Description of Interest Regions with Local Binary Patterns

Marko Heikkilä[a,*], Matti Pietikäinen[a], Cordelia Schmid[b]

[a]*Machine Vision Group, Infotech Oulu and Department of Electrical and Information Engineering, PO Box 4500, FI-90014, University of Oulu, Finland*

[b]*INRIA Grenoble, 655 Avenue de l'Europe, 38330 Montbonnot, France*

Abstract

This paper presents a novel method for interest region description. We adopted the idea that the appearance of an interest region can be well characterized by the distribution of its local features. The most well-known descriptor built on this idea is the SIFT descriptor that uses gradient as the local feature. Thus far, existing texture features are not widely utilized in the context of region description. In this paper, we introduce a new texture feature called center-symmetric local binary pattern (CS-LBP) that is a modified version of the well-known local binary pattern (LBP) feature. To combine the strengths of the SIFT and LBP, we use the CS-LBP as the local feature in the SIFT algorithm. The resulting descriptor is called the CS-LBP descriptor. In the matching and object category classification experiments, our descriptor performs favorably compared to the SIFT. Furthermore, the CS-LBP descriptor is computationally simpler than the SIFT.

Key words:
Region description, region detection, local binary patterns, SIFT, image matching, object recognition

1 Introduction

Local image feature detection and description have received a lot of attention in recent years. The basic idea is to first detect interest regions that are covariant to a class of transformations. Then, for each detected region, an invariant descriptor is built. Once we have the descriptors computed, we can match interest regions between images. This approach has many advantages. For example, local features can be made very tolerant to illumination changes, perspective distortions, image blur, image zoom, and so on. The approach is also very robust to occlusion. Local features have performed very well in many computer vision applications, such as image retrieval [1], wide baseline matching [2], object recognition [3], texture recognition [4], and robot localization [5].

The interest regions that are used as input to region description methods are provided by the interest region detectors. Many different approaches to region detection have been proposed. For example, some detectors detect corner-like regions while others extract blobs. Since this paper focuses on interest region description, we refer the reader to [6] for more information on interest region detection.

As with the interest region detection, many different approaches to interest region description have been proposed. The methods emphasize different image

* Corresponding author. Tel.: +358 8 553 2996; fax: +358 8 553 2612.
Email address: `markot@ee.oulu.fi` (Marko Heikkilä).

properties such as pixel intensities, color, texture, and edges. Many of the proposed descriptors are distribution-based, i.e. they use histograms to represent different characteristics of appearance or shape. The *intensity-domain spin image* [4] is a 2D histogram where the dimensions are the distance from the center point and the intensity value. The *SIFT* descriptor [3] is a 3D histogram of gradient locations and orientations where the contribution to the location and orientation bins is weighted by the gradient magnitude and a Gaussian window overlaid over the region. Very similar to the SIFT descriptor is the the *GLOH* descriptor [7], which replaces the Cartesian location grid used by the SIFT with a log-polar one, and applies PCA to reduce the size of the descriptor. Another SIFT-like descriptor using log-polar location grid is the extension to the *shape context* presented in [7], which is a 3D histogram of edge point locations and orientations. Original shape context was computed only for edge point locations and not for orientations [8]. The *SURF* descriptor [9] builds on the strengths of the leading existing detectors and descriptors. It uses a Hessian matrix-based measure for the detector and Haar wavelet responses for the descriptor. By relying on integral images for image convolutions, computation time is significantly reduced. In [10], geodesic sampling is used to get neighborhood samples for interest points and then a geodesic-intensity histogram (GIH) is used as a deformation invariant local descriptor. Other interest region descriptors proposed in the literature include *PCA-SIFT* [11], *steerable filters* [12], *moment invariants* [13], and *complex filters* [14].

There exist several recent comparative studies on region descriptors [15,7,16]. Almost without an exception, the best results are reported for distribution-based descriptors such as SIFT. Recently, in [17], an interesting study on region descriptors was published. The authors break up the descriptor extraction process into a number of modules and put these together in different combinations. Many of these combinations give rise to published descriptors such as the SIFT but many are untested. Furthermore, learning is used to optimize the choice of parameter values for each candidate descriptor algorithm.

Many existing texture operators have not been used for describing interest regions so far [18]. One reason might be that, by using these methods, usually a large number of dimensions is required to build a reliable descriptor. The *local binary pattern* (LBP) texture operator [19–21], has been highly successful for various computer vision problems such as face recognition [22], background subtraction [23], and recognition of 3D textured surfaces [24], but it has not been used for describing interest regions so far. The LBP has properties that favor its usage in interest region description such as tolerance against illumination changes and computational simplicity. Drawbacks are that the operator produces a rather long histogram and is not too robust on flat image areas. To address these problems, in this paper, we propose a new LBP-based texture feature, denoted as *center-symmetric local binary pattern* (CS-LBP) that is more suitable for the given problem.

Since the SIFT and other distribution-based descriptors similar to it [8,7,9] have shown state-of-the-art performance in different problems, we decided to focus on this approach. We were especially interested to see if the gradient orientation and magnitude based feature used in the SIFT algorithm could be replaced by a different feature that offers better or comparable performance. In this paper, we propose a new interest region descriptor, denoted as *CS-LBP descriptor*, that combines the good properties of the SIFT and LBP. This is achieved by adopting the SIFT descriptor and using the novel CS-LBP feature instead of original gradient feature. The new feature allows simplifications of several steps of the algorithm which makes the resulting descriptor computationally simpler than SIFT. It also appears to be more robust to illumination changes than the SIFT descriptor. A preliminary version of this article has appeared in [25].

The rest of the paper is organized as follows. In Section 2, we first briefly describe the starting point for our work, i.e. the SIFT and LBP methods. Sections 3 and 4 give details for the CS-LBP operator and the CS-LBP descriptor, respectively. The experimental evaluation is carried out in Section 5. Finally, we conclude the paper in Section 6.

2 SIFT and LBP Methods

Before presenting in detail the CS-LBP operator and the CS-LBP descriptor, we give a brief review of the SIFT and LBP methods that form the basis for our work.

2.1 SIFT Descriptor

The SIFT descriptor is a 3D histogram of gradient locations and orientations. Location is quantized into a 4×4 location grid and the gradient angle is quantized into 8 orientations, resulting in a 128-dimensional descriptor. First, the gradient magnitudes and orientations are computed within the interest region. The gradient magnitudes are then weighted with a Gaussian window overlaid over the region. To avoid boundary effects in the presence of small shifts of the interest region, a trilinear interpolation is used to distribute the value of each gradient sample into adjacent histogram bins. The final descriptor is obtained by concatenating the orientation histograms over all bins. To reduce the effects of illumination change the descriptor is first normalized to unit length. Then, the influence of large gradient magnitudes is reduced by thresholding the descriptor entries, such that each one is no larger than 0.2, and renormalizing to unit length.

2.2 LBP Operator

The local binary pattern (LBP) is a powerful illumination invariant texture primitive. The histogram of the binary patterns computed over a region is used for texture description. The operator describes each pixel by the relative graylevels of its neighboring pixels, see Fig. 1 for an illustration with 8 neighbors. If the graylevel of the neighboring pixel is higher or equal, the value is set to one, otherwise to zero. The descriptor describes the result over the neighborhood as a binary number (binary pattern):

$$LBP_{R,N}(x,y) = \sum_{i=0}^{N-1} s(n_i - n_c)2^i, \quad s(x) = \begin{cases} 1 & x \geq 0 \\ 0 & otherwise \end{cases}, \tag{1}$$

where n_c corresponds to the graylevel of the center pixel of a local neighborhood and n_i to the graylevels of N equally spaced pixels on a circle of radius R. Since correlation between pixels decreases with distance, a lot of the texture information can be obtained from local neighborhoods. Thus, the radius R is usually kept small. In practice, (1) means that the signs of the differences in a neighborhood are interpreted as a N-bit binary number, resulting in 2^N distinct values for the binary pattern. From above, it's easy to figure out that the LBP has several properties that favor its usage in interest region description. The features have proven to be robust against illumination changes, they

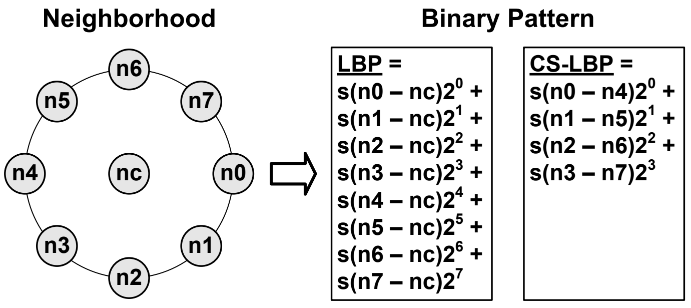


Fig. 1. LBP and CS-LBP features for a neighborhood of 8 pixels.

are very fast to compute, and do not require many parameters to be set [21].

