A New Feature Local Binary Patterns (FLBP) Method

Jiayu Gu and Chengjun Liu
The Department of Computer Science, New Jersey Institute of Technology, Newark, NJ 07102, USA

Abstract - *This paper presents a new Feature Local Binary Patterns (FLBP) that encodes the information of both local texture and features. The features are broadly defined by, for example, the edges, the Gabor wavelet features, the color features, etc. Specifically, a binary image is first derived by extracting feature pixels from a given image I, and then a distance vector field is obtained by computing the distance vector between each pixel and its nearest feature pixel defined in the binary image. Based on the distance vector field and the FLBP parameters, a FLBP representation of the given image I can be formed. Rather than the traditional LBP which only compares a pixel with the pixels in its own neighborhood, the FLBP can compare a pixel with the pixels in its own neighborhood as well as in other neighborhoods. We apply the FLBP to eye detection. The experimental results using the BioID database show that the FLBP method significantly improves upon the LBP method. The FLBP method displays superior representational power and flexibility to the LBP method due to the introduction of feature pixels as well as its parameters.*

Keywords: Feature Local Binary Pattern (FLBP), Local Binary Pattern (LBP), Distance Vector.

1 Introduction

The Local Binary Patterns (LBP) method [1], which defines a gray-scale invariant texture description by comparing a center pixel with its neighbors, is a popular method for texture analysis. In recent years, the LBP method has been applied in many pattern recognition tasks, such as face detection and recognition, scene and image texture classification [2 -6].

While we are inspired by the achievement of LBP, two problems occurred to us. First, LBP only compares a pixel with the pixels in its own neighborhood. We believe more information could be revealed if we can compare a pixel with the pixels in other neighborhoods. However arbitrarily comparing a pixel with any other neighborhoods might not provide useful information. Our first problem is how to locate a pixel and a neighborhood which will provide useful information after comparing them each other. Second, LBP encodes a little information about the relationship of local texture with the features, such as edges, peaks and valleys. Our second problem is how to design a texture descriptor which encodes more information of the relationship of local texture with the features. These two problems motivated us to this study. The goal of our study is to find a new texture descriptor which can solve the two problems.

We present in this paper a new Feature Local Binary Patterns (FLBP) method that encodes the information of both local texture and features. The features are broadly defined by any features which meet the requirements of specific applications, such as the edges, the intensity peaks or valleys, the Gabor wavelet features, the color features. The contributions of the paper are as follows.

- A new FLBP method is presented. The FLBP encodes both local and feature information. Rather than the traditional LBP which only compares a pixel with the pixels in its own neighborhood, the FLBP can compare a pixel with the pixels in its own neighborhood as well as in other neighborhoods. The FLBP generalize the LBP which can be considered as a special case of the FLBP. The FLBP is expected to perform better than the LBP approach for texture description and pattern recognition.
- As the FLBP method encodes both local and feature information, the performance of FLBP depends on the extraction of the feature pixels. To improve FLBP performance, we present a new feature pixel extraction method, the LBP with Relative Bias Thresholding (LRBT) method.
- For the application of FLBP on eye detection, experimental results using the BioID show that (i) the FLBP method significantly improves upon the LBP method; (ii) the new LRBT feature pixel extraction method helps improve the FLBP eye detection performance when compared with the Canny edge detection method; and (iii) the FLBP method displays superior representational power and flexibility to the LBP method due to the introduction of feature pixels as well as its parameters.

2 Feature Local Binary Patterns

Our FLBP method generalizes the LBP approach by introducing feature pixels that may be broadly defined by any features which meet the requirements of specific applications, such as the edges, the intensity peaks or valleys, the Gabor wavelet features, the color features.

First we briefly review the Local binary patterns, or LBP. LBP defines a gray-scale invariant texture description by comparing a center pixel, which is used as a threshold, with those pixels in its local neighborhood. Specifically, for a 3×3 neighborhood of a pixel $\mathbf{p} = [x, y]^t$, each neighbor is

labeled by a number from 0 to 7 as shown in Fig. 1. The neighbors of the pixel **p** may be defined below:

$$\mathbf{N}(\mathbf{p}, i) = [x_i,\ y_i]^t, \qquad i = 0, 1, 2, \ldots, 7 \tag{1}$$

where *i* is the number used to label the neighbors. The value of the LBP code of a pixel **p**(*x*, *y*) is calculated below:

$$LBP(\mathbf{p}) = \sum_{i=0}^{7} 2^i S\{G[\mathbf{N}(\mathbf{p}, i)] - G(\mathbf{p})\} \tag{2}$$

where $G(\mathbf{p})$ and $G[\mathbf{N}(\mathbf{p},i)]$ are the gray levels of the pixel **p** and its neighbor $\mathbf{N}(\mathbf{p},i)$, respectively. *S* is a threshold function that is defined as follows:

$$S(x_i - x_c) = \begin{cases} 1, & \text{if } x_i \geq x_c; \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

0	1	2
7	***p***	3
6	5	4

Fig. 1 The 3 × 3 neighborhood of a pixel **p** and the label of its neighbors

Next we review the concepts of distance transform and distance vector field. In a binary image, each pixel assumes one of two discrete values: 0 or 1. While pixels of value 0 are called the background pixels, pixels of 1 are called feature pixels. For a given metric δ, the distance transform of an image is an assignment to each pixel **p** of the distance between **p** and the nearest feature pixel **q**. The distance map can be defined as follows:

$$D(\mathbf{p}) = \delta(\mathbf{p}, \mathbf{q}) \tag{4}$$

$$\mathbf{q} = \arg \min_{\mathbf{r} \in F} \delta(\mathbf{p}, \mathbf{r}) \tag{5}$$

where *F* is the set of feature pixels of the binary image, and the distance map *D* is called the distance transform. Since the Euclidean distance is widely used in many image applications, an algorithm with linear time complexity have been developed for the fast computation of the Euclidean distance transform [7]. One shortcoming of the distance transform is that it does not contain the exact location of the nearest feature pixels. To overcome this shortcoming, a new concept of Distance Vector Field (DVF) is presented by assigning to each pixel **p** of a vector **dv** that points to its nearest feature point **q** [8]. Specifically, for a given distance metric δ, the DVF of an image may be defined as follows:

$$\mathbf{dv}(\mathbf{p}) = \mathbf{q} - \mathbf{p} \tag{6}$$

Fig. 2 shows an example of a binary image, its Distance Vector Field (DVF), and the Euclidean distance transform. Note that the upper left pixel has coordinates (1, 1) in a Cartesian coordinate system with a horizontal axis pointing to the right and a vertical axis pointing downwards. In particular, the binary image in Fig. 2(a) has only one feature pixel at the location (2, 2). Fig. 2(b) displays the DVF where the numbers are derived using Eq. 6, and Fig. 2(c) shows the Euclidean distance transform where the numbers are calculated using Eqs. 4 and 5.

0	0	0	0
0	1	0	0
0	0	0	0

(a)

1,1	0,1	-1,1	-2,1
1,0	0,0	-1,0	-2,0
1,-1	0,-1	-1,-1	-2,-1

(b)

1.4	1.0	1.4	2.2
1.0	0.0	1.0	2.0
1.4	1.0	1.4	2.2

(c)

Fig. 2 (a) An example of a binary image with only one feature pixel at the location (2, 2). (b) Distance Vector Field (DVF) (c) Euclidean distance transform

2.1 Feature Local Binary Patterns - the General Form (FLBP)

In order to define the general form of FLBP, we first introduce the concepts of True Center (TC) and Virtual Center (VC).

Definition 1: True Center (TC) is the center pixel of a given neighborhood.

Definition 2: Virtual Center (VC) is a pixel used to replace the center pixel of a given neighborhood.

Let **p** and **q** represent a pixel and its nearest feature point, respectively. Let **dv** be the distance vector of **p** pointing to **q** as defined by Eq. 6. Note that we use **dv** to replace **dv(p)** for simplicity. The TC, which may be any pixel on the path from **p** to **q**, is defined below:

$$\mathbf{C}_t(\mathbf{p}) = \mathbf{p} + \alpha_t \mathbf{dv} \tag{7}$$

where $\alpha_t \in [0, 1]$ is a parameter that controls the location of the TC. Round() is When $\alpha_t = 0$, the TC is **p**; when $\alpha_t = 1$, the TC is **q**; and when $0 < \alpha_t < 1$, the TC is a pixel on the path between **p** and **q**. Similarly, the VC, which may be any pixel on the path from **p** to **q** as well, is defined as follows:

$$\mathbf{C}_v(\mathbf{p}) = \mathbf{p} + \alpha_v \mathbf{dv} \tag{8}$$

where $\alpha_v \in [0, 1]$ is a parameter that controls the location of the VC.

The general form of FLBP is defined as follows:

$$FLBP(\mathbf{p}) = \sum_{i=0}^{7} 2^i S\{G[\mathbf{N}(\mathbf{C}_t(\mathbf{p}), i)] - G[\mathbf{C}_v(\mathbf{p})]\} \tag{9}$$

where $\mathbf{N}(\mathbf{C}_t(\mathbf{p}), i)$ defined by Eq. 1 represent the neighbors of the TC. $G[\mathbf{C}_v(\mathbf{p})]$ and $G[\mathbf{N}(\mathbf{C}_t(\mathbf{p}), i)]$ are the gray levels of the VC and the neighbor of the TC, respectively. S is a threshold function. Eq. 3 provides one definition of the S function. Another definition of the threshold function introduces a fixed bias b [9]:

$$S(x_i - x_c) = \begin{cases} 1, & \text{if } x_i \geq x_c + b; \\ 0, & \text{otherwise.} \end{cases} \quad (10)$$

To increase flexibility, we present a threshold function using a relative bias:

$$S(x_i - x_c) = \begin{cases} 1, & \text{if } x_i \geq (1 + \beta)x_c; \\ 0, & \text{otherwise.} \end{cases} \quad (11)$$

where β is a parameter that controls the contribution of x_c to the bias.

Next, we use the grayscale image shown in Fig. 3(a) to illustrate how to compute the FLBP code. We assume that the upper left pixel is at location (1, 1) in the Cartesian coordinate system with the horizontal axis pointing to the right and the vertical axis pointing downwards. As discussed before, feature pixels are broadly defined by any features which meet the requirements of specific applications. In order to make the examples easier to understand, here we define the feature pixels in Fig. 3(a) to be those with gray level greater than 98. Because the pixel at the coordinates (6, 6) in Fig. 3(a) is the only pixel whose gray level is greater than 98, the pixel becomes the only feature pixel in the binary image shown in Fig. 3(b). And this feature pixel becomes the nearest one for all the pixels in Fig. 3(a).

60	7	1	4	3	0	7
4	40	45	66	2	1	3
8	72	30	76	5	7	6
5	81	82	37	10	28	6
4	6	8	34	70	22	1
8	0	7	33	24	99	2
2	1	4	3	8	6	30

(a)

0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	1	0
0	0	0	0	0	0	0

(b)

Fig. 3 (a) A grayscale image used in the examples of FLBP computation. (b) The binary feature image derived by extracting feature pixel with gray level greater than 98 from the grayscale image.

We select the pixel **p** at coordinates (2, 2) in Fig. 3(a) as an example to compute the FLBP code. We first compute the **dv**. Given $\mathbf{p} = [2, 2]^t$, and $\mathbf{q} = [6, 6]^t$, we have $\mathbf{dv} = \mathbf{q} - \mathbf{p} = [4, 4]^t$. On the path pointed by the **dv**, we can determine the locations of TC and VC which are controlled by the parameters α_t and α_v. After TC and VC are located, we can computer the FLBP. Fig. 4 shows two examples of the computation of FLBP with different locations of TC and VC. In Fig. 4(a) given $\alpha_t = 0.75$ and $\alpha_v = 0.25$, we have $\mathbf{C}_t(\mathbf{p}) = \mathbf{p} + \alpha_t\mathbf{dv} = [5, 5]^t$, and $\mathbf{C}_v(\mathbf{p}) = \mathbf{p} + \alpha_v\mathbf{dv} = [3, 3]^t$. Therefore, the TC is the pixel at location (5, 5) and the VC is the pixel at location (3, 3). According to Eq. 9 we replace the gray level 70 of TC at location (5, 5) by the gray level 30 of VC at location (3, 3), and threshed the neighbors of the TC. We have the binary FLBP code: $FLBP(2, 2) = 10001010$. Fig. 4(b) shows another example of the $FLBP(2, 2)$ computation when $\alpha_t = 0.25$, and $\alpha_v = 0.75$. Similarly, we locate the TC is the pixel at location (3, 3) and the VC is the pixel at location (5, 5). The binary FLBP code becomes: $FLBP(2,2) = 00010111$.

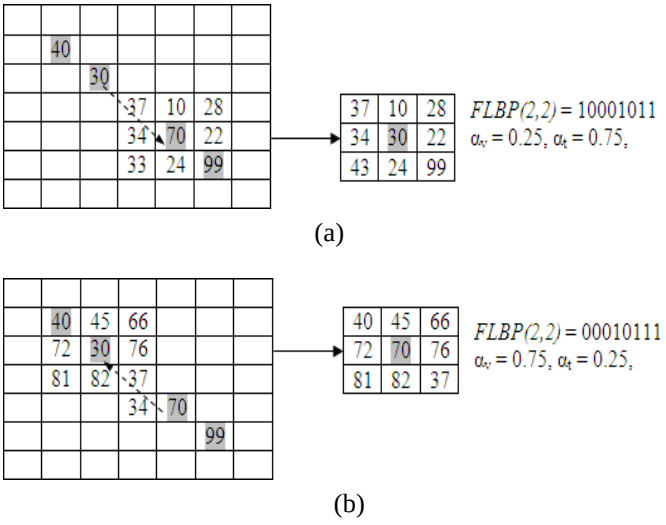


Fig. 4 The computation of FLBP for the pixel at (2, 2) (a) An example when TC ($\alpha_t = 0.75$) is at (5, 5) and VC ($\alpha_v = 0.25$) is at (3, 3) (b) An example when TC ($\alpha_t = 0.25$) is at (3, 3) and VC ($\alpha_v = 0.75$) is at (5, 5)

FLBP has the following special cases:

- FLBP Form 1 — FLBP1
  When $\alpha_v = 0$, pixel **p** becomes the VC, and the TC may be any pixel on the distance vector **dv**. FLBP1 compares the center pixel **p** which is used as the threshold, with the neighbors of the TC.
- FLBP Form 2 — FLBP2
  When $\alpha_t = 0$, pixel **p** becomes the TC, and the VC may be any pixel on the distance vector **dv**. FLBP2 compares the VC which is used as the threshold, with the neighbors of the center pixel **p**.
- LBP is a special case of FLBP
  When $\alpha_v = \alpha_t = 0$, the VC and TC coincide with the center pixel **p**, and FLBP becomes LBP, where no feature pixels are involved. LBP compares the center pixel **p**, with its own neighbors.

2.2 Feature Local Binary Patterns - Form 1 (FLBP1) and Form 2 (FLBP2)

FLBP1, which is a special case of FLBP, when $\alpha_v = 0$, may be defined as follows:

$$FLBP1(\mathbf{p}) = \sum_{i=0}^{7} 2^i S\{G[\mathbf{N}(\mathbf{C}_t(\mathbf{p}), i)] - G(\mathbf{p})\} \qquad (12)$$

where $G(\mathbf{p})$ and $G[\mathbf{N}(\mathbf{C}_t(\mathbf{p}), i)]$ are the gray levels of the center pixel **p** and the neighbors of the TC, respectively. Eq. 12 shows that FLBP1 compares the center pixel **p** with the neighbors of the TC, which may be any pixel on the distance vector **dv**.

FLBP2, which is another special case of FLBP, when $\alpha_t = 0$, may be defined as follows:

$$FLBP2(\mathbf{p}) = \sum_{i=0}^{7} 2^i S\{G[\mathbf{N}(\mathbf{p}, i)] - G[\mathbf{C}_v(\mathbf{p})]\} \qquad (13)$$

where $G[\mathbf{C}_v(\mathbf{p})]$ and $G[\mathbf{N}(\mathbf{p}, i)]$ are the gray levels of the VC and the neighbors of the center pixel **p**, respectively. Eq. 13 shows that FLBP2 compares the VC, which may be any pixel on the distance vector **dv**, with the neighbors of the center pixel **p**.

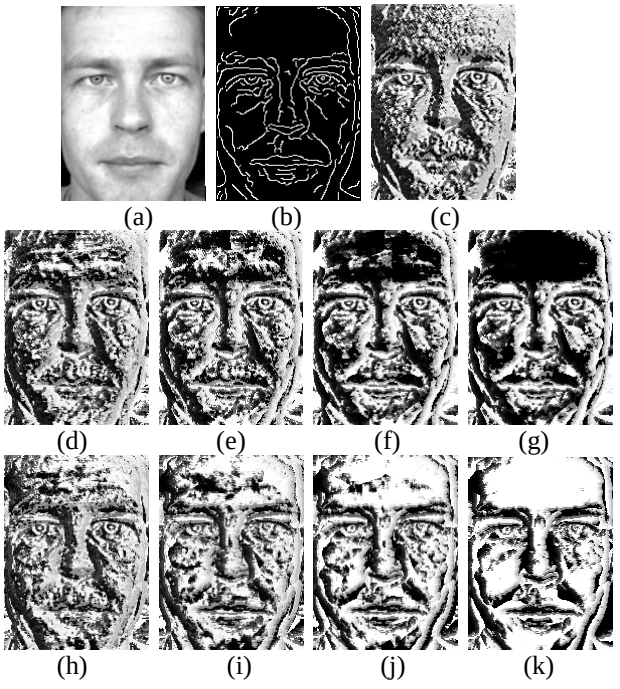


Fig. 5 (a) A face image. (b) The binary feature image of the face image derived using the Canny edge detector. (c) The LBP representation of the face image (d) - (g) The FLBP1 representation when α_t = 0.25, 0.5, 0.75 1, respectively. (h) - (k) The FLBP2 representation when α_v = 0.25, 0.5, 0.75 1, respectively.

Fig. 5 shows an example of the FLBP representation of a face image, where the traditional LBP representation is also included for comparison. Specifically, Fig. 5(a) and (b) display a face image and its binary feature image derived using the Canny edge detector. Fig. 5(c) shows the LBP representation of the face image of Fig. 5(a). Fig. 5(d), (e), (f), and (g) exhibit the FLBP1 representations when α_t = 0.25, 0.5, 0.75, 1, respectively. Fig. 5(h), (i), (j), and (k) show the FLBP2 representations when α_v = 0.25, 0.5, 0.75, 1, respectively.

3 LBP with Relative Bias Thresholding for Feature Pixel Extraction

As FLBP encodes both local and feature information, the performance of FLBP depends on the extraction of the feature pixels. We present a new feature pixel extraction method, the LBP with Relative Bias Thresholding (LRBT) method. The LRBT method first computes the LBP representation of an input grayscale image using the relative bias threshold function of Eq. 11 with a given β. An LBP image is then defined by the LBP representation. The LRBT method converts the LBP image to a binary LRBT feature image, whose feature pixels correspond to those whose LBP code is greater than 0, and the background pixels correspond to the pixels in the LBP image with the LBP code 0. Note that different binary LRBT feature images can be generated with different β values. Fig. 6 shows the LBP images and the corresponding binary LRBT feature images of a face image shown in Fig. 5(a)

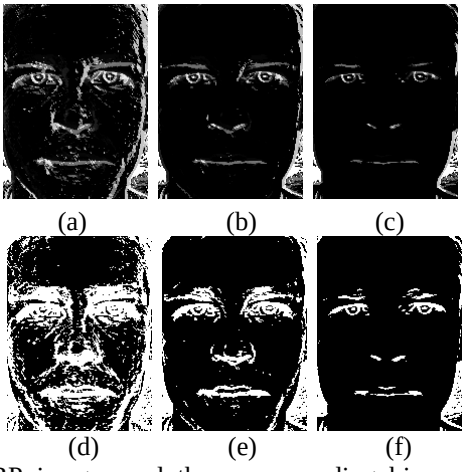


Fig. 6 LBP images and the corresponding binary LRBT feature images of a face image. (a)–(c) The LBP images when β = 0.05, 0.1, 0.2, respectively. (d)–(f) The binary LRBT feature images when β = 0.05, 0.1, 0.2, respectively.

Fig. 7 shows the FLBP1 and FLBP2 representations of the face image that applies the binary LRBT feature image shown in Fig. 6(e). Specifically, Fig. 7(a), (b), (c), and (d) display the FLBP1 representations when α_t = 0.25, 0.5, 0.75, 1, respectively. Fig. 7(e), (f), (g), and (h) exhibit the FLBP2 representations when α_v = 0.25, 0.5, 0.75, 1, respectively.

Note that in our experiments on eye detection in Sect. 5 we apply both the edge feature pixels from the Canny edge detector and the LRBT feature pixels from the LRBT feature pixel extraction method. The experimental results show that the FLBP-based eye detection method achieves

better performance when applying the LRBT feature pixel extraction method than the Canny edge detector.

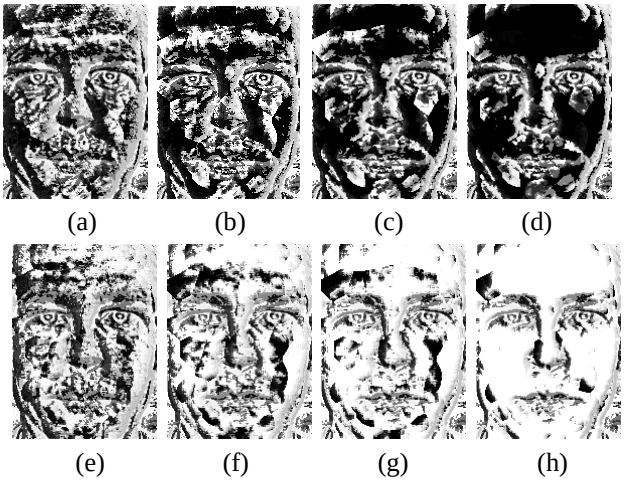


Fig. 7 The FLBP1 and FLBP2 images that uses the binary LRBT feature image of Fig. 6(e) when β = 0.1. (a)–(d) The FLBP1 images when α_t = 0.25, 0.5, 0.75, 1, respectively. (e)–(h) The FLBP2 images when α_v = 0.25, 0.5, 0.75, 1, respectively

Fig. 5 and Fig. 7 show that different parameter values lead to different FLBP representations. The feature pixels used in the FLBP are broadly defined. Different feature pixels also lead to different FLBP representations. Different FLBP representations can serve different purposes for texture description and pattern recognition. The FLBP method encodes much richer information than the LBP method does. Not only does the FLBP encode both local and feature information, but it also enhances its representational power and flexibility by incorporating a number of parameters, such as the CT parameter α_t, the VT parameter α_v, as well as the relative bias parameter β.

4 Application of FLBP to Eye Detection

We apply the FLBP method on eye detection. Fig. 8 shows the system architecture of our FLBP-based eye detection method that consists of three major steps. First, a binary image, which contains the feature pixels is derived by applying the new LRBT feature pixel extraction method. Second, the FLBP representation of the face image is formed based on the grayscale image and a distance vector field or DVF, which is obtained by computing the distance vector between each pixel and its nearest feature pixel defined in the binary image. Finally, for eye detection, an eye template is first constructed from a number of training eye samples, and each eye candidate is then compared with the eye template based on the FLBP histogram and a similarity measure, whose computation is implemented by a fast algorithm.

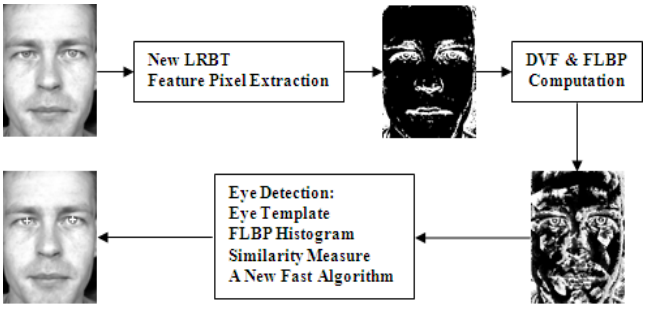


Fig. 8 The system architecture of our FLBP-based eye detection method

4.1 A Fast Algorithm for FLBP Histogram and Similarity Computation

We present in this section a fast algorithm for FLBP histogram and similarity computation. Let the eye template and an eye candidate window be divided into a grid of $r \times c$ cells. A FLBP histogram of a cell is formed by the FLBP codes of the pixels in the cell. The eye template is defined by the $r \times c$ mean FLBP histograms of the training eye samples. Let T and C represent the eye template and eye candidate windows, respectively. We apply the following similarity measure $M(C, T)$ to compare T and C:

$$M(C,T) = -\sum_{i=1}^{g}\sum_{j=1}^{b}\frac{(C_{i,j} - T_{i,j})^2}{C_{i,j} + T_{i,j}} \qquad (14)$$

where $C_{i,j}$ represents the j-th bin of the FLBP histogram of the i-th cell of the eye candidate window, $T_{i,j}$ represents the j-th bin of the FLBP histogram of the i-th cell of the eye template, $g = rc$ is the total number of cells of the $r \times c$ grid, and b is the number of bins of a histogram.

An exhaustive search of eye location may compare an eye candidate window centered at every pixel in a face image with the eye template. The pixel whose eye candidate window has the largest similarity value with the eye template is the location of the detected eye. Let the spatial resolution of the face image and the eye template be $m \times n$ and $w \times h$, respectively. The computational complexity of deriving the DVF and FLBP is O(mn), and the complexity of computing the FLBP histogram and similarity for an eye candidate window is O(wh) and O(gb), respectively. As a result, the total time for searching a face image is O($whmn$) and O($gbmn$) for FLBP histogram and similarity computation, respectively.

We now present a fast algorithm that can reduce the computational complexity of FLBP histogram and similarity computation. Fig. 9 shows a search region, which contains an eye candidate window with a spatial resolution of $w \times h$, the top left pixel at (x_1, y_1), and the lower right pixel at (x_w, y_h). For simplicity, let us use the upper left pixel to represent an eye candidate window. Suppose we have computed the FLBP histogram and similarity for window at (x_1, y_1). We move the eye template to the next column to compare the next eye

candidate window at (x_1 + 1, y_1) with the eye template. Now the difference between these two eye candidate windows resides in column x_1 and column x_w + 1. If we remove column x_1 from window (x_1, y_1).and add column x_w + 1, the new window is window (x_1 + 1, y_1). Since we have already computed the FLBP histogram and similarity for window (x_1, y_1), we can obtain the new results for window (x_1 + 1, y_1) by examining the difference between columns x_1 and x_w +1.

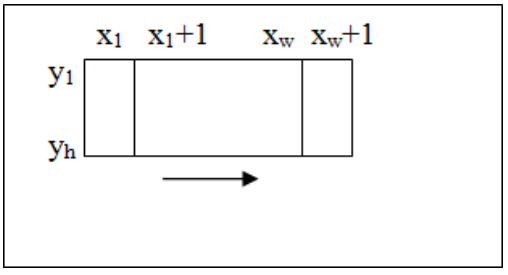


Fig.9 A search region that contains an eye candidate window with a spatial resolution of $w \times h$, the top left pixel at (x_1, y_1), and the lower right pixel at (x_w, y_h)

The fast algorithm works as follows.

First, assign the FLBP histogram and similarity of window (x_1, y_1) to window (x_1 + 1, y_1).

Second, update the FLBP histogram for window (x_1 + 1, y_1) as follows: (i) for each pixel in column x_1, reduce 1 from the histogram bin corresponding to its FLBP code; and (ii) for each pixel in column x_1 + 1, add 1 to the histogram bin corresponding to its FLBP code. The computational complexity for FLBP histogram computation is now reduced to O(h) from O(wh).

Third, update the similarity for window (x_1 + 1, y_1) as follows: (i) save the similarity values for every histogram bin of window (x_1, y_1); (ii) for every histogram bin that has been updated, subtract the old similarity value, recalculate the similarity value, and add the new value to the similarity. The computational complexity for similarity computation is now reduced to O(h) from O(b), as h, which is the height of the eye candidate window, is much smaller than b, which is the number of bins of the histogram. Note that the reduced computational complexity is independent of b. The significance of the fast algorithm is that it runs equally fast no matter it is applied to the three-level texture analysis method with 6561 histogram bins or the LBP method with 256 histogram bins.

Fig. 9 shows that the eye candidate window moves horizontally to the next column. If the eye candidate window moves vertically, the fast algorithm works as well by examining the difference between the rows y_1 and y_h + 1. The computational complexity for FLBP histogram and similarity computation is O(w), where w is usually not equal to h. If w is greater than h, moving the window row by row is faster, otherwise, moving the window column by column is faster.

5 Experimental Results

We assess the eye detection performance of FLBP and LBP methods using the public BioID databases which contains 1,521 grayscale frontal face images with spatial resolution of 384 × 286. All facial images in our experiments are cropped and normalized to the size of 132 × 178. To construct the eye template, we collected 70 pairs of eye samples that are not from the BioID database. As only the right eye template is constructed due to the symmetry between right and left eyes, the 70 left eyes are flipped horizontally to double the number of the right eye samples. To detect the left eye, we first flip the face image horizontally, and then detect eye in the flipped image by comparing it with the right eye template. Eye samples are cropped to 37 × 17.

Eye detection performance is determined by a relative distance error and is defined as follows:

$$\gamma = d_1 / d_2 \quad (15)$$

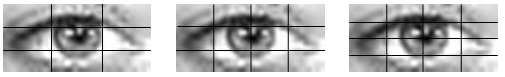
where d_1 is the Euclidean distance between the detected eye center and the ground truth eye center, and d_2 is the interocular distance between the two ground truth eye centers. The detected eye center is considered inside the eye region, inside the iris, and inside the pupil, when $\gamma \leq 0.25$, 0.1, 0.05, respectively. We use the success rate for $\gamma \leq 0.25$, 0.1, 0.05, to assess the performance of the FLBP-based eye detection method.

As the FLBP method may apply a neighborhood of different size, we assess the effect of 3 × 3 and 5 × 5 neighborhood on eye detection performance. Our experimental results show that the 5×5 neighborhood shown in Fig. 10 is better than the 3×3 neighborhood shown in Fig. 1.

0		**1**		**2**
7		**8**		**3**
6		**5**		**4**

Fig.10 The 5 × 5 neighborhood where only the labeled neighbors with numbers from 0 to 7 are used to compute the FLBP code and the center pixel with a label 8 is used as the threshold

We also assess the effect of the grid size of the eye candidate window on eye detection performance on BioID database. We divide the eye candidate window into three different grids which are 3 × 3, 3 × 4, and 4 × 4 grids shown in Fig. 11. Our experimental results show that 3 × 4 grid yields the best overall eye detection performance.



(a) (b) (c)

Fig. 11 The eye window grids (a) The 3 × 3 grid (b) The 3 × 4 grid. (c) The 4 × 4 grid

After determining the best neighborhood size and grid size, we assess the eye detection performance of FLBP and LBP methods comparatively on the BioID database. We apply the 5×5 neighborhood size and divide the eye template and candidate windows to 3 × 4 grid in all experiments. The feature pixels for the FLBP method are derived by either the LRBT method or the Canny edge detector in our experiments.

Table 1 shows the eye detection success rates of the FLBP1, FLBP2, and LBP methods. The results show that the success rates of both FLBP1 and FLBP2 when γ = 0.25, 0.1 and 0.05 are around 98%, 95% and 87%, respectively. In comparison, the success rates of LBP are about 92%, 90% and 83%. Therefore, the results show that our FLBP method significantly improves the eye detection performance upon the LBP method. Table 1 also shows the success rates of the FLBP method using the features extracted by LRBT method are higher than the success rates of the FLBP method using the features extracted by Canny edge detector. This finding indicates that the new LRBT feature pixel extraction method helps improve the FLBP eye detection performance when compared with the Canny edge detection method.

Table 1 The eye detection success rates of the FLBP1, FLBP2, and LBP methods

Method		$\gamma \leq$ 0.25	$\gamma \leq$ 0.1	$\gamma \leq$ 0.05
FLBP 1	LRBT, β = 0.2, α_t = 0.12	97.86	95.10	87.41
	LRBT, β = 0.1, α_t = 0.25	98.03	94.74	87.21
	LRBT, β = 0.15, α_t = 0.12	98.03	95.00	87.15
	Canny Edge Pixels, α_t = 0.12	97.80	94.67	86.65
FLBP 2	LRBT, β = 0.2, α_v = 0.25	98.65	95.23	87.84
	LRBT, β = 0.15, α_v =0.25	98.55	95.04	87.51
	LRBT, β = 0.1, α_v = 0.25	97.90	94.67	86.88
	Canny Edge Pixels, α_v = 0.12	97.63	94.48	86.46
LBP		92.34	90.34	83.14

6 Conclusion

We present in this paper a new Feature Local Binary Patterns (FLBP) method which encodes both local and feature information. The features are broadly defined by any features which meet the requirements of specific applications. The FLBP generalizes the LBP which can be considered as a special case of the FLBP. The FLBP method displays superior representational power and flexibility to the LBP method due to the introduction of feature pixels as well as its parameters. We assess the FLBP method on eye detection using the BioID database. The experimental results show that the FLBP method significantly improves upon the LBP method. We present a new feature pixel extraction method, the LBP with Relative Bias Thresholding (LRBT) method. The experimental results show that the new LRBT feature pixel extraction method helps improve the FLBP eye detection performance when compared with the Canny edge detection method.

7 References

[1] T. Ojala, M. Pietikainen and D. Harwood, "A comparative study of texture measures with classification based on feature distributions," Pattern Recognition, vol. 29, no. 1, pp. 51–59, 1996.

[2] T. Ahonen, A. Hadid, and M. Pietikainen, "Face recognition with local binary patterns," in 8th European Conference on Computer Vision, Prague, Czech Republic, May 11-14, 2004, pp. 469–481,

[3] X. Huang, S. Li, and Y. Wang, "Shape localization based on statistical method using extended local binary pattern," in Third International Conference on Image and Graphics, Hong Kong, China, Dec.18-20, 2006, pp. 184–187.

[4] H. Jin, Q. Liu, H. Lu, and X. Tong, "Face detection using improved lbp under bayesian framework," in Third International Conference on Image and Graphics, Hong Kong, China, Dec.18-20, 2006, pp. 306–309.

[5] Z. Liu and C. Liu, "Fusion of color, local spatial and global frequency information for face recognition," Pattern Recognition, vol. 43, no. 8, pp. 2882–2890, 2010.

[6] S. Banerji, A. Verma, and C. Liu, "Novel color LBP descriptors for scene and image texture classification," in 15th International Conference on Image Processing, Computer Vision, and Pattern Recognition, 2011.

[7] C. Maurer, R. Qi, and V. Raghavan, "A linear time algorithm for computing exact Euclidean distance transforms of binary images in arbitrary dimensions," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 25, no. 2, pp. 265–270, 2003.

[8] P. Danielson, "Euclidean distance mapping," Computer Graphics and Image Processing, vol. 14, no. 3, pp. 227–248, 1980.

[9] V. Kumar, N. Rao, and A. Rao, "Reduced texture spectrum with lag value based image retrieval for medical images," International Journal of Future Generation Communication and Networking, vol. 2, no. 4, pp. 39–48, 2009.