Accurate Eye Localization in Low and Standard Definition Content

Bart Kroon  
Delft University of Technology  
Mekelweg 4, Delft, Netherlands  
b.kroon@tudelft.nl

Sabri Boughorbel  
Philips Research Eindhoven  
HTC 34, Eindhoven, Netherlands  
sabri.boughorbel@philips.com

Alan Hanjalic  
Delft University of Technology  
Mekelweg 4, Delft, Netherlands  
a.hanjalic@tudelft.nl

Abstract

In this paper we address the problem of eye localization in low and standard definition content, such as webcam-generated and TV images. We present a probabilistic eye localization method based on well-known multiscale local binary patterns (LBPs), which provide a simple and powerful spatial description of texture, and are robust to the noise typical to low and standard definition content. Our primary contribution is in the proposed method of combining binary patterns (LBPs) that is targeted towards achieving spatial accuracy under mentioned conditions. Evaluation performed on a standard dataset of webcam-quality images shows that our approach has superior performance with respect to the state of the art, while having a reasonable complexity and a low memory footprint.

1. Introduction

The increasing amount of multimedia content has raised new research problems in the area of content analysis. In particular the easy access to capturing devices has drastically increased the amount of low and standard definition content. This type of content is characterized by a relatively poor quality and uncontrolled conditions, which makes eye localization a challenging research topic, also because it is not possible to use other modalities such as RFID information or infrared cameras.

Accurate localization of the eyes is a necessary step for many face related applications such as face registration, face recognition, face validation, face clustering, blink detection, and gaze tracking. For instance, accurate face registration is achieved by (affinely) transforming the face image to align the eyes on predefined image positions [14]. The relation between eye localization accuracy and face recognition performance is analyzed in more detail in [14][15][18].

In spite of considerable amount of previous work on the subject (e.g., [3][4][5][6][10][11][12][15][16][18]), eye localization is not a solved problem. This especially holds for realistic application contexts in which the resolution of faces may vary, and is substantially lower than in high resolution (e.g., digital camera produced) images. For instance in a typical 320 × 240 webcam image the eye distance is about 50 pixels versus a few hundred for digital camera images.

In this paper we focus on low and standard definition content, and propose a method that uses the entire eye region to learn the eye pattern and therefore does not require clear visibility of the pupils. We base our method on local binary patterns (LBPs) [2][9] that not only provide a low complexity of the localizer, but also lead to a superior performance compared to state-of-the-methods. Our primary contribution is our model which is able to achieve both high spatial accuracy and robustness.

After introducing the problem of eye localization in Section 2 we describe our proposed method in Section 3 and evaluate it in Section 4 on representative datasets. Section 5 concludes the paper.

2. The Eye Localization Problem

To find eyes in an image we can either search for them directly in the entire image [7][17], or rely on the output of a face detector indicating that eyes are present in the image. The first case is the eye detection problem, while in the second, more investigated case, eyes only need to be localized within the bounding box supplied by a face detector. Generally an eye localizer can be described as a function

\[ f : (b, I) \rightarrow (\hat{\eta}_L, \hat{\eta}_R) \]  

where \( I \) is the image, \( b = (b_x, b_y, b_w, b_h) \) is the vector containing the main parameters of the face bounding box, namely the center position, width, and height, respectively, and \( \hat{\eta}_L = (\hat{\eta}_{Lx}, \hat{\eta}_{Ly}) \) and \( \hat{\eta}_R = (\hat{\eta}_{Rx}, \hat{\eta}_{Ry}) \) are the vectors indicating an estimate of the left and right eye center position respectively. We assume that the left eye is on the left side of the image (\( \hat{\eta}_{Lx} < \hat{\eta}_{Rx} \)) and that all sizes and positions are expressed in pixels if not stated otherwise.