3 Center-Symmetric Local Binary Patterns

The LBP operator, described in Section 2.2, produces rather long histograms and is therefore difficult to use in the context of a region descriptor. To address the problem we modified the scheme of how to compare the pixels in the neighborhood. Instead of comparing each pixel with the center pixel, we compare center-symmetric pairs of pixels as illustrated in Fig. 1. This halves the number of comparisons for the same number of neighbors. We can see that for 8 neighbors, LBP produces 256 (2^8) different binary patterns, whereas for CS-LBP this number is only 16 (2^4). Furthermore, robustness on flat image regions is obtained by thresholding the graylevel differences with a small value T as proposed in [23]:

$$CS-LBP_{R,N,T}(x,y) = \sum_{i=0}^{(N/2)-1} s(n_i - n_{i+(N/2)})2^i,\ s(x) = \begin{cases} 1 & x > T \\ 0 & otherwise \end{cases},(2)$$

where n_i and $n_{i+(N/2)}$ correspond to the grayvalues of center-symmetric pairs of pixels of N equally spaced pixels on a circle of radius R. It should be noticed that the CS-LBP is closely related to gradient operator, because like some gradient operators it considers graylevel differences between pairs of opposite pixels in a neighborhood. Since the focus of this paper is in the region description we do not present any operator level comparison between the LBP and CS-LBP. Instead, the comparison is done in the context of the region descriptor in Section 5. In addition to our work, there also exists other ways to reduce the histogram size of the LBP. Maybe the best known method is to use *uniform patterns* proposed in [26]. For N neighbors we have $N(N-1)+2$ different uniform patterns. This makes 58 patterns for 8 neighbors. For a great source of information on LBP related research, the reader is referred to [21].

4 CS-LBP Descriptor

In the following, we present our CS-LBP descriptor in detail. The input for the descriptor is a normalized interest region. The process is depicted in Figure 2. The region detection and normalization steps are described in Section 5.1. In our experiments, the region size after normalization is fixed to 41×41 pixels and the pixel values lie between 0 and 1.

4.1 Feature Extraction with Center-Symmetric Local Binary Patterns

We extract a feature for each pixel of the input region by using the CS-LBP operator introduced in Section 3. The operator has 3 parameters: radius R, number of neighboring pixels N, and threshold on the graylevel difference T. Our experiments have shown that good values for these parameters are in general $\{1, 2\}$ for R, $\{6, 8\}$ for N, and $\{0, ..., 0.02\}$ for T. We believe that these values provide a good starting point for parameter value selection in

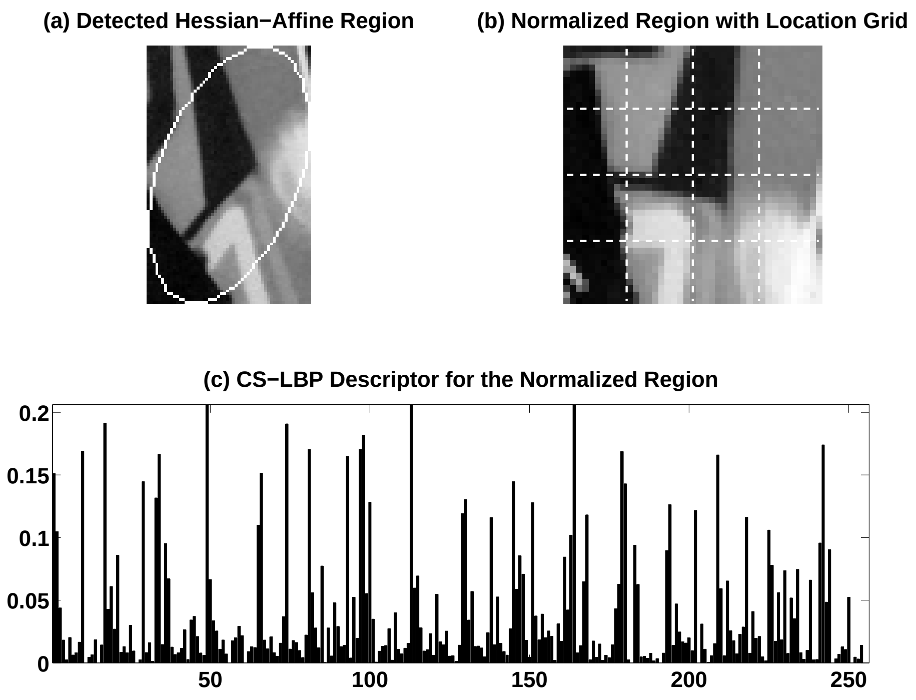


Fig. 2. The CS-LBP descriptor. (a) An elliptical image region detected by Hessian-Affine detector. (b) The region with Cartesian location grid after affine normalization. (c) The resulting CS-LBP descriptor computed for the normalized region.

other applications too.

4.2 Feature Weighting

A weight is associated with each pixel of the input region based on the used feature. A comparison of three different weighting strategies, namely *uniform*, *Gaussian-weighted gradient magnitude (SIFT)*, and *Gaussian*, showed that simple uniform weighting is the most suitable choice for the CS-LBP feature. In other words, the feature weighting step can be omitted in our case.

4.3 Descriptor Construction

In order to incorporate spatial information into the descriptor, the input region is divided into cells with a location grid. We tried two different grids, namely *Cartesian* and *log-polar*, and found that the Cartesian one gives better performance. In the experiments presented in this paper, we use either a 3×3 (9 cells) or 4×4 (16 cells) Cartesian grid. For each cell a CS-LBP histogram is built. Thus, the resulting descriptor is a 3D histogram of CS-LBP feature locations and values. As explained earlier, the number of different feature values ($2^{N/2}$) depends on the neighborhood size (N) of the chosen CS-LBP operator. In order to avoid boundary effects in which the descriptor abruptly changes as a feature shifts from one cell to another, bilinear interpolation over x and y dimensions is used to share the weight of the feature between 4 nearest cells. The share for a cell is determined by the bilinear interpolation weights. We do not interpolate over feature value dimension because the CS-LBP feature is quantized by its nature.

4.4 Descriptor Normalization

The final descriptor is built by concatenating the feature histograms computed for the cells to form a $M \times M \times 2^{N/2}$-dimensional vector, where the M and N are the grid size and CS-LBP neighborhood size, respectively. For $(M = 3, N = 6)$, $(M = 3, N = 8)$, $(M = 4, N = 6)$, and $(M = 4, N = 8)$ the lengths of the CS-LBP descriptors are 72, 144, 128, and 256, respectively. The descriptor is then normalized to unit length. The influence of very large descriptor elements is reduced by thresholding each element to be no larger than a threshold. This means that the distribution of features has greater emphasis than individual large values. After empirical testing we fixed the threshold to 0.2 which is exactly the same value as used in the SIFT algorithm. Finally, the descriptor is renormalized to unit length.

4.5 Computational Complexity

In this section, we show that the computational complexity of the CS-LBP descriptor is less than that of the SIFT descriptor. The fact that the two descriptors consist of similar steps allows us to make the comparison one step at a time. Table 1 shows computation times for two CS-LBP descriptors ($CS - LBP_{2,8,0.01}$ and $CS - LBP_{2,6,0.01}$) and the SIFT descriptor. The feature weighting step is combined into the feature extraction step since, in our SIFT implementation, the two steps share some computations. According to the results, our descriptor is on average 2.3 or 3.2 times faster than the SIFT, depending on the used CS-LBP operator. For the CS-LBP descriptors, 4×4 grid was used. The input for a region description algorithm is an interest region extracted by a region detector. Here, we have assumed that the detection and description steps are completely separated so that the description algorithm cannot reuse computations from the detection algorithm. This should be reasonable assumption since region detection and normalization can be done by using many different features and methods. In the experiments, the input region size was fixed to 41×41 pixels. Next, a detailed description on computations involved in different steps, namely feature extraction, feature weighting, descriptor construction, and descriptor normalization, is given.

