A Robust Descriptor based on Weber's Law

Jie Chen[1,2,3] Shiguang Shan[1] Guoying Zhao[2] Xilin Chen[1] Wen Gao[1,3] Matti Pietikäinen[2]

[1]Key Laboratory of Intelligent Information Processing, Chinese Academy of Sciences (CAS),
Institute of Computing Technology, CAS, Beijing, 100080, China
[2] Machine Vision Group, Department of Electrical and Information Engineering, P. O. Box 4500 FI-90014
University of Oulu, Finland
[3]School of Computer Science and Technology, Harbin Institute of Technology, Harbin, 150001, China
{jiechen, gyzhao, mkp}@ee.oulu.fi, { sgshan, xlchen, wgao}@jdl.ac.cn

Abstract

Inspired by Weber's Law, this paper proposes a simple, yet very powerful and robust local descriptor, Weber Local Descriptor (WLD). It is based on the fact that human perception of a pattern depends on not only the change of a stimulus (such as sound, lighting, et al.) but also the original intensity of the stimulus. Specifically, WLD consists of two components: its differential excitation and orientation. A differential excitation is a function of the ratio between two terms: One is the relative intensity differences of its neighbors against a current pixel; the other is the intensity of the current pixel. An orientation is the gradient orientation of the current pixel. For a given image, we use the differential excitation and the orientation components to construct a concatenated WLD histogram feature. Experimental results on Brodatz textures show that WLD impressively outperforms the other classical descriptors (e.g., Gabor). Especially, experimental results on face detection show a promising performance. Although we train only one classifier based on WLD features, the classifier obtains a comparable performance to state-of-the-art methods on MIT+CMU frontal face test set, AR face dataset and CMU profile test set.

1. Introduction

In this paper, we propose a simple, yet very powerful and robust local descriptor. It is inspired by Weber's Law, which is a psychological law [7]. It states that the change of a stimulus (such as sound, lighting, et al.) that will be just noticeable is a constant ratio of the original stimulus. When the change is smaller than this constant, human being would recognize it as a background noise rather than a valid signal. Motivated by this point, the proposed Weber Local Descriptor (WLD) is computed based on the ratio between the two terms: One is the relative intensity differences of its neighbors against a current pixel; the other is the intensity of the current pixel.

Several descriptors have been proposed to represent textured regions in practical applications, such as texture classification [15], object recognition [11], and face detection [21] et al. Recently, Mikolajczyk and Schmid evaluate the performance of some descriptors computed for local interest regions in [14]. Several researchers have used Weber's Law in computer vision, such as [2], et al.

The rest of this paper is organized as follows: In Section 2, we propose a local descriptor WLD. In Section 3 and 4, some experimental results are presented about the applications of WLD on texture classification and face detection, followed by conclusion in Section 5.

2. WLD for Image Representation

In this section, we review Weber's Law and then propose a descriptor WLD.

2.1. Weber's Law

Ernst Weber, an experimental psychologist in 19th century, observed that the ratio of the increment threshold to the background intensity is a constant [7]. This relationship, known since as Weber's Law, can be expressed as:

$$\frac{\Delta I}{I} = k \, , \tag{1}$$

where ΔI represents the increment threshold (just noticeable difference for discrimination); I represents the initial stimulus intensity and k signifies that the proportion on the left side of the equation remains constant despite of variations in the I term. The fraction $\Delta I/I$ is known as the Weber fraction.

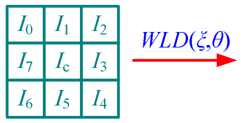
2.2 WLD

Motivated by Weber's Law, we propose a descriptor WLD. It consists of two components: its differential excitation (ξ) and orientation (θ). ξ is a function of the Weber fraction (i.e., the relative intensity differences of its neighbors against a current pixel and the current pixel itself). θ is a gradient orientation of the current pixel.

2.2.1 Differential excitation

We use the intensity differences between its neighbors and a current pixel as the changes of the current pixel. By this means, we hope to find the salient variations within an image to simulate human beings perception of patterns. Specifically, a differential excitation $\xi(I_c)$ of a current pixel is computed as illustrated in Fig. 1, where I_c denotes the intensity of the current pixel; I_i (i=0, 1, …p-1) denote the intensities of p neighbors of I_c (p=8 here).

978-1-4244-2243-2/08/$25.00 ©2008 IEEE



$$\begin{cases} \xi(I_c) = \arctan\left[\sum_{i=0}^{p-1}\left(\frac{I_i - I_c}{I_c}\right)\right] \\ \theta(I_c) = median(\theta_i) \quad (i=0,1,\ldots,p\text{-}1) \end{cases}$$

$$\theta_i = \arctan\left(\frac{I_{R(i+4)} - I_i}{I_{R(i+6)} - I_{R(i+2)}}\right)$$

$$R(x) = \mathrm{mod}(x, p)$$

Fig. 1. An illustration of computing a WLD feature of a pixel.