The performance of an eye localization method is typically measured by computing the maximum of the left and
right eye position estimation errors and by normalizing this error to the annotated eye distance [10], that is

$$\epsilon_{\text{max}} = \frac{\max\{\|\hat{\eta}_L - \eta_L\|_2, \|\hat{\eta}_R - \eta_R\|_2\}}{\|\eta_L - \eta_R\|_2},$$  \hspace{1cm} (2)$$

where $(\eta_L, \eta_R)$ are the annotated eye positions.

An alternative measure for the eye localization error is the root mean square (RMS) error [14], defined as

$$\epsilon_{\text{RMS}} = \sqrt{\|\hat{\eta}_L - \eta_L\|_2^2 + \|\hat{\eta}_R - \eta_R\|_2^2} \div \|\eta_L - \eta_R\|_2. \hspace{1cm} (3)$$

The measures are related since an RMS error of $\epsilon_{\text{RMS}}$ bounds the maximum error $\epsilon_{\text{max}}$ to

$$\epsilon_{\text{max}} \in \left[\epsilon_{\text{RMS}} \sqrt{\frac{1}{2}} \epsilon_{\text{RMS}} \right]. \hspace{1cm} (4)$$

Acceptable $\epsilon_{\text{max}}$ error limits for applications such as face recognition are 5% or 7%.

### 3. Description of the Proposed Method

As depicted in Figure 1, the proposed method consists of four major parts and is used in conjunction with a face detector that accepts an image $I$ and outputs a bounding box $b$ for each detected face.

During the training phase, the prior learning part (Section 3.2) uses the annotation of the eye positions $(\eta_L, \eta_R)$ and the detected face bounding boxes $b_i$ of each image in the training set to estimate prior eye locations $(\hat{\eta}_L, \hat{\eta}_R)$ and the size of the search region $(w_S, h_S)$ that is used for both eyes (Figure 2). In the current implementation of the method we use a frontal face detector. The benefits of using a multiview face detector will be explained in Section 5. Then all images $I_i$ in the training set are rescaled and processed by the multiscale LBP extraction part (Section 3.1) to extract LBP windows $F_{x,y}$ for each position within the search region $(x, y)$. The amount of scaling is determined by the annotated eye positions $(\eta_L, \eta_R)$. These windows are then used by the model learning part (Section 3.3) to learn a LBP model consisting of histograms $H_{\text{eye}}$ and $H_{\text{eye}}$ for each scale and eye. The histogram $H_{\text{eye}}$ is based on the LBP description at the eye position and $H_{\text{eye}}$ on the descriptions within the entire search region $(w_S, h_S)$.

During the testing phase, a new image is processed by the multiscale LBP extraction part and the localizer (Section 3.4) respectively. Because the annotated eye positions are not available, the amount of scaling is now determined by the eye position priors $(\hat{\eta}_L, \hat{\eta}_R)$. The localizer part calculates the model response within the search region $(w_S, h_S)$ for each scale and both eyes. The response images of multiple scales are combined to form one for each eye and then the positions of the maximum responses is used as the estimated eye positions $(\hat{\eta}_L, \hat{\eta}_R)$.

### 3.1. Extracting multiscale local binary patterns

Eyes are considered to be deformable objects appearing under various poses and lighting conditions. The amount of variability is especially high in video with relatively low resolution such as webcam images. The most accurate existing method [11] tackles the variability issue directly by choosing features (Gabor wavelets) that are to some extent invariant to lighting and pose. The extraction of Gabor wavelets is however complex in comparison to local binary patterns [9] which have similar invariance properties and are successfully applied in related problems such as face validation. By basing our eye localization approach on local binary patterns we are able to achieve a high performance at a low complexity compared to the existing methods.

A local binary pattern can be computed for a given pixel as shown in Figure 3. The center pixel is compared with each of the surrounding pixels. If the intensity of the surrounding pixel is higher, this gives a 1, otherwise a 0. As we
only work with the sign of intensity difference, the obtained patterns are invariant to contrast, brightness and gamma. Different light directions can however result in different patterns. In this paper we restrict the size of the LBPs to $3 \times 3$ pixels to limit the classifier model complexity (as explained in Section 3.3).