Feature Extraction The feature used by the SIFT descriptor is gradient orientation. In Figure 1, if we set the radius of the neighborhood to 1, the gradient orientation for the center pixel n_c equals to $tan^{-1}((n_2 - n_6)/(n_0 - n_4))$. The CS-LBP feature used by the CS-LBP descriptor is computed using (2). It's easy to see that the CS-LBP feature needs only simple arithmetic operations while the gradient orientation requires time consuming inverse tangent computation.

Feature Weighting. In the case of SIFT descriptor the weight is Gaussian-weighted gradient magnitude, $\sqrt{(n_0 - n_4)^2 + (n_2 - n_6)^2} * e^{-(x^2+y^2)/2\sigma^2}$, where x and y are the coordinates of the center pixel g_c. It should be noticed that this requires time consuming square root computation. In the case of CS-LBP descriptor the weight is 1 since we are using uniform weighting.

Descriptor Construction. The dimensions of the SIFT descriptor are x, y, and *gradient orientation*, quantized into 4, 4, and 8 bins, respectively. The descriptor is constructed by distributing the weight of each pixel into adjacent histogram bins. The share for a bin is determined by triliear interpolation weights. This means that each weight is shared between 8 bins. Correspondingly, the dimensions of the CS-LBP descriptor are x, y, and *CS-LBP value*. The difference to the SIFT descriptor is that the interpolation is needed only for x and y dimensions. This is because the *CS-LBP value* is quantized by its nature. Thus each weight is shared between 4 bins. It should be clear that bilinear interpolation is computationally more efficient than trilinear one.

Descriptor Normalization. This step is identical for the two descriptors.

From Table 1, we see that the most time consuming step is the feature extraction (includes weight calculation). The simplicity of the CS-LBP operator over the gradient one is clearly shown in the computation times. Also, the computation times for descriptor construction and normalization steps are in conformance with the above discussion. The reason why the two CS-LBP descriptors give different results is that the other produces twice as long descriptor as the other.

Table 1

Computation times, in milliseconds, for $CS - LBP_{2,8,0.01}$, $CS - LBP_{2,6,0.01}$, and SIFT descriptors. The corresponding descriptor sizes are 256, 128, and 128. The feature weights are computed during the *Feature Extraction* step.

	Feature Extraction	**Descriptor Construction**	**Descriptor Normalization**	**Total**
$CS - LBP_{2,8,0.01}$	0.1609	0.0961	0.0070	0.2640
$CS - LBP_{2,6,0.01}$	0.1148	0.0749	0.0022	0.1919
$SIFT$	0.4387	0.1654	0.0025	0.6066

5 Experimental Evaluation

We use two well-known protocols to evaluate the proposed CS-LBP descriptor. Both are freely available on the Internet. The first protocol is a matching protocol that is designed for matching interest regions between a pair of images [27]. The second protocol is the *PASCAL Visual Object Classes Challenge 2006* protocol which is an object category classification protocol [28]. We compare the performance of the CS-LBP descriptor to the state-of-the-art descriptor SIFT. This allows us to evaluate how the CS-LBP feature compares to the gradient one. Next, we first explain the interest region detection methods used in the experiments, and then describe in detail the test protocols.

5.1 Interest Region Detection and Normalization

The interest region detectors extract the regions which are used to compute the descriptors. In the experiments, we use four different detectors:

Hessian-Affine (HesAff), *Harris-Affine (HarAff)*, *Hessian-Laplace (HesLap)*, and *Harris-Laplace (HarLap)* [6,29]. The HesAff and HarAff detectors output different types of image structures. HesAff detects blob-like structures while HarAff looks for corner-like structures. Both detectors output elliptic regions of varying size determined by the detection scale. Before computing the descriptors, the detected regions are mapped to a circular region of constant radius to obtain scale and affine invariance. Rotation invariance, if required by the application, is obtained by rotating the normalized regions in the direction of the dominant gradient orientation, as suggested in [3]. HesLap and HarLap are the scale invariant versions of the HesAff and HarAff detectors, respectively. They differ from the affine invariant detectors in that they omit the *affine adaptation* step [29]. In the experiments, the normalized region size is fixed to 41×41 pixels. For region detection, normalization, and computing SIFT descriptors we use the software routines provided by the matching protocol explained in the next subsection [27].

5.2 Image Matching

The protocol is available on the Internet together with the test data [27]. The test data contains images with different geometric and photometric transformations and for different scene types. Six different transformations are evaluated: *viewpoint change*, *scale change*, *image rotation*, *image blur*, *illumination change*, and *JPEG compression*. The two different scene types are *structured* and *textured* scenes. These test images are shown in Fig. 3. The images are either of planar scenes or the camera position was fixed during acquisition. The images are, therefore, always related by a homography (included in the

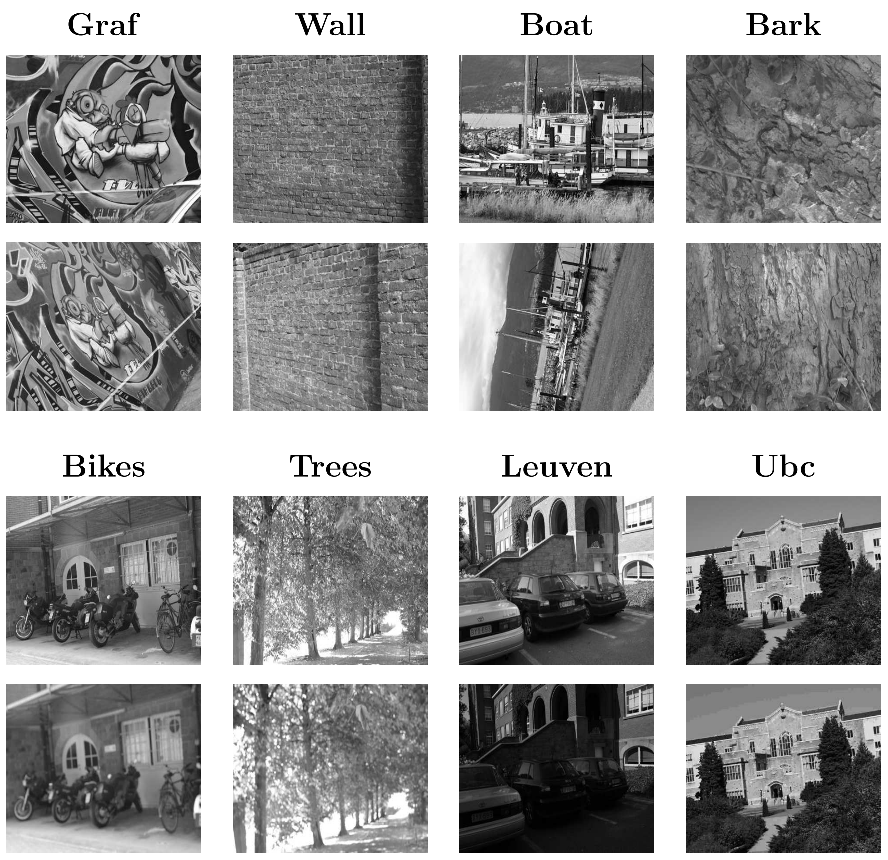


Fig. 3. Test images: **Graf** (viewpoint change, structured scene), **Wall** (viewpoint change, textured scene), **Boat** (scale change + image rotation, structured scene), **Bark** (scale change + image rotation, textured scene), **Bikes** (image blur, structured scene), **Trees** (image blur, textured scene), **Leuven** (illumination change, structured scene), and **Ubc** (JPEG compression, structured scene).

test data). In order to study in more detail the tolerance of our descriptor to illumination changes, we captured four additional image pairs shown in Fig. 4.

The evaluation criterion is based on the number of correct and false matches between a pair of images. The definition of a match depends on the matching strategy. As in [7], we declare two interest regions to be matched if the Euclidean distance between their descriptors is below a threshold. The number of correct matches is determined with the *overlap error* [30]. It measures how well the regions A and B correspond under a known homography

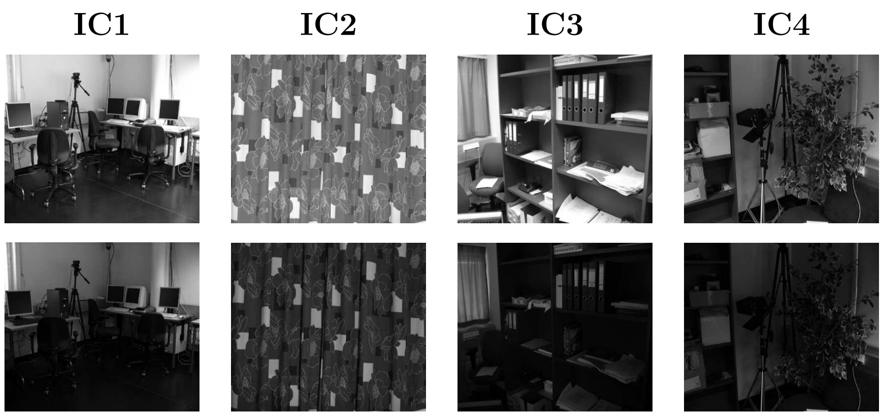


Fig. 4. Additional test images for illumination changes.