To compute $\xi(I_c)$, we first calculate the differences between its neighbors and a center point:

$$f_{diff}(I_i) = \Delta I_i = I_i - I_c. \tag{2}$$

Hinted by Weber's Law, we then compute the ratios of the differences to the intensity of the current point:

$$f_{ratio}(\Delta I_i) = \frac{\Delta I_i}{I_c}. \tag{3}$$

Subsequently, we consider the neighbor effects on the current point using a sum of the difference ratios:

$$f_{sum}\left(\frac{\Delta I_i}{I_c}\right) = \sum_{i=0}^{p-1}\left(\frac{\Delta I_i}{I_c}\right). \tag{4}$$

To improve the robustness of a WLD to noise, we use an arctangent function as a filter on $f_{sum}(\cdot)$. That is:

$$f_{\arctan}\left[f_{sum}(\cdot)\right]=\arctan\left[f_{sum}(\cdot)\right]. \tag{5}$$

Combining Eqs. (2), (3), (4) and (5), we have:

$$f_{\arctan}\left[\sum_{i=0}^{p-1}\left(\frac{\Delta I_i}{I_c}\right)\right]=\arctan\left[\sum_{i=0}^{p-1}\left(\frac{I_i - I_c}{I_c}\right)\right]. \tag{6}$$

So, $\xi(I_c)$ is computed as:

$$\xi(I_c) = \arctan\left[\sum_{i=0}^{p-1}\left(\frac{I_i - I_c}{I_c}\right)\right]. \tag{7}$$

Note that $\xi(I_c)$ may take a minus value if the intensities of neighbors are smaller than that of a current pixel. By this means, we attempt to preserve more discriminating information in comparison to using the absolute value of $\xi(I_c)$. Intuitively, if $\xi(I_c)$ is positive, it simulates the case that the surroundings is lighter than the current pixel. In contrast, if $\xi(I_c)$ is negative, it simulates the case that the surroundings is darker than the current pixel.

As shown in Fig. 2, we plot an average histogram of the differential excitations on 2,000 texture images. One can find that there are more frequencies at the two sides of the average histogram (e.g., [-π/2, -π/3] and [π/3, π/2]). It results from the approach of computing the differential excitation ξ of a pixel (i.e., a *sum* of the difference ratios of p neighbors against a central pixel) as shown in Eq. (7). However, it is valuable for a classification task. For more details, please refer to Section 2.2.4, Section 3 and 4.

2.2.2. Orientation

For the orientation component of WLD, it is computed as:

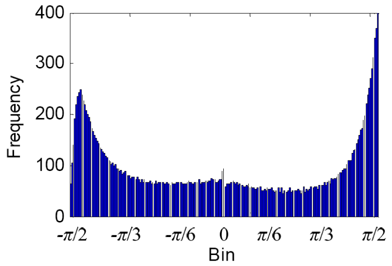


Fig. 2. A plot of an average histogram of the differential excitations on 2,000 texture images.

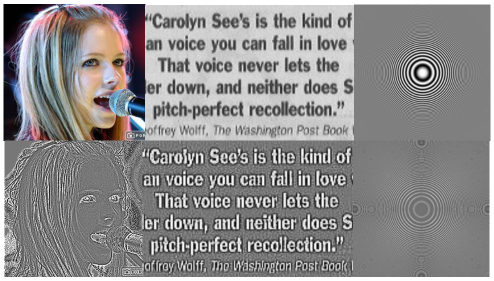

"Carolyn See's is the kind of
an voice you can fall in love
That voice never lets the
er down, and neither does S
pitch-perfect recollection."
offrey Wolff, The Washington Post Book

Fig. 3. The upper row is original images and the bottom is filtered images.

$$\theta(I_c) = median(\theta_i),\ (i=0,1,\ldots p/2\text{-}1), \tag{8}$$

where θ_i is the angle of a gradient difference:

$$\theta_i = \arctan\left(\frac{I_{R(i+4)} - I_i}{I_{R(i+6)} - I_{R(i+2)}}\right); \tag{9}$$

where I_i (i=0,1,…p/2-1) are the neighbors of a current pixel; $R(x)$ is to perform the modulus operation, i.e.,