For the purpose of eye localization and in the context of the scheme in Figure 1, we used LBPs to describe the spatial distribution of eye texture. For each scale $k$, we resize, translate and rotate the image $I$ such that the eye locations $(\eta_L, \eta_R)$ move to preset positions with a chosen eye distance. In the testing phase the real positions are replaced by the priors $(\bar{\eta}_L, \bar{\eta}_R)$. Concatenating all binary values of a LBP produces a bitstring, which in case of a $3 \times 3$ pixel neighborhood is of range $p \in \{0, 255\}$. For a transformed image and potential eye position $(x, y)$ we compute the LBPs on a $n \times m$ region giving us LBP windows as matrices $F_{x,y} \in \{0, 255\}^{n \times m}$.

### 3.2. Learning eye position priors

The face bounding box provides useful input for estimating the position of the eyes. To use this information we have to map the bounding box $b$ of a face in a new image to a set of eye position priors $(\bar{\eta}_L, \bar{\eta}_R)$. That is, we have to estimate

$$\begin{bmatrix} \bar{\eta}_L \\ \bar{\eta}_R \end{bmatrix} = \mathbb{E}\left[ \begin{bmatrix} \eta_L \\ \eta_R \end{bmatrix} \right] | b,$$

which can be modelled as

$$\begin{bmatrix} \bar{\eta}_L \\ \bar{\eta}_R \end{bmatrix} = b_w \begin{bmatrix} c_L \\ c_R \end{bmatrix} + \begin{bmatrix} b_x \\ b_y \end{bmatrix}.$$  

The 2-D vector constants $c_L$ and $c_R$ are considered to be model parameters and can be estimated using the following expression obtained based on (6) and on a training set with $N$ faces with detected bounding boxes $(b_{x_i}, b_{y_i}, b_{w_i}, b_{h_i})$, and annotated eye positions $\eta_{L_i}$ and $\eta_{R_i}$:

$$\begin{bmatrix} c_L \\ c_R \end{bmatrix} = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{b_{w_i}} \left( \begin{bmatrix} \eta_{L_i} \\ \eta_{R_i} \end{bmatrix} - \begin{bmatrix} b_{x_i} \\ b_{y_i} \end{bmatrix} \right).$$  

The constants $c_L$ and $c_R$ allow to calibrate the eye localizer for use with any face detector.

When searching for eyes within the scaled images, the choice of the search region is important for the performance of the method. As shown in Figure 2, we limit the search to a rectangular area centered at the eye position priors $(\bar{\eta}_L, \bar{\eta}_R)$ with fixed width and height $(w_S, h_S) = (e, \frac{3}{4}e)$, measured in eye distance units

$$e = \| \eta_L - \eta_R \|_2.$$  

In case of testing, the eye distance is an estimate $\tilde{e}$ based on the eye position priors

$$\tilde{e} = \| \tilde{\eta}_L - \tilde{\eta}_R \|_2 = b_w \| c_L - c_R \|_2.$$  

Experiments have shown that the size of the search region is not critical to the performance of the algorithm as long as the region is large enough. A bigger search region did increase the computational complexity without providing more performance. Making the search region smaller would lead to a decrease in performance.

### 3.3. Learning the LBP model

The multiscale LBP extractor provides a LBP window $F_{x,y}$ for each position $(x, y)$ and for all scales $k$. Considering only one scale, the maximum a posteriori eye position $\eta_k$ can be determined by taking the maximum of the likelihood ratio

$$\hat{\eta}_k = \arg\max_{(x,y)} \frac{P(\text{eye} | F_{x,y})}{P(\neg \text{eye} | F_{x,y})}.$$  

Using Bayes rule, the argument of (10) is equal to

$$\frac{P(\text{eye} | F_{x,y})}{P(\neg \text{eye} | F_{x,y})} = \frac{P(\text{eye})P(F_{x,y} | \text{eye})}{P(\neg \text{eye})P(F_{x,y} | \neg \text{eye})}.$$  

If we ignore dependencies among pixels to reduce model complexity, and ignore constant terms that only affect scaling, the naive Bayesian formulation of the localization problem can be expressed as

$$\frac{P(\text{eye} | F_{x,y})}{P(\neg \text{eye} | F_{x,y})} \propto \prod_{i,j=1}^{n,m} P(F_{x,y}(i, j) | \text{eye}) \frac{P(F_{x,y}(i, j) | \neg \text{eye})}. $$  

To ensure computational stability, we substitute (12) by a logarithmic representation (log likelihood) and we approximate the probabilities by calculating histograms on a training set. In this way for each scale $k$ we create a log likelihood map of the image

$$L_k(x, y) = \sum_{i,j=1}^{n,m} \log \frac{H_{\text{eye}}(i, j, F_{x,y}(i, j)) + \lambda}{H_{\neg \text{eye}}(i, j, F_{x,y}(i, j)) + \lambda}.$$  

where the histograms $H_{\text{eye}}$ and $H_{\neg \text{eye}}$ of size $n \times m \times 256$ count how often patterns occur at each pixel position within
the search region. Due to the sparseness of histograms bins, a regularization constant of $\lambda$ is added to avoid numerical problems. The histograms are estimated using training samples with annotated eye positions, where the positive histograms are based on the correct eye positions and the negative histograms from other locations within the search region $(w_S, h_S)$.

An important aspect of the proposed method is, that (13) implies spatial weighting. The influence of spatial positions $(i, j)$ depends namely on the dissimilarity of eye and non-eye distributions. In Figure 4, the Kullback-Leibler divergence is used to visualize this dissimilarity.

3.4. Locating eyes in new images

Section 3.3 explains how the model is learned for each scale and both eyes. Our multiscale approach is based on combining the responses $L_k(x, y)$ to one multiscale response $L(x, y)$ (Figure 5). We assume that all scales are equally important:

$$L(x, y) = \sum_k L_k(x, y).$$

(14)

Because different scales have different resolutions, interpolation of $L_k(x, y)$ is required. The estimated eye positions $(\bar{\eta}_L, \bar{\eta}_R)$ are determined by

$$\bar{\eta} = \arg \max_{x, y} L(x, y).$$

(15)

4. Evaluation

4.1. Choice of the training set

For all algorithms that learn a model, it holds that the training set is of influence on the performance of the algorithm. This however, makes it more difficult to reproduce the results reported in papers. Therefore we decided to use the frontal images of the publicly available dataset YaleB [8] for training our algorithm. The first advantage of this dataset is that there is no ambiguity in the eye positions because the annotation is public. The second advantage stems from the controlled lighting conditions that ensure that the training includes most of the binary patterns that could be encountered in an arbitrary image. This is because in YaleB the scene setup is designed to cover a wide range of lighting directions. We are therefore confident that the eye localizer is trained to be robust to lighting conditions. A disadvantage of YaleB is that the resulting model is not as invariant to pose as in the case of optimal training.

4.2. Choice of the test sets

The faces found in webcam-generated and standard definition video, have a medium resolution, and eye distances are expected to be small. Furthermore views are not strictly frontal, and lighting conditions may vary. The best way to test if our system works under these conditions is to use movies or home videos as test data. There is however no publicly available dataset that directly matches these requirements. We found that BioID [10] is the dataset that matches the medium resolution criteria with an average eye distance of 54.9 pixels. An extensive evaluation of the state-of-the-art also revealed that BioID is often used for evaluation (see Table 1) and thus allows for direct comparison with those works. We therefore choose BioID as one component of our test set. The other component consists of self-collected webcam images and standard definition video frames taken from movies.

4.3. Detecting faces in training and test images

Realistic evaluation of an eye localizer is only possible in conjunction with a face detector or tracker (Figure 1). Unfortunately other contributions such as [11] are unclear on the precise procedure.