H, and is defined by the ratio of the intersection and union of the regions: $\epsilon_S = 1 - (A \cap H^T BH)/(A \cup H^T BH)$. A match is assumed to be correct if $\epsilon_S < 0.5$. A descriptor can have several matches and several of them may be correct. The results are presented with *recall* versus *1-precision*:

$$recall = \frac{\#correct\ matches}{\#correspondences}, \quad 1 - precision = \frac{\#false\ matches}{\#all\ matches}, \tag{3}$$

where the *#correspondences* stands for the ground truth number of matching regions between the images. The curves are obtained by varying the distance threshold and a perfect descriptor would give a recall equal to 1 for any precision.

Next, we first evaluate the performance of our CS-LBP descriptor for different parameter settings and then compare the resulting versions to the SIFT descriptor.

5.2.1 Parameter Evaluation.

The evaluation of different parameter settings was carried out for several pairs of images with similar results. Here, we show the results for a pair of images with a viewpoint change of more than 50 degrees (see Table 2). We use the HesAff detector which extracts 2973 and 3058 interest regions in the left and right images, respectively. The performance is measured with nearest neighbor matching, i.e., a descriptor has only one match. We keep the 500 best matches and report the percentage of correct matches. There are 668 possible nearest neighbor correspondences identified between the images.

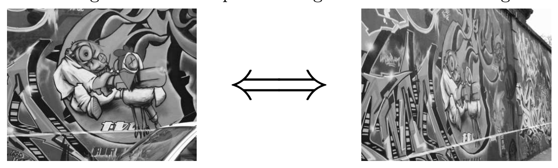
We compare the matching performance (percentage of correct matches) for differently spaced Cartesian location grids, three weighting schemes, and the three CS-LBP-related parameters. Because of a huge amount of different combinations, only one parameter was varied at a time while the others were kept fixed. The results are shown in Table 2. From the results we see that for all weighting schemes, 4×4 grid provides the best performance followed by 3×3 and 5×5 ones. 4×4 Cartesian grid is also the one used in the SIFT descriptor. Out of the three weighting schemes the best performance is reported for uniform weighting. Gaussian weighting gives almost exactly the same performance but is computationally more expensive. Gaussian-weighted gradient magnitude used by the SIFT gives clearly worst results. The best values for CS-LBP-related parameters, i.e. R, N, and T, seem to be 2, 8, and 0.01, respectively. In conclusion, 4×4 grid and the $CS-LBP_{2,8,0.01}$ with uniform weighting is a good choice. In the following experiments, if not explicitly mentioned, the CS-LBP descriptor uses this parameter setting. The resulting descriptor size is 256.

Although the results are not shown in Table 2, in addition to Cartesian location grid, we also experimented with log-polar grid. According to the results, the Cartesian grid gives better performance. For example, if we see the number of location bins, the closest log-polar grid for the 4×4 Cartesian one is the grid

Table 2

Parameter evaluation results. Only one parameter was varied at a time while the others were fixed at values M=4, W=W_U, R=2, N=8, and T=0.01. Abbreviations: M=Cartesian grid size, W=Weighting method, W_U=Uniform, W_{GG}=Gaussian-weighted gradient magnitude, W_G=Gaussian, (R, N, T)=CS-LBP operator parameters.

Test images with a viewpoint change of more than 50 degrees.



$W\|M$	1	2	3	4	5	6	7	8
W_U	0.0140	0.4060	0.5160	0.5220	0.5100	0.4900	0.4700	0.4520
W_{GG}	0.0060	0.3960	0.4740	0.4800	0.4560	0.4280	0.4040	0.3800
W_G	0.0140	0.4040	0.5140	0.5200	0.5100	0.4880	0.4700	0.4500

R	1	2	3	4	5	6	7	8
	0.5180	0.5220	0.4960	0.4620	0.3880	0.3440	0.2960	0.2500

N	2	4	6	8	10	12	14	16
	0.1500	0.4100	0.4600	0.5220	0.4840	0.5200	0.4820	0.4860

T	0	0.01	0.02	0.03	0.04	0.05	0.06	0.07
	0.5060	0.5220	0.4920	0.4560	0.4080	0.3760	0.3420	0.3240

with 3 bins in radial direction and 8 in angular direction, which results in 17 location bins. Note that the central bin is not divided in angular directions. The matching result for this grid is 0.5040, which is outperformed by the Cartesian counterpart (0.5220).

5.2.2 *Matching Results*

Figure 5 shows matching results for HesAff and HarAff regions. The ranking between the two descriptors seems to be more or less invariant to region detection method. Better overall performance is obtained with the HesAff regions. The results show that for most of the test cases the CS-LBP descriptor is able to outperform the SIFT descriptor. For the rest of the cases, comparable performance is achieved. In the case of illumination changes, i.e. for the *Leuven* and *IC1-IC4* sequences, we can see that CS-LBP is more robust. This could be explained by the fact that by its nature LBP is invariant with respect to monotonic gray scale transformations at pixel level while in SIFT normalization is done at region level. Figure 6 shows the corresponding matching results for HesLap and HarLap regions. As can be seen, the results are in conformance with the affine invariant ones.

In the previous experiments, the $CS-LBP_{2,8,0.01}$ operator yields descriptors of length 256, i.e. twice as long as the SIFT descriptor. We were interested to see how the performance changes if we modify the parameters of our descriptor so that the descriptor dimension is the same as for the SIFT descriptor. This is achieved by decreasing the neighborhood size of the CS-LBP operator from 8 to 6. Figure 7 shows the results for HesAff and HarAff regions. It can be seen that there are only slight changes in the results when compared to the ones with longer CS-LBP descriptor.

We also ran some tests for the original LBP operator. Here we show the results for two operators, namely $LBP_{2,4,0.01}$ and $LBP_{2,3,0.01}$. These operators yield descriptors of length 256 and 128, respectively. If we look at the descriptor size, the corresponding CS-LBP operators are $CS-LBP_{2,8,0.01}$ and $CS-LBP_{2,6,0.01}$. To make the comparison fair, also the LBP operators use the threshold parameter T. The results in Figure 8 show that for the equal-size descriptors, CS-LBP clearly outperforms the original LBP operator. It would not be reasonable to increase the neighborhood size of the LBP operator to that of the CS-LBP counterpart because of the huge increase in descriptor size. For example, for $LBP_{2,8,0.01}$ the descriptor size would be 4096.

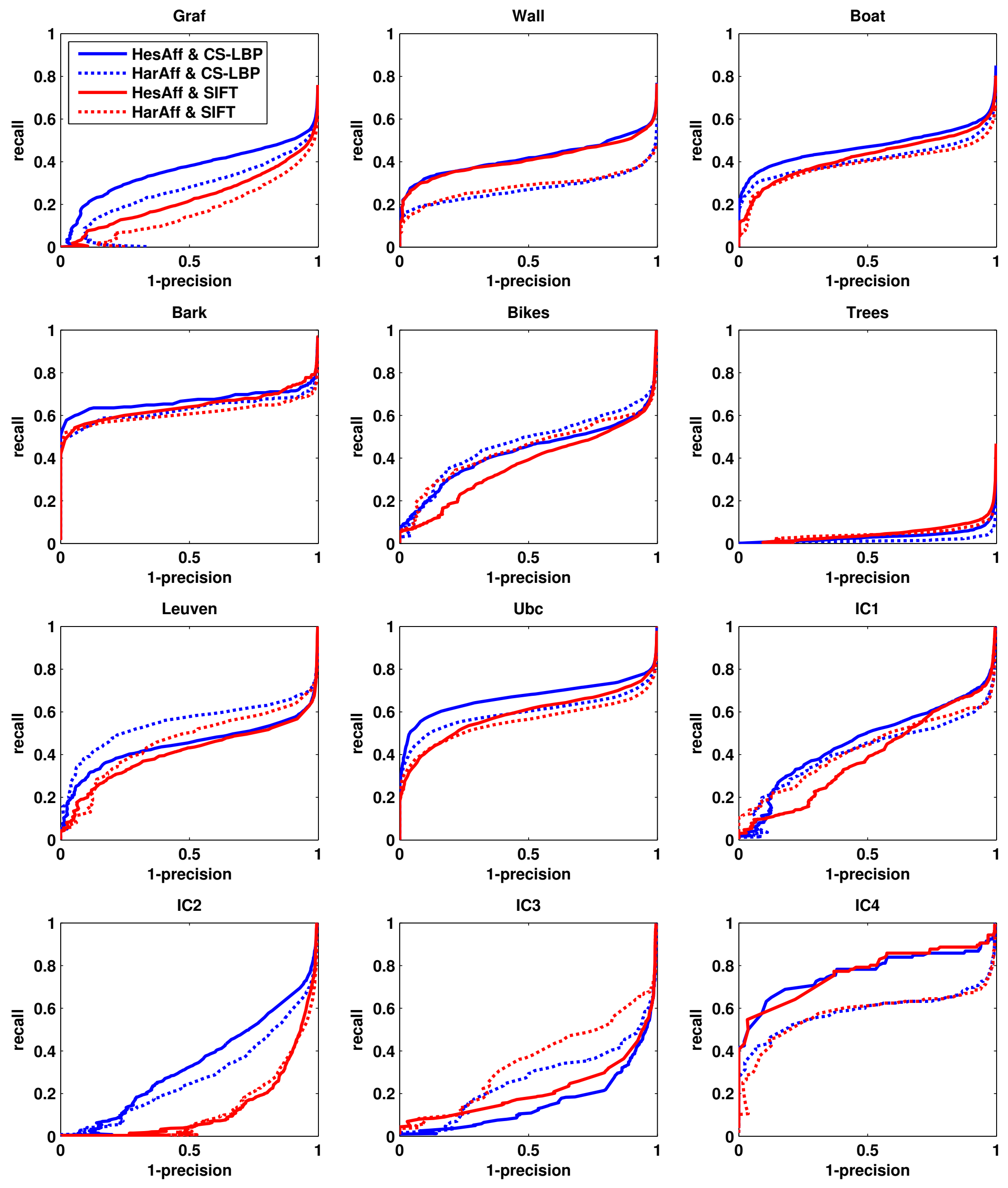


Fig. 5. Matching results for HesAff and HarAff regions. The CS-LBP descriptor uses 4×4 grid and the $CS-LBP_{2,8,0.01}$ with uniform weighting.

5.3 Object Category Classification

The recognition of object categories is one of the most challenging problems in computer vision. Recent achievements in object recognition have shown that using local features, or descriptors computed at a sparse set of scale or affine

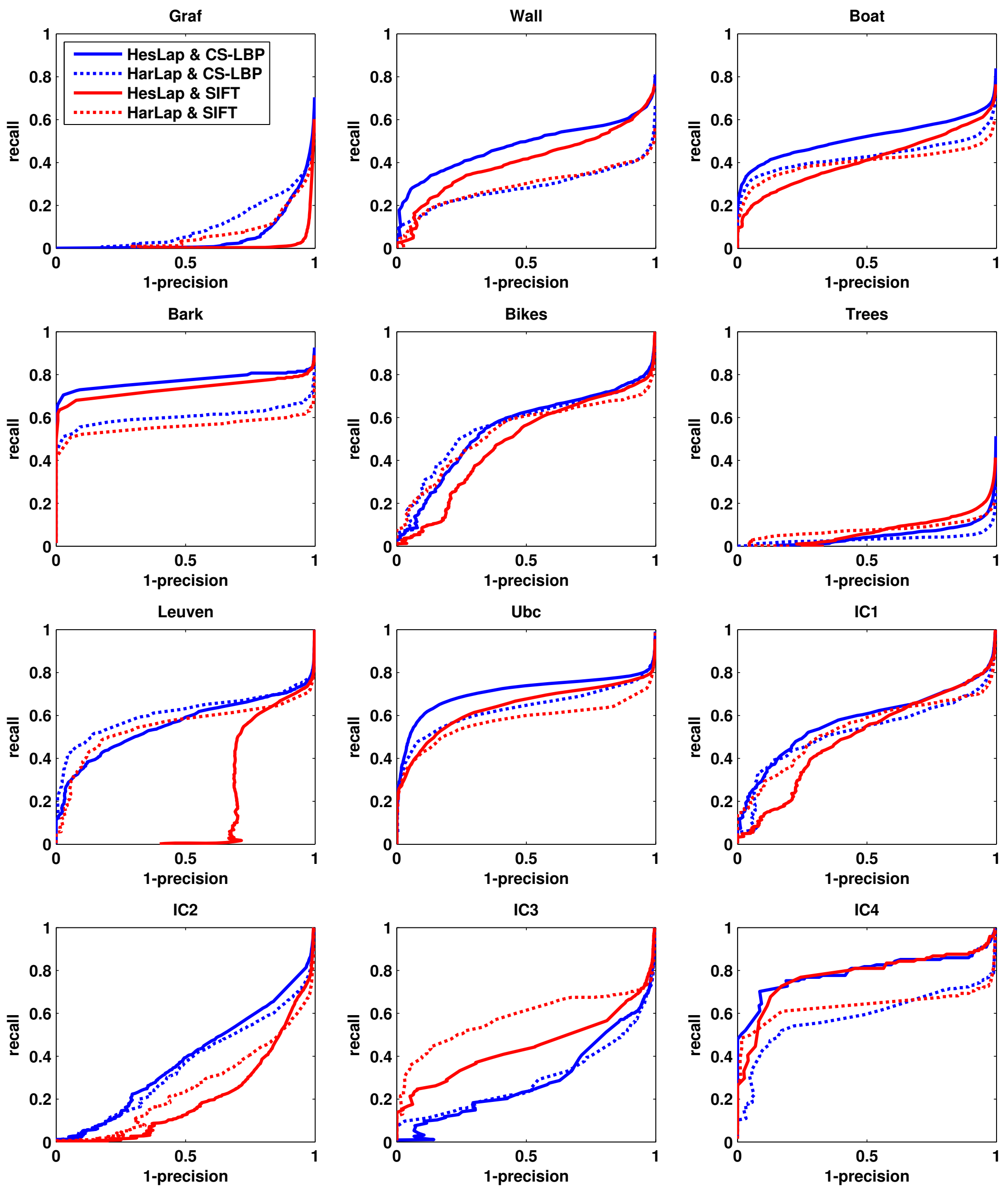


Fig. 6. Matching results for HesLap and HarLap regions. The CS-LBP descriptor uses 4×4 grid and the $CS-LBP_{2,8,0.01}$ with uniform weighting.

invariant keypoints, tends to be an effective approach. For comparing the two descriptors in the context of object category classification, we use the *PASCAL Visual Object Classes Challenge 2006* protocol [28]. The dataset includes ten categories: *bicycle*, *bus*, *car*, *cat*, *cow*, *dog*, *horse*, *motorbike*, *person*, and *sheep*. See Fig. 9 for an example image of each category. There are four sets of images

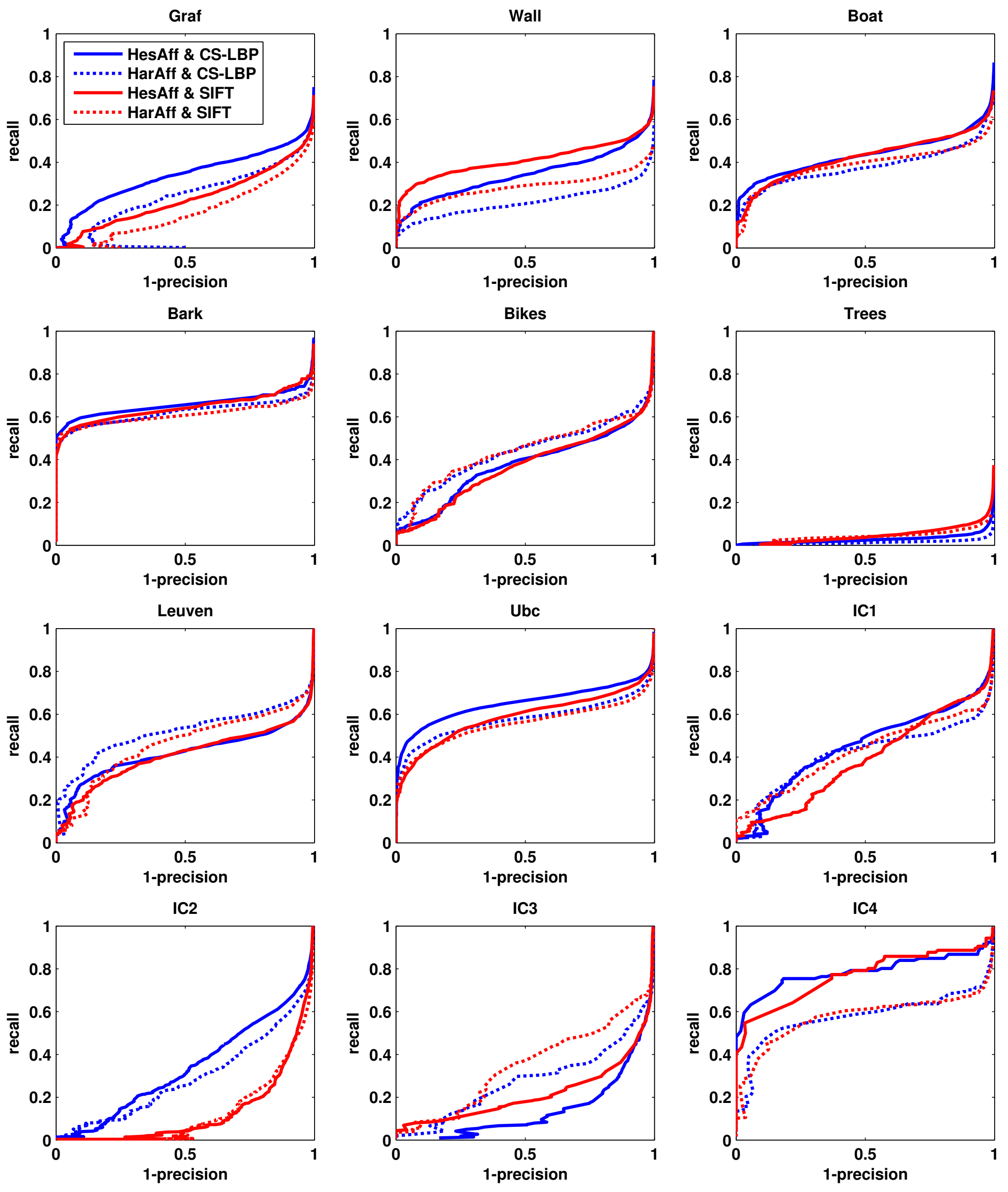


Fig. 7. Matching results for HesAff and HarAff regions. The CS-LBP descriptor uses 4×4 grid and the $CS - LBP_{2,6,0.01}$ with uniform weighting.

provided for the classification task: *train*, *val*, *trainval* (*train*+*val*), and *test*. We use *trainval* images for training and *test* images for testing our classifier. Table 3 summarizes the number of images for each class and image set. The classification task is judged by the *Receiver Operating Characteristic* (ROC) curve. The quantitative measure used is the *Area Under Curve* (AUC). Next,

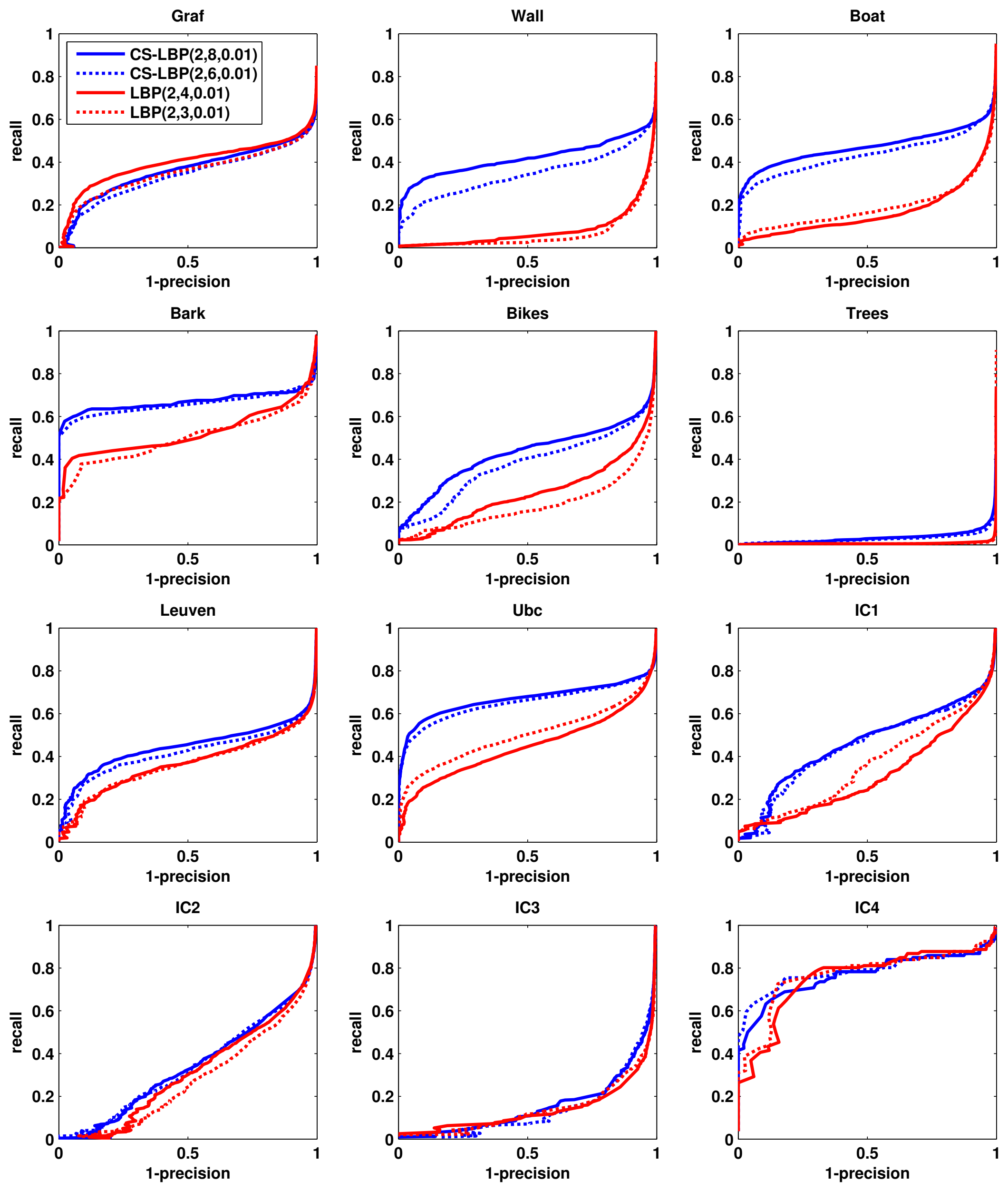


Fig. 8. Matching results for different CS-LBP and LBP operators on HesAff regions.

we present our classification framework which is the same as used in [31].

Classifier Training.

(1) For each training image, we detect Hessian-Laplace interest regions.

(2) For each detected region, we build a descriptor (SIFT or CS-LBP).

(3) For each of the ten classes we compute 200 textons by clustering the

descriptors of the class with k-means (k = 200). By concatenating the textons over the ten classes, we obtain a global texton vocabulary of size 2000 (10 × 200).

(4) We represent each training image as a histogram of texton labels. Given the global texton vocabulary, the ith entry of a histogram is the proportion of all descriptors in the image having label i.

(5) For classification, we use *Support Vector Machines* (SVM) [32]. We follow the PASCAL VOC 2006 setup and train a detector for each class. To compare image histograms $S_1 = (u_1, ..., u_m)$ and $S_2 = (w_1, ..., w_m)$ we use the χ^2-distance defined as

$$D(S_1, S_2) = \frac{1}{2} \sum_{i=1}^{m} \frac{(u_i - w_i)^2}{u_i + w_i}. \tag{4}$$

To incorporate χ^2-distance into the SVM framework, we use extended Gaussian kernel defined as

$$K(S_i, S_j) = exp(-\frac{1}{A} D(S_i, S_j)). \tag{5}$$

The resulting kernel is the χ^2-kernel. The parameter A of the kernel is the mean value of the χ^2-distances between all training images.

Classifier Testing.

(1) Identical to the training step (1).
(2) Identical to the training step (2).
(3) Identical to the training step (4).
(4) We classify the test images with the previously trained SVM.

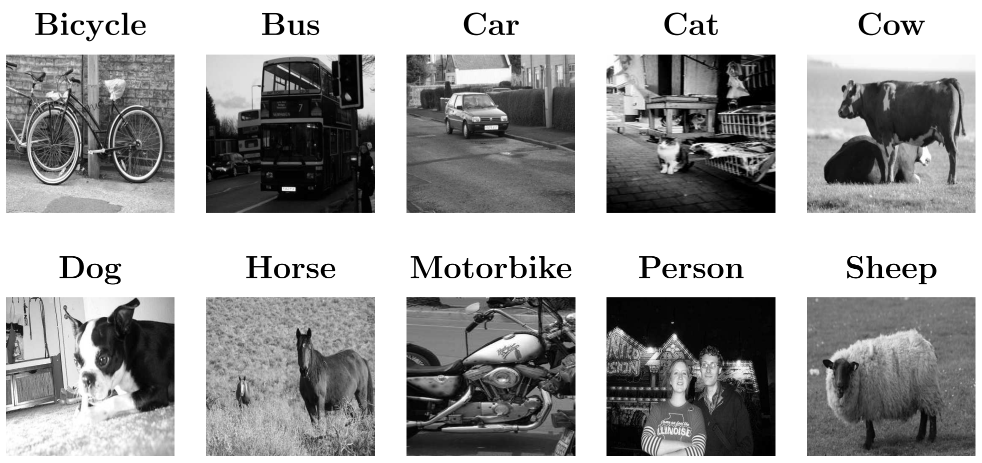


Fig. 9. An example image on each category in the *PASCAL Visual Object Classes Challenge 2006* dataset.

Table 3

Statistics of the *PASCAL Visual Object Classes Challenge 2006* image sets.

	Bicycle	**Bus**	**Car**	**Cat**	**Cow**
trainval	270	174	553	386	206
test	268	180	544	388	197

	Dog	**Horse**	**Motorbike**	**Person**	**Sheep**
trainval	365	247	235	666	251
test	370	254	234	675	238

5.3.1 Classification Results

The object category classification experiments were done for Hessian-Laplace regions with different CS-LBP descriptors. The results are given in Table 4. Interestingly, all the tested CS-LBP descriptors give more or less the same performance. Slightly the best overall performance is obtained for a 4×4 grid and the $CS-LBP_{1,8,0.01}$ with uniform weighting. The SIFT descriptor gives only slightly worse results. In order to see how the original LBP operator works when compared to CS-LBP operator, we ran the tests also for different LBP operators. The results are shown in Table 5. The results for this operator are in line with the ones with CS-LBP. This indicates that the CS-LBP loses its advantage over the LBP in the context of object category classification. In fact, as can be seen from the results, the original LBP slightly outperforms the CS-LBP. The ROC curves for the SIFT and the best performing CS-LBP and LBP descriptors are show in Figure 10.

Table 4

The object category classification results in AUC for the SIFT descriptor and for 8 different CS-LBP descriptors. Abbreviations: M=Cartesian grid size, W=Weighting method, W_U=Uniform, (R, N, T)=CS-LBP operator parameters.

	SIFT	CS-LBP: W=W_U, T=0.01							
		M=4				M=3			
Class		R,N=2,8	R,N=1,8	R,N=2,6	R,N=1,6	R,N=2,8	R,N=1,8	R,N=2,6	R,N=1,6
Bicycle	0.9191	0.9167	0.9171	0.9029	0.9007	0.9220	0.9143	0.9067	0.9077
Bus	0.9726	0.9731	0.9745	0.9738	0.9712	0.9727	0.9740	0.9699	0.9690
Car	0.9595	0.9666	0.9665	0.9682	0.9672	0.9645	0.9675	0.9644	0.9660
Cat	0.8824	0.8883	0.8838	0.8829	0.8921	0.8853	0.8822	0.8827	0.8845
Cow	0.8967	0.9155	0.9113	0.9077	0.9138	0.9059	0.9128	0.9113	0.9091
Dog	0.8192	0.8317	0.8303	0.8254	0.8350	0.8363	0.8384	0.8274	0.8299
Horse	0.8449	0.8869	0.8932	0.8879	0.8948	0.9036	0.8794	0.8911	0.8763
Motorbike	0.9391	0.9502	0.9523	0.9346	0.9419	0.9397	0.9515	0.9264	0.9409
Person	0.8068	0.8193	0.8295	0.8079	0.8172	0.8131	0.8200	0.8083	0.8118
Sheep	0.8959	0.9197	0.9241	0.9207	0.9176	0.9231	0.9197	0.9235	0.9199
Mean	0.8936	0.9068	0.9083	0.9012	0.9052	0.9066	0.9060	0.9012	0.9015

Table 5

The object category classification results in AUC for 8 different LBP descriptors. Abbreviations: M=Cartesian grid size, W=Weighting method, W_U=Uniform, (R, N, T)=LBP operator parameters.

	LBP: W=W_U, T=0.01							
	M=4				M=3			
Class	R,N=2,4	R,N=1,4	R,N=2,3	R,N=1,3	R,N=2,4	R,N=1,4	R,N=2,3	R,N=1,3
Bicycle	0.9268	0.9303	0.9075	0.9199	0.9187	0.9238	0.9138	0.9219
Bus	0.9741	0.9717	0.9711	0.9664	0.9724	0.9718	0.9656	0.9629
Car	0.9695	0.9701	0.9652	0.9656	0.9684	0.9716	0.9632	0.9663
Cat	0.8969	0.9014	0.8960	0.8909	0.8917	0.8982	0.8988	0.8962
Cow	0.9156	0.9215	0.9132	0.9156	0.9236	0.9239	0.9159	0.9167
Dog	0.8357	0.8351	0.8410	0.8339	0.8402	0.8449	0.8324	0.8397
Horse	0.8877	0.8877	0.8794	0.8848	0.8923	0.8941	0.8753	0.8766
Motorbike	0.9431	0.9524	0.9411	0.9459	0.9426	0.9477	0.9366	0.9333
Person	0.8248	0.8328	0.8022	0.8128	0.8217	0.8277	0.7914	0.8071
Sheep	0.9273	0.9294	0.9223	0.9219	0.9308	0.9316	0.9239	0.9241
Mean	0.9102	0.9132	0.9039	0.9058	0.9102	0.9135	0.9017	0.9045

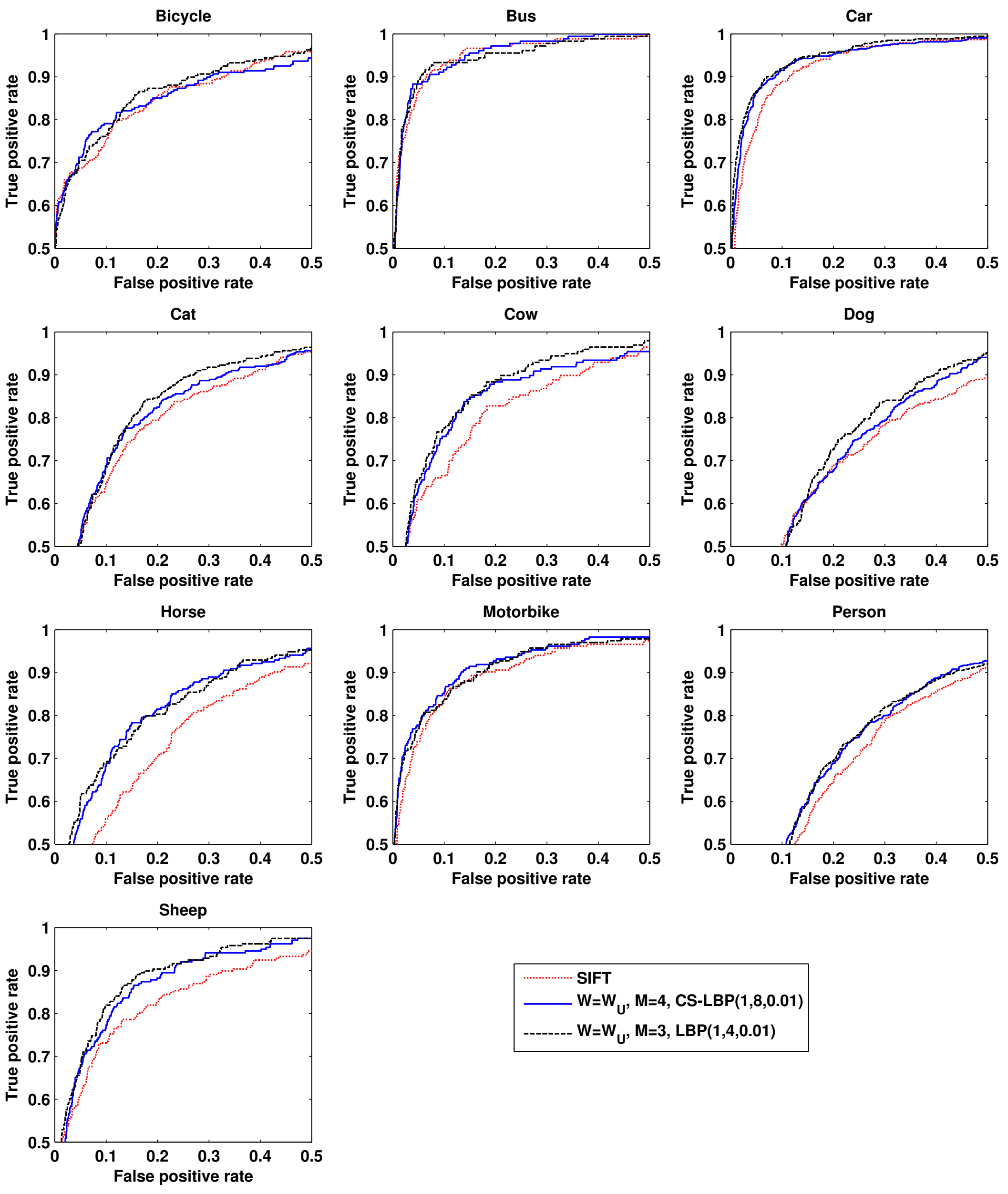


Fig. 10. The ROC curves for the SIFT and the best performing CS-LBP and LBP descriptors. See Tables 4 and 5 for the AUC (Area Under Curve) values.

5.4 Discussion

In comparison with the SIFT, for most of the test cases, our descriptor gives either better or comparable performance. If we look at the results of our descriptor for the CS-LBP and LBP operators, we clearly see that the CS-LBP provides better performance in the matching experiments while equal performance is obtained for object category classification. One possible explanation might be that the discriminative power of the descriptor, which is very important in image matching where exact correspondences are needed, is less important in the context of category classification where we are looking for similar regions without a need for exact matches. Since the CS-LBP performs well in all the tested contexts, it should be preferred over the LBP.

6 Conclusions

A new method for interest region description was presented. The proposed CS-LBP descriptor combines the strengths of two well-known methods, the SIFT descriptor and the LBP texture operator. It uses a SIFT-like grid and replaces SIFT's gradient features with a LBP-based feature, i.e. CS-LBP which was also introduced in this paper. The CS-LBP feature has many properties that make it well suited for this task, namely a relatively short feature histogram, tolerance to illumination changes, and computational simplicity. Furthermore, it does not require many parameters to be set. The performance of the CS-LBP descriptor was compared to that of the SIFT descriptor in the contexts of matching and object category classification. For many of the test cases, our descriptor outperforms the SIFT descriptor. Comparable performance is obtained for other test cases. The CS-LBP descriptor proved to be tolerant of illumination changes. This could be explained by the fact that by its nature LBP is invariant with respect to monotonic gray scale transformations at pixel level. In SIFT normalization is done at region level. Also, the CS-LBP descriptor is computationally simpler than the SIFT.

Acknowledgment

The financial support provided by the Academy of Finland and the Infotech Oulu Graduate School is gratefully acknowledged.

References

[1] K. Mikolajczyk, C. Schmid, Indexing based on scale invariant interest points, in: 8th IEEE International Conference on Computer Vision, Vol. 1, 2001, pp. 525–531.

[2] T. Tuytelaars, L. V. Gool, Matching widely separated views based on affine invariant regions, International Journal of Computer Vision 59 (1) (2004) 61–85.

[3] D. G. Lowe, Distinctive image features from scale-invariant keypoints, International Journal of Computer Vision 60 (2) (2004) 91–110.

[4] S. Lazebnik, C. Schmid, J. Ponce, A sparse texture representation using local affine regions, IEEE Transactions on Pattern Analysis and Machine Intelligence 27 (8) (2005) 1265–1278.

[5] S. Se, D. Lowe, J. Little, Global localization using distinctive visual features, in: IEEE/RSJ International Conference on Intelligent Robots and System, Vol. 1, 2002, pp. 226–231.

[6] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, L. V. Gool, A comparison of affine region detectors, International Journal of Computer Vision 65 (1/2) (2005) 43–72.

[7] K. Mikolajczyk, C. Schmid, A performance evaluation of local descriptors, IEEE

Transactions on Pattern Analysis and Machine Intelligence 27 (10) (2005) 1615–1630.

[8] S. Belongie, J. Malik, J. Puzicha, Shape matching and object recognition using shape contexts, IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (4) (2002) 509–522.

[9] H. Bay, T. Tuytelaars, L. V. Gool, SURF: Speeded up robust features, in: European Conference on Computer Vision, Vol. 1, 2006, pp. 404–417.

[10] H. Ling, D. W. Jacobs, Deformation invariant image matching, in: 10th IEEE International Conference on Computer Vision, Vol. 2, 2005, pp. 1466–1473.

[11] Y. Ke, R. Sukthankar, PCA-SIFT: A more distinctive representation for local image descriptors, in: IEEE Conference on Computer Vision and Pattern Recognition, Vol. 2, 2004, pp. 506–513.

[12] W. Freeman, E. Adelson, The design and use of steerable filters, IEEE Transactions on Pattern Analysis and Machine Intelligence 13 (9) (1991) 891–906.

[13] L. J. V. Gool, T. Moons, D. Ungureanu, Affine/photometric invariants for planar intensity patterns, in: 4th European Conference on Computer Vision, 1996, pp. 642–651.

[14] F. Schaffalitzky, A. Zisserman, Multi-view matching for unordered image sets, in: 7th European Conference on Computer Vision, 2002, pp. 414–431.

[15] G. Carneiro, A. D. Jepson, Multi-scale phase-based local features, in: IEEE Conference on Computer Vision and Pattern Recognition, Vol. 1, 2003, pp. 736–743.

[16] P. Moreels, P. Perona, Evaluation of features detectors and descriptors based on 3D objects, in: 10th IEEE International Conference on Computer Vision, Vol. 1, 2005, pp. 800–807.

[17] S. Winder, M. Brown, Learning local image descriptors, in: IEEE Conference on Computer Vision and Pattern Recognition, 2007.

[18] T. Randen, J. H. Husoy, Filtering for texture classification: A comparative study, IEEE Transactions on Pattern Analysis and Machine Intelligence 21 (1999) 291–310.

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

[20] T. Ojala, M. Pietikäinen, T. Mäenpää, Multiresolution gray-scale and rotation invariant texture classification with local binary patterns, IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (2002) 971–987.

[21] Http://www.ee.oulu.fi/mvg/page/lbp_bibliography.

[22] T. Ahonen, A. Hadid, M. Pietikäinen, Face description with local binary patterns: Application to face recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence 28 (12) (2006) 2037–2041.

[23] M. Heikkilä, M. Pietikäinen, A texture-based method for modeling the background and detecting moving objects, IEEE Transactions on Pattern Analysis and Machine Intelligence 28 (4) (2006) 657–662.

[24] M. Pietikäinen, T. Nurmela, T. Mäenpää, M. Turtinen, View-based recognition of real-world textures, Pattern Recognition 37 (2) (2004) 313–323.

[25] M. Heikkilä, M. Pietikäinen, C. Schmid, Description of interest regions with center-symmetric local binary patterns, in: 5th Indian Conference on Computer Vision, Graphics and Image Processing, Vol. 4338, 2006, pp. 58–69.

[26] T. Mäenpää, T. Ojala, M. Pietikäinen, M. Soriano, Robust texture classification by subsets of local binary patterns, in: 15th International Conference on Pattern Recognition, Vol. 3, 2000, pp. 935–938.

[27] Http://www.robots.ox.ac.uk/˜vgg/research/affine/.

[28] Http://www.pascal-network.org/challenges/VOC/voc2006.

[29] K. Mikolajczyk, C. Schmid, Scale & affine invariant interest point detectors, International Journal of Computer Vision 60 (1) (2004) 63–86.

[30] K. Mikolajczyk, C. Schmid, An affine invariant interest point detector, in: European Conference on Computer Vision, Vol. 1, 2002, pp. 128–142.

[31] J. Zhang, M. Marszałek, S. Lazebnik, C. Schmid, Local features and kernels for classification of texture and object categories: a comprehensive study, International Journal of Computer Vision 73 (2) (2007) 213–238.

[32] B. Scholkopf, A. J. Smola, Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond, MIT Press, Cambridge, MA, USA, 2001.