$$R(x) = \mathrm{mod}(x, p), \tag{10}$$

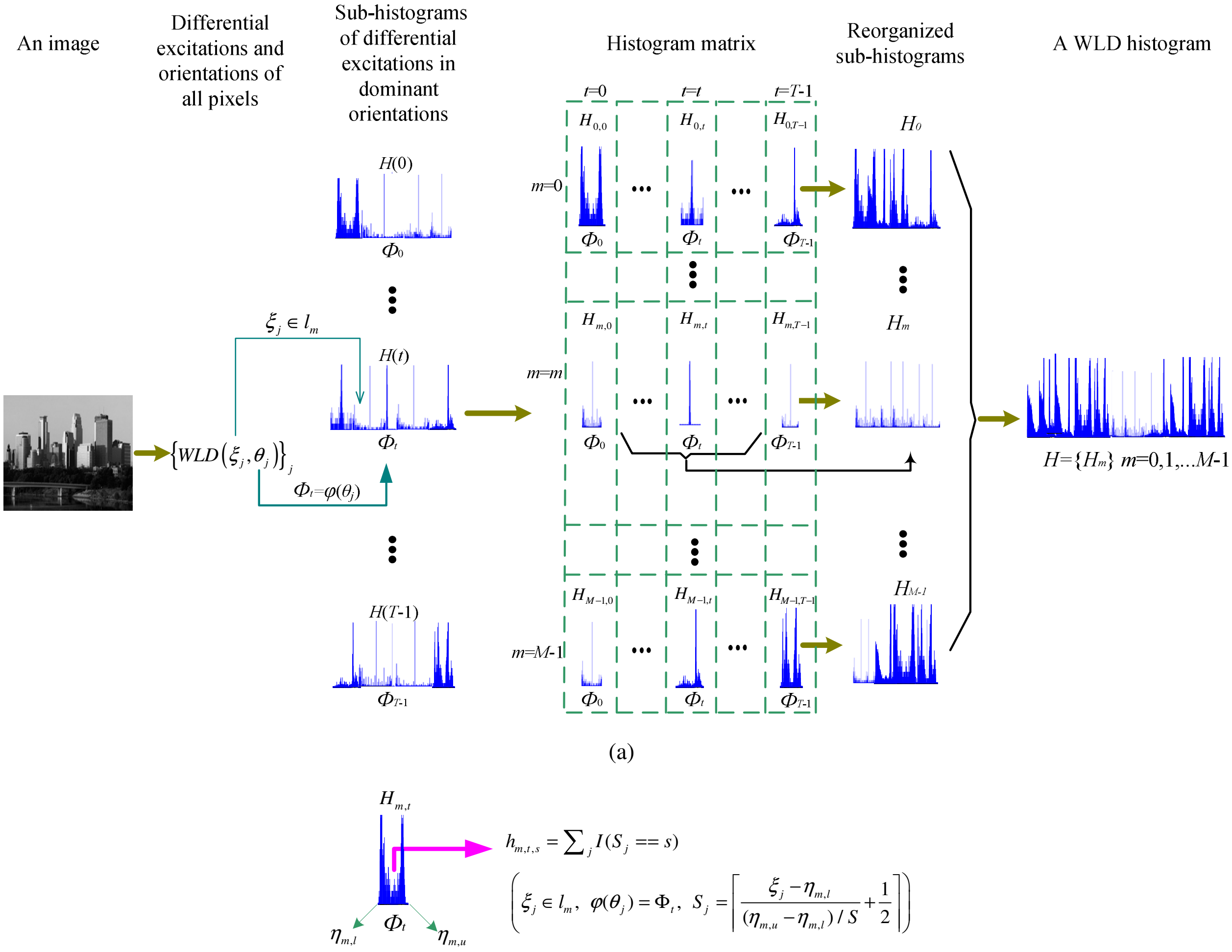
where p is the number of neighbors as mentioned in Section 2.2.1. Note that in Eqs. (8) and (9), we are only needed to compute half of these angles because there exists symmetry for θ_is when i takes its values in the two intervals [0, p/2-1] and [p/2, p-1].

For simplicity, θ's value is quantized into T dominant orientations. Before the quantization, the value of θ is mapped into the interval [0, 2π] according to its value computed using Eq. (9) and the sign of the denominator and numerator of the right side of Eq. (9). Thus, the quantization function is as follows:

$$\Phi_t=\varphi(\theta)=\frac{2t}{T}\pi, \text{ and } t=\mathrm{mod}\left(\left\lfloor\frac{\theta}{2\pi/T}+\frac{1}{2}\right\rfloor,T\right). \tag{11}$$

For example, if T=8, these T dominant orientations are computed as: Φ_t=($t\pi$)/4, (t=0, 1, …, T-1). In other words, those orientations located within the interval [Φ_t-($t\pi$)/8, Φ_t +($t\pi$)/8] are quantized as Φ_t.

As ilustrasted in Fig. 3, we show some filtered images by the descriptor WLD, from which one could conclude that a WLD extracts the edges of images perfectly even with heavy noise (e.g., the middle column of Fig. 3).


An image
Differential excitations and orientations of all pixels
Sub-histograms of differential excitations in dominant orientations
Histogram matrix
Reorganized sub-histograms
A WLD histogram
$\{WLD(\xi_j, \theta_j)\}_j$
$\xi_j \in l_m$
$\Phi_t = \varphi(\theta_j)$
$H(0)$
$H(t)$
$H(T\text{-}1)$
Φ_0
Φ_t
Φ_{T-1}
$t=0$
$t=t$
$t=T\text{-}1$
$m=0$
$m=m$
$m=M\text{-}1$
$H_{0,0}$
$H_{0,t}$
$H_{0,T-1}$
$H_{m,0}$
$H_{m,t}$
$H_{m,T-1}$
$H_{M-1,0}$
$H_{M-1,t}$
$H_{M-1,T-1}$
H_0
H_m
H_{M-1}
$H=\{H_m\}\ m=0,1,\dots M\text{-}1$
$H_{m,t}$
$\eta_{m,l}$
$\eta_{m,u}$
$h_{m,t,s} = \sum_j I(S_j == s)$
$\left(\xi_j \in l_m,\ \varphi(\theta_j) = \Phi_t,\ S_j = \left\lceil \frac{\xi_j - \eta_{m,l}}{(\eta_{m,u} - \eta_{m,l})/S} + \frac{1}{2} \right\rceil \right)$

(a)

(b)

Fig. 4. An illustration of a WLD histogram feature for a given image, (a) H is concatenated by M sub-histograms $\{H_m\}(m=0, 1, \ldots, M\text{-}1)$. Each H_m is concatenated by T histogram segments $H_{m,t}$ (t=0, 1, …, T-1). Meanwhile, for each column of the histogram matrix, all of M segments $H_{m,t}$ (m=0, 1, …, M-1) have the same dominant orientation Φ_t. In contrast, for each row of the histogram matrix, the differential excitations ξ_j of each segment $H_{m,t}$ (t=0, 1, …, T-1) belongs to the same interval l_m. (b) A histogram segment $H_{m,t}$. Note that if t is fixed, for any m or s, the dominant orientation of a bin $h_{m,t,s}$ is fixed (i.e., Φ_t).

Table 1 Weights for a WLD histogram

	H_0	H_1	H_2	H_3	H_4	H_5
Frequency percent	0.2519	0.1168	0.1175	0.0954	0.0864	0.3268
Weights (ω_m)	0.2688	0.0854	0.0958	0.1000	0.1021	0.3497

2.2.3. WLD histogram

Given an image, as shown in Fig. 4 (a), we encode the WLD features into a histogram H. We first compute the WLD features for each pixel (i.e., $\{WLD(\xi_j, \theta_j)\}_j$). The differential excitations ξ_j are then grouped as T sub-histograms $H(t)$ (t=0, 1, …, T-1), each sub-histogram $H(t)$ corresponding to a dominant orientation (i.e., Φ_t). Subsequently, each sub-histogram $H(t)$ is evenly divided into M segments, i.e., $H_{m,t}$, (m=0, 1, ..., M-1, and in our implementation we let M=6.). These segments $H_{m,t}$ are then reorganized as the histogram H. Specifically, H is concatenated by M sub-histograms, i.e., $H=\{H_m\}$, m=0, 1, ..., M-1. For each sub-histogram H_m, it is concatenated by T segments $H_m=\{H_{m,t}\}$, t=0,1,…T-1.

Note that after each sub-histogram $H(t)$ is evenly divided into M segments, the range of differential excitations ξ_j (i.e., l=[-π/2, π/2]) is also evenly divided into M intervals l_m (m=0, 1, ..., M-1). Thus, for each interval l_m, we have $l_m=[\eta_{m,l}, \eta_{m,u}]$, here, the lower bound $\eta_{m,l}$ = $(m/M\text{-}1/2)\pi$ and the upper bound $\eta_{m,u}$ =$[(m+1)/M\text{-}1/2]\pi$. For examples, l_0=[-π/2, -π/3].

Furthermore, As shown in Fig. 4 (b), a segment $H_{m,t}$ is composed of S bins, i.e., $H_{m,t}=\{h_{m,t,s}\}$, $s=0, 1, \ldots, S-1$. Herein, $h_{m,t,s}$ is computed as:

$$h_{m,t,s} = \sum\nolimits_j I(S_j == s), \quad \left(\xi_j \in l_m,\ \varphi(\theta_j) = \Phi_t,\ S_j = \left\lceil \frac{\xi_j - \eta_{m,l}}{(\eta_{m,u} - \eta_{m,l})/S} + \frac{1}{2} \right\rceil \right), \tag{12}$$

where $I(\cdot)$ is a function as follows:

$$I(X) = \begin{cases} 1 & X \text{ is ture} \\ 0 & \text{otherwise} \end{cases}. \tag{13}$$

Thus, $h_{m,t,s}$ means the number of the pixels whose differential excitations ξ_j belong to the same interval l_m and orientations θ_j are quantized to the same dominant orientation Φ_t and that the computed index S_j is equal to s.

We segment the range of ξ into several intervals due to the fact that different intervals correspond to the different variances in a given image. For example, given two pixels P_i and P_j, if their differential excitations $\xi_i \in l_0$ and $\xi_j \in l_2$, we say that the intensity variance around P_i is larger than that of P_j. That is, flat regions of an image produce smaller values of ξ while non-flat regions produce larger values. However, besides the flat regions of an image, there are two kinds of intensity variations around a central pixel which might lead to smaller differential excitations. One is the clutter noise around a central point; the other is the "uniform" patterns as shown in [15] (The term "uniform" means that there are a limited number of transitions or discontinuities in the circular presentation of the pattern.). Meanwhile, the latter provides a majority of variations in comparison to the former, and the latter can be discriminated by the orientations of the current pixels.

Here, we let M=6 for the reason that we attempt to use these intervals to approximately simulate the variances of high, middle or low frequency in a given image. That is, for a pixel P_i, if its differential excitation $\xi_i \in l_0$ or l_5, we call that the variance near P_i is of high frequency; if $\xi_i \in l_1$ or l_4, or $\xi_i \in l_2$ or l_3, we call that the variance near P_i is of middle frequency or low frequency, respectively.

2.2.4. Weight for a WLD histogram

Intuitively, one often pays more attention to the variances in a given image compared to the flat regions. That is, the different frequency segments H_m play different roles for a classification task. Thus, we can weight the different frequency segments with different weights for a better classification performance.

For weight selection, a heuristic approach is to take into account the different contributions of the different frequency segments H_m (m=0, 1, …, M-1). First, by computing the recognition rate for each sub-histogram H_m separately, we obtain M rates $R=\{r_m\}$; then, we let each weight $\omega_m = r_m / \sum_i r_i$ as shown in table 1. Simultaneously, as shown in table 1, we collect statistics of the percent of frequencies of each sub-histogram. From this table, one can find that these two groups of values (i.e., frequency percent and weights) are very similar.

3. Application to Texture classification

In this section, we use WLD features for texture classification and compare the results with that of the state-of-the-art methods.

3.1. Background

Several approaches for the extraction of texture features have been proposed in literature. Dorkó and Schmid optimize the keypoint detection and then use Scale Invariant Feature Transform (SIFT) for the image representation [3]. Jalba et al. present a multi-scale method based on mathematical morphology [8]. Lazebnik et al. present a probabilistic part-based approach to describe the texture and object [9]. Manjunath and Ma use Gabor filters for texture analysis [12]. Ojala et al. propose to use signed gray-level differences and their multidimensional distributions for texture description [16]. Urbach et al. describe a multiscale and multishape morphological method for pattern-based analysis and classification of gray-scale images using connected operators [20]. A recent comprehensive study about local features and kernels for texture classification please refer to [24].

3.2. Dataset

Brodatz dataset [1] is a well-known benchmark dataset. It contains 111 different texture classes where each class is represented by one image. Some examples are shown in Fig. 5. Using the similar experimental set-ups as [8, 16, 20], images of 640×640 pixels are divided into 16 disjoint squares of size 160×160. For each of these smaller images, three additional versions are created by one of the following transformations: 1) 90 degrees rotation, 2) scaling the 120×120 subimage in the center to 160×160, or 3) a combination of 1) or 2).

Note that for Brodatz dataset, experiments are carried out for ten-fold cross validation to avoid bias. For each round, we randomly divide the samples in each class into two subsets of the same size, one for training and the other for testing. The results are reported as the average value and standard deviation over the ten runs.

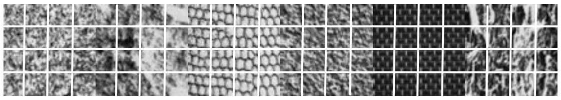


Fig. 5 Some examples from Brodatz dataset (http://www.ux.uis.no/~tranden/brodatz.html).

3.3. WLD Feature for Classification

For texture representation, given an image, we extract WLD features as shown in Fig. 4. Here, we experientially let M=6, T=8, S=20. In addition, we also weight each sub-histogram H_m using the same weights as shown in table 1.

For the classifier, we use K-nearest neighbor. In our case, K=3. To compute the distance between two given images I_1 and I_2, we first obtain their WLD feature histograms H^1 and H^2. We then measure the similarity between H^1 and H^2. In our experiments, we use the normalized histogram intersection $\Pi(H^1, H^2)$ as a similarity measurement of two histograms:

$$\Pi(H^1, H^2) = \sum_{i=1}^{L} \min(H^{1,i}, H^{2,i}) / \sum_{i=1}^{L} H^{1,i} , \qquad (14)$$

where L is the number of bins in a histogram.

3.4. Experimental Results

Experimental results on Brodatz textures are illustrated in Fig. 6. In this figure, we also compare our method with others on the classification task of Brodatz textures: Dorkó [3], Jalba [8], Lazebnik [9], Manjunath [12], Ojala [16] and Urbach [20]. Note that all the results by other methods in Fig. 6 are quoted directly from the original papers except Manjunath [12]. The approach in [12] is a "traditional" texture analysis method using global mean and standard deviation of the responses of Gabor filters. However, the results of Manjunath [12] are a little out-of-date. We use the results in [24] for a substitution. From Fig. 6, one can find that our approach works in a very robust way in comparison to other methods.

4. Application to face detection

In this section, we use WLD features for face detection. Although we train only one classifier, we use it to detect frontal, occluded and profile faces. Furthermore, experimental results show that this classifier obtains comparable performance to state-of-the-art methods.

4.1. Background

The goal of face detection is to determine whether there are any faces in a given image, and return the location and extent of each face in the image if one or more faces are present. Recently, many methods for detecting faces have been proposed [23]. Among these methods, learning based approaches to capture the variations in facial appearances have attracted much attention, such as [18, 19]. One of the most important progresses is the appearance of boosting-based method, such as [6, 10, 17, 21, 22]. In addition, Hadid et al. use Local Binary Pattern (LBP) not only for face detection but also for recognition [5]. Garcia and Delakis use a convolutional face finder for fast and robust face detection [4].

4.2. WLD Feature for Face Samples

We use WLD features as a facial representation and build a face detection system. For the facial representation, as illustrated in Fig. 7, we divide an input sample into

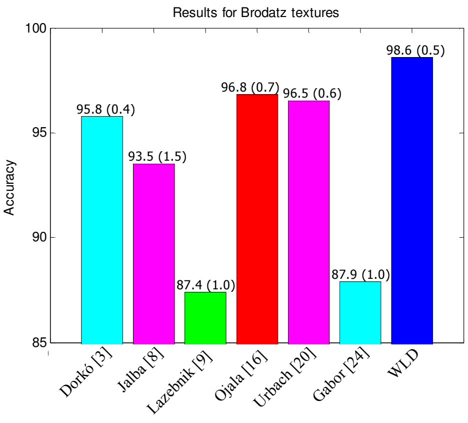


Fig. 6. Results comparison with state-of-the-art methods on Brodatz textures, where the values above the bars are the accuracy and corresponding standard deviations.

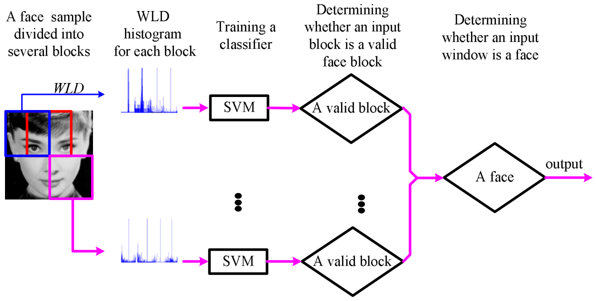

A face sample divided into several blocks
WLD histogram for each block
Training a classifier
Determining whether an input block is a valid face block
Determining whether an input window is a face
WLD
SVM
A valid block
A face
output
SVM
A valid block

Fig. 7. An illustration of WLD histogram feature for face detection.

overlapping regions and use a p-neighborhood WLD operator (here, p=8). In our case, we normalize the samples into w×h (i.e., 32×32) and derive WLD representations as follows:

We divide a face sample of size w×h into K overlapping blocks (Here, K=9 in our experiments) of size (w/2)×(h/2) pixels. The overlapping size is equal to w/4 pixels. For each block, we compute a concatenated histogram H^k, k=0, 1, …, K-1. Herein, each H^k is computed as shown in Fig. 4. In addition, for this group of experiments, we experientially let M=6, T=4, S=3. Thus, each H^k is a 72-bin histogram (Note that for each sub-histogram H_m^k, we use the same weights as shown in table 1.).

For each block, we train a Support Vector Machine (SVM) classifier using an H^k histogram feature to verify whether k^{th} block is a valid face block. If the number of the valid face blocks is larger than a given threshold Ξ, we say that a face exists in the input window. However, the value of Ξ varies with the pose of faces. For more details, please refer to section 4.4.

4.3. Dataset

The training set is composed of two sets, i.e., a positive set S_f and a negative set S_n. For the positive set, it consists of 50,000 frontal face samples. They are then rotated, translated and scaled. After these preprocessing, we obtain the set S_f including 100,000 face samples. For the negative set S_n, it consists of 31,085 images containing no faces and they are collected from Internet.

As for the test sets, we use three sets: The first one is the MIT+CMU frontal face test set, which consists of 130 images showing 507 upright faces [18]. The second one is a subset from Aleix Martinez-Robert (AR) face database [13]. Here, we choose those images with occlusions (i.e., conditions of 8-13 from the first session, and conditions of 21-26 from the second session). The resulting test set consists of 1,512 images. The third one is the CMU profile testing set [19] (441 multiview faces in 208 images).

Note that the face samples are of the size 32×32. In order to detect some faces smaller or larger than the sample size, we enlarge and shrink each input image.

4.4. Classifier Training

As described in Section 4.3, the set S_f is composed of a large number of face samples. Furthermore, we can also extract hundreds of thousands of non-face samples from the set S_n. Thus, it is extremely time consuming to train a SVM classifier using the two sets S_f and S_n. To tackle this problem, we use the resampling methods to train a SVM classifier. That is to say that we resample both the positive and negative samples during classifier training.

For the positive samples, we first randomly pick out a sub-set S_{f1} with the size N_p (in our experiments, N_p=3,000). Likewise, we also randomly crop out a sub-set S_{n1} with the size N_n (in our experiments, N_n=3,000) from the non-face database S_n. Note that for the samples in S_{n1}, we normalize their sizes to w×h (i.e., 32×32). Subsequently, we extract WLD histogram features of both the face and non-face samples as shown in Fig.7. Using the extracted features of faces and non-faces, we train a lower-performance SVM classifier. Simultaneously, we obtain a support-vector set S^1, which includes a face support-vector subset S_f^1 and a non-face support-vector subset S_n^1.

Using the resulting lower-performance SVM classifier, we test it on the two training subsets (i.e., S_f and S_n) to collect N_p misclassified face samples S_{f2} and N_n misclassified non-face samples S_{n2}. Combining the newly-collected sample sets (i.e., S_{f2} and S_{n2}) and the two support-vector subsets obtained last time(i.e., S_f^1 and S_n^1), we obtain two new training sets: (S_f^1 +S_{f2}) and (S_n^1+S_{n2}). We then train another SVM classifier with a better performance. After several iterations of the aforementioned procedure, we finally train a well performed SVM classifier.

Note that we actually train K sub-classifiers of SVM. Each sub-classifier corresponds to a block as shown in Fig. 7. Combining these K sub-classifiers, we obtain a final strong classifier.

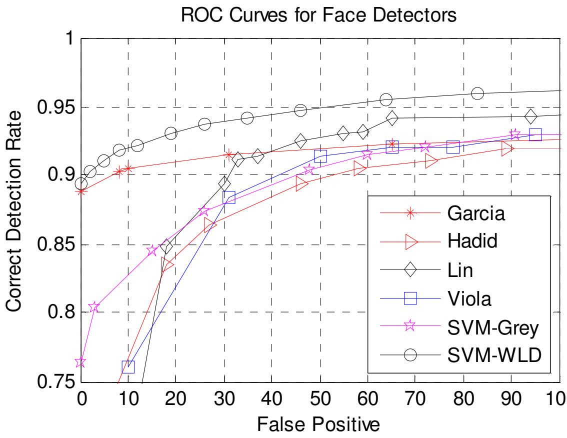


Fig. 8. A performance comparison of our method with some existing methods on MIT+CMU frontal face test set. Here, "SVM-Grey" denotes that we only use the grey intensities as input of SVM classifier, and other experimental set-ups are the same as "SVM-WLD".

Table 2. Performance of our method on AR test set.

Detection rate	False Alarms
99.74%	0
100%	3

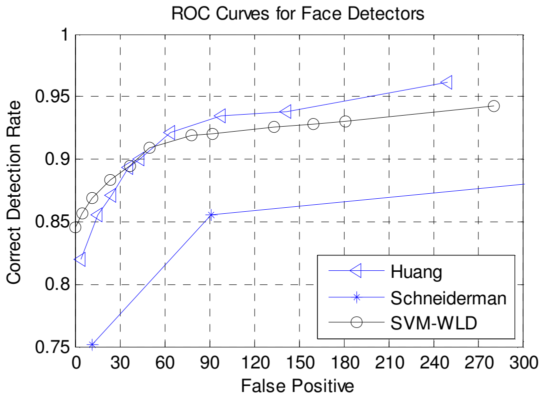


Fig. 9. A performance comparison of our method with some existing methods on CMU profile testing set.

4.5. Experimental Results

The resulting final strong SVM classifier is tested on three testing sets described in Section 4.3. Experimental results are shown in Figs. 8, 9 and table 2, respectively. Herein, we also compare the performance of the resulting SVM classifier (we call it "SVM-WLD") with some existing methods. Note that all the results by other methods in Figs. 8 and 9 are quoted directly from the original papers except Hadid [5] (which is implemented by us following their idea). During testing on these sets, the parameter Ξ takes the different values as described in Section 4.2. For MIT+CMU frontal test set, AR test set, and CMU profile test set, Ξ is equal to 8, 7 and 6 respectively.

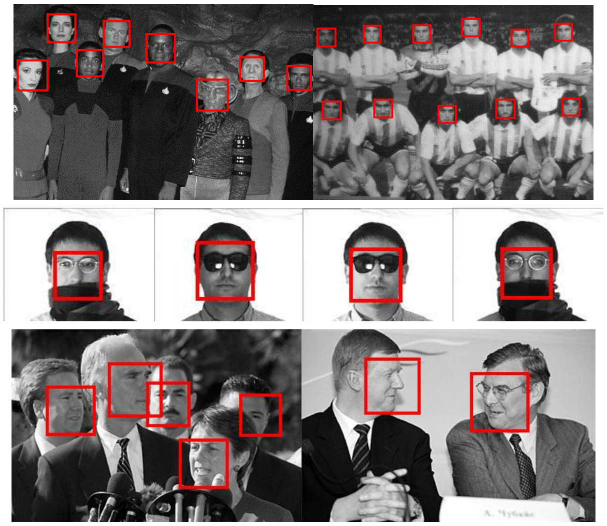


Fig. 10. Some results of our detector on the MIT+CMU frontal test set (first row), AR database (second row) and CMU profile test set (third row).