We detect faces using the OpenCV [1] face detector with the default frontal classifier and default parameters. As the eye localizer by definition is not required to compensate for face detector false alarms or misses, we make sure that the detected bounding box corresponds to a correct detection by comparing each face detector bounding box $b$ with the annotated eye positions $(\bar{\eta}_L, \bar{\eta}_R)$ and only use the detections that match following criteria:

$$\Delta = (b_x, b_y) - \frac{1}{2}(\bar{\eta}_L + \bar{\eta}_R),$$

(16)

$$\Delta_x < \frac{c}{4} \land \Delta_y < \frac{c}{4} \land \|\bar{\eta}_L - \bar{\eta}_R\|_2 < bw,$$

(17)

where $c$ is the eye distance (8), and $\Delta = (\Delta_x, \Delta_y)$ is the offset of the center point in between the eyes to the center of the bounding box.
4.4. Parameter optimization

An important aspect of the evaluation is the choice of parameters. Our method has four main parameters: the eye distance, the LBP window width and height, and the number of scales. To optimize these parameters we select the setting that works best on the training set and use this one on the test sets.

We tried three eye distances: 32, 64, and 128 pixels, three window sizes: $11 \times 11$, $17 \times 17$, and $21 \times 21$ pixels, and three, four or five scales. We found that the best setting is the eye distance of 64 pixels, the window size of $21 \times 21$ pixels and only three scales. The scales are octaves, so the eye distance of 64 pixels with three scales gives eye distances 64, 32, and 16 pixels for $k = 0$, $k = 1$, and $k = 2$ respectively. There is a wide range of parameters that give similar results as changes in these parameters lead to a gradual performance degradation. The most influential parameter, both on performance and complexity is the window size.

Table 1: Comparison of eye localization methods on the BioID image set for different $\epsilon_{max}$ (2) error limits.

<table>
<thead>
<tr>
<th>Method</th>
<th>$&lt; 5%$</th>
<th>$&lt; 7%$</th>
<th>$&lt; 10%$</th>
<th>$&lt; 25%$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section 3.2</td>
<td>91.6%</td>
<td>95.4%</td>
<td>97.5%</td>
<td>99.8%</td>
</tr>
<tr>
<td>[3]</td>
<td>37%</td>
<td>67%</td>
<td>85%</td>
<td>98%</td>
</tr>
<tr>
<td>[5]</td>
<td>79%</td>
<td>85.2%</td>
<td>96.1%</td>
<td></td>
</tr>
<tr>
<td>[6]</td>
<td>89.7%</td>
<td>95.7%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[11]</td>
<td>93.8%</td>
<td>96.4%</td>
<td>98.8%</td>
<td></td>
</tr>
<tr>
<td>[12]</td>
<td>93.0%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[16]</td>
<td>91.8%</td>
<td>98.1%</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2: Comparison of eye localization methods on the FRGC image set for different $\epsilon_{RMS}$ (3) error limits.

<table>
<thead>
<tr>
<th>Method</th>
<th>$&lt; 5%$</th>
<th>$&lt; 10%$</th>
<th>$&lt; 25%$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section 3.4</td>
<td>84.1%</td>
<td>98.1%</td>
<td>99.6%</td>
</tr>
<tr>
<td>[18] A</td>
<td>92%</td>
<td>99%</td>
<td>100%</td>
</tr>
<tr>
<td>[18] B</td>
<td>44%</td>
<td>90%</td>
<td>100%</td>
</tr>
<tr>
<td>[5]</td>
<td>92.8%</td>
<td>97.1%</td>
<td></td>
</tr>
</tbody>
</table>

4.5. Performance of the method

We ran our method on webcam-generated images, movie frames and the BioID test set. Figure 6 shows some examples of each of the test sets. For a direct comparison with other methods, we have performed quantitative evaluations on BioID (Table 1) and FRGC [13] (Table 2).

Table 1 shows that our method substantially outperforms all other methods. The biggest improvement is obtