Visual Stardard [4]. The correlogram was applied with four distances (1, 3, 5, and 7) and four quantization levels per color channel, that is 64 quantization levels in total. The edge histogram used the standard parameters as used by Park et al. [21], thus the method produced histograms of 80 bins.

LBP was applied both on full images and image blocks of sizes 128×128 and 96×96. Two clearly different LBP operators were tried out: one using eight uninterpolated samples and a predicate of 1 ($\mathrm{LBP}^{u2}_{8.1}$) and a multiresolution version with three different radii and eight interpolated samples in each ($\mathrm{LBP}^{u2}_{8.1} + ^{u2}_{8,2.4} + ^{u2}_{8,5.4}$). Both operators relied on uniform patterns with U value of 2 (u2), so the corresponding histograms had 59 and 177 bins, respectively.

### 4.2 Results

The results of the experiments are shown in Table 1. The used measures are precision (the ratio between the correct and all retrieved images) and recall (the ratio between the correct retrieved images and all correct images in the

**Table 1.** The results (precision/recall) for different methods

| Method | Block size(overlap) | 10 images(%) | 25 images | 50 images |
|---|---|---|---|---|
| Color Correlogram | full image | 37.8/7.5 | 25.3/12.7 | 18.7/18.7 |
| Edge Histogram | image dependent | 26.3/5.3 | 18.4/9.2 | 14.2/14.2 |
| $\mathrm{LBP}^{u2}_{8.1}$ | full image | 34.7/6.9 | 24.6/12.3 | 19.0/19.0 |
| $\mathrm{LBP}^{u2}_{8.1} + ^{u2}_{8,2.4} + ^{u2}_{8,5.4}$ | full image | 36.9/7.5 | 25.3/12.7 | 18.7/18.7 |
| $\mathrm{LBP}^{u2}_{8.1}$ | $128 \times 128(0 \times 0)$ | 36.9/7.4 | 26.5/13.2 | 21.3/21.3 |
| $\mathrm{LBP}^{u2}_{8.1} + ^{u2}_{8,2.4} + ^{u2}_{8,5.4}$ | $128 \times 128(0 \times 0)$ | 37.4/7.5 | 26.6/13.3 | 20.4/20.4 |
| $\mathrm{LBP}^{u2}_{8.1}$ | $128 \times 128(64 \times 64)$ | 43.0/8.6 | 31.3/15.7 | 23.7/23.7 |
| $\mathrm{LBP}^{u2}_{8.1} + ^{u2}_{8,2.4} + ^{u2}_{8,5.4}$ | $128 \times 128(64 \times 64)$ | 43.3/8.7 | 31.0/15.5 | 24.0/24.0 |
| $\mathrm{LBP}^{u2}_{8.1}$ | $96 \times 96(0 \times 0)$ | 40.5/8.1 | 29.0/14.5 | 22.5/22.5 |
| $\mathrm{LBP}^{u2}_{8.1} + ^{u2}_{8,2.4} + ^{u2}_{8,5.4}$ | $96 \times 96(0 \times 0)$ | 41.0/8.2 | 29.2/14.6 | 21.5/21.5 |
| $\mathrm{LBP}^{u2}_{8.1}$ | $96 \times 96(48 \times 48)$ | 45.5/9.1 | 31.6/15.8 | 24.0/24.0 |
| $\mathrm{LBP}^{u2}_{8.1} + ^{u2}_{8,2.4} + ^{u2}_{8,5.4}$ | $96 \times 96(48 \times 48)$ | 45.3/9.0 | 32.5/16.2 | 23.4/23.4 |

database), and the numbers are marked as percentages. The LBPs using overlapping blocks achieve precision rates of about 45 % (over 9 % for recall) for 10 images and are clearly better than any of the other extractors. Their results for 50 images are almost as good as those obtained with the best full image operators for 25 images. Using blocks, especially overlapping ones, instead of full images seems to make a clear difference. It is also to be noticed that using overlap has a large impact regardless of the block size. Several percentages are gained through 50 % overlapping.

The test with the primitive block approach was performed with an $\mathrm{LBP}^{u2}_{8.1}$ operator without interpolation. Fig. 6 shows an example query obtained by using

$32 \times 32$ sized primitive blocks with overlap of two pixels (the overlap between search windows was set to 50 %). The query images have been taken from an original database image and they have been outlined in such a way that they are composed of mostly homogeneous texture. The actual retrieval task was not an easy one: two subimages were needed to get satisfying results, that is seven out of 16 images from the right category (Lighthouses). The chosen subimages have both very distinctive appearance but the image of a rock appeared to have more positive influence on the outcome than the carefully outlined picture of the white lighthouse itself. The probable reason for this is the clear distinction in the properties of the two subimages – the image of the rock is rich in its texture content.

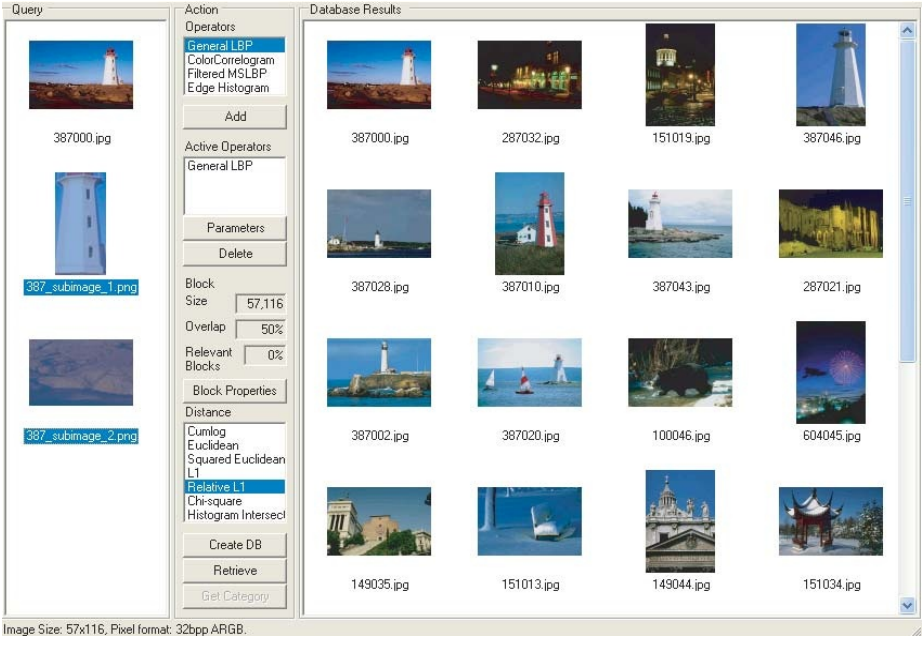


**Fig. 6.** A test query. The left box of images in the user interface is the query group from which only the outlined (darkened) subimages were selected for the eventual search

Some additional tests were also conducted with a stamp database consisting of about 900 German stamp images [22]. A couple of LBP extractors were used and their performance was evaluated against a commercial stamp image retrieval software. The block division method fared at least as well or even better than the matured retrieval application making use of multiple different image features (color, texture, motive, and image size and aspect ratios).

## 5 Conclusions

In this paper, we considered the use of LBP texture features combined with two different block-based image division methods. The results obtained show that the LBP can be successfully used to retrieve images with general content as it is fast to extract and it has useful qualities like invariance to monotonic transitions in gray scale and small descriptor size. The color correlogram, that represents the current state of the art in CBIR, was clearly outperformed by one of the developed subimage approaches.

The increased retrieval rates of the tested methods come at the expense of higher computational demands. The time needed for query grows linearly with the amount of used image blocks. With large images and small block sizes the required processing capacity slips easily out of the grasp of applications that have real-time requirements. Still, it should be noted that it does not seem to be necessary to use large numbers of small blocks as, according to the obtained results, a few blocks per image is usually enough to make a considerable difference when compared to descriptors calculated for full images.

The method based on primitive blocks was hard to assess as there is a level of user interaction involved in the query procedure. Nevertheless, it has some important properties that increase its value in the field of CBIR: It is faster than conventional search window approaches as it does not extract features for every possible search window size separately. Another noteworthy feature is that it can be used to find objects consisting of a single texture or larger entities with several different areas of interest as the query can be adjusted by using more than one sample image.

For the future studies, there are many ways that could enhance the performance and speed of the studied methods. For instance, different block matching algorithms, like the three-step search method, could be used to speed up the matching process. Another possibility could be to use image blocks that are weighted according to their spatial positions. In the case of multiresolution LBP, the use of weights could be extended to emphasize the LBPs containing the most relevant texture information. These and other enhancements could improve the usability of LBP features in the CBIR of the future.

**Acknowledgments.** This study was funded in part by the Academy of Finland.

## References

1. Swain M., Ballard D.: Color Indexing. In: Third International Conference on Computer Vision. (1990) 11–32
2. Huang J., Kumar S. R., Mitra M., Zhu W.-J., Zabih R.: Image Indexing Using Color Correlograms. In: IEEE Conference on Computer Vision and Pattern Recognition (1997) 762–768
3. Stricker M., Orengo M.: Similarity of Color Images In: SPIE Conference on Storage and Retrieval for Image and Video Databases (1995) 381–392

4. Manjunath B. S., Ohm J.-R., Vasudevan V., Yamada A.: Color and Texture Descriptors. IEEE Transactions on Circuits and Systems for Video Technology **11** (2001) 703–715
5. Manjunath B. S., Ma W. Y.: Texture Features for Browsing and Retrieval of Image Data. In: IEEE Transactions on Pattern Analysis and Machine Intelligence **18** (1996) 837–842
6. Sim D.-G., Kim H.-K., Oh D.-H.: Translation, Rotation, and Scale Invariant Texture Descriptor for Texture-Based Image Retrieval. In: International Conference on Image Processing (2000) 742–745
7. Mao J., Jain A.: Texture Classification and Segmentation Using Multiresolution Simultaneous Autoregressive Models. Pattern Recognition **25** (1992) 173–188
8. Francos J. M., Meiri A. Z., Porat B.: On a Wold-Like Decomposition of 2-D Discrete Random Fields. In: International Conference on Acoustics, Speech, and Signal Processing (1990) 2695–2698
9. Tamura H., Mori S., Yamawaki T.: Textural Features Corresponding to Visual Perception. IEEE Transactions on Systems, Man, and Cybernetics **8** (1978) 460–473
10. Park S. J., Park D. K., Won C. S.: Core Experiments on MPEG-7 Edge Histogram Descriptor ISO/IEC JTC1/SC29/WG11 - MPEG2000/M5984 (2000)
11. Ojala T., Pietikäinen M., Hardwood D.: A Comparative Study of Texture Measures with Classification Based on Feature Distribution. Pattern Recognition **29** (1996) 51–59
12. Yao C.-H., Chen S.-Y.: Retrieval of Translated, Rotated and Scaled Color Textures. Pattern Recognition **36** (2003) 913–929
13. Ahonen T., Hadid A., Pietikäinen M.: Face Recognition with Local Binary Patterns Lecture Notes in Computer Science, Vol. 3021. Springer-Verlag, Berlin Heidelberg New York (2004) 469–481
14. Mäenpää T., Turtinen M., Pietikäinen M.: Real-Time Surface Inspection by Texture. Real-Time Imaging **9** (2003) 289–296
15. Pietikäinen M., Nurmela T., Mäenpää T., Turtinen M.: View-Based Recognition of Real-World Textures. Pattern Recognition **37** (2004) 313–323
16. Veltkamp R. C., Tanase M.: Content-Based Image Retrieval Systems: A Survey. Revised and extended version of Technical Report UU-CS-2000-34, Department of Computing Science, Utrecht University (2002)
17. Laaksonen J., Koskela M., Oja E.: PicSOM: Self-Organizing Maps for Content-Based Image Retrieval. In: IEEE International Joint Conference on Neural Networks (1999) 2470–2473
18. Ojala T., Pietikäinen M., Mäenpää T.: Multiresolution Gray-Scale and Rotation Invariant Texture Classification with Local Binary Patterns. IEEE Transactions on Pattern Analysis and Machine Intelligence **24** (2002) 971–987
19. Puzicha J., Rubner Y., Tomasi C., Buhmann J. M.: Empirical Evaluation of Dissimilarity Measures for Color and Texture. In: Seventh IEEE International Conference on Computer Vision (1999) 1165–1172
20. Corel Corporation. URL: http://www.corel.com/ (2005)
21. Park D. K., Jeon Y. S., Won C. S.: Efficient Use of Local Edge Histogram Descriptor. In: 2000 ACM workshop on Standards, Interoperability and Practices (2000) 52–54
22. Takala V.: Local Binary Pattern Method in Context-Based Image Retrieval. Master's thesis, Department of Electrical and Information Engineering, University of Oulu (2004) (in Finnish)