As shown in Figs. 8 and 9, we also compare the performance of our method with some existing methods. From these figures, one can find that SVM-WLD obtains a comparable performance to these methods.

However, different criteria (e.g., training examples involved and the number of scanned windows during detection etc.) can be used to favor one over another, which makes it difficult to evaluate the performance of different methods even though they use the same benchmark data sets [23]. Some results of our detector on these three test sets are shown in Fig.10.

5. Conclusion

We propose a novel discriminative descriptor WLD. It is inspired by Weber's Law, which is a law developed according to the perception of human beings. We organize WLD features to compute a histogram by encoding both differential excitations and orientations at certain locations. Experimental results clearly show that WLD illustrates a favorable performance on Brodatz textures compared to state-of-the-art methods. Especially for the face detection task, we train only one classifier but it can be used to detect the frontal, occluded and profile faces. Experimental results show that this detector works comparable to some existed methods.

Acknowledgement

This work was partially supported by National Natural Science Foundation of China under contract No.60332010, No.60673091 and No.60772071, Hi-Tech Research and Development Program of China under contract No.2006AA01Z122 and No.2007AA01Z163, 100 Talents Program of CAS, and the Academy of Finland.

Reference

[1] P. Brodatz. *Textures: A Photographic Album for Artists and Designers*, Dover Publications, New York, 1966

[2] V. Bruni and D. Vitulano. A Generalized Model for Scratch Detection, *IEEE TIP*, 2004

[3] G. Dorkó and C. Schmid. Maximally Stable Local Description for Scale Selection, *ECCV*, 2006

[4] C. Garcia and M. Delakis. Convolutional Face Finder: A Neural Architecture for Fast and Robust Face Detection. *PAMI*, 2004

[5] A. Hadid, M. Pietikäinen and T. Ahonen. A Discriminative Feature Space for Detecting and Recognizing Faces, *CVPR*, 2004

[6] C. Huang, H. Ai, Y. Li and S. Lao. High-Performance Rotation Invariant Multiview Face Detection, *PAMI*, 2007

[7] A. K. Jain. *Fundamentals of Digital Signal Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1989

[8] A.C. Jalba, M.H.F. Wilkinson and J.B.T.M. Roerdink. Morphological Hat-transform Scale Spaces and Their Use in Pattern Classification. *PR*, 2004

[9] S. Lazebnik C. Schmid J. Ponce. A Maximum Entropy Framework for Part-Based Texture and Object Recognition, *ICCV*, 2005

[10] Y. Y. Lin, T.-L. Liu, and C.-S. Fuh. Fast Object Detection with Occlusions. *ECCV*, 2004

[11] D. Lowe. Distinctive Image Features form Scale Invariant Key Points. *IJCV*, 2004

[12] B. Manjunath and W. Ma. Texture Features for Browsing and Retrieval of Image Data, *PAMI*, 1996

[13] A. M. Martinez, R. Benavente. The AR Face Database. CVC Technical Report, #24, 1998

[14] K. Mikolajczyk and C. Schmid. A Performance Evaluation of Local Descriptors. *PAMI*, 2005

[15] T. Ojala, M. Pietikäinen, T. Mäenpää. Multiresolution Gray-Scale and Rotation Invariant Texture Classification with Local Binary Patterns, *PAMI*, 2002

[16] T. Ojala, K. Valkealahti, E. Oja and M. Pietikäinen, Texture Discrimination with Multidimensional Distributions of Signed Gray Level Differences. *PR*, 2001

[17] M.-T. Pham T.-J. Cham, Fast Training and Selection of Haar Features Using Statistics in Boosting-based Face Detection, *ICCV*, 2007

[18] H. A. Rowley, S. Baluja, and T. Kanade. Neural Network-Based Face Detection, *PAMI*, 1998

[19] H. Schneiderman and T. Kanade. A Statistical Method for 3D Object Detection Applied to Faces and Cars. *CVPR*, 2000

[20] E. R. Urbach, J. B.T.M. Roerdink, and M. H.F. Wilkinson. Connected Shape-Size Pattern Spectra for Rotation and Scale-Invariant Classification of Gray-Scale Images, *PAMI*, 2007

[21] P. Viola and M. Jones. Rapid Object Detection Using a Boosted Cascade of Simple Features. *CVPR*, 2001

[22] R. Xiao, H. Zhu, H. Sun and X. Tang, Dynamic Cascades for Face Detection, *ICCV*, 2007

[23] M. H. Yang, D. Kriegman, and N. Ahuja. Detecting Faces in Images: A Survey, *PAMI*, 2002

[24] J. Zhang, M. Marszałek, S. Lazebnik, C. Schmid. Local Features and Kernels for Classification of Texture and Object Categories: A Comprehensive Study, *IJCV*, 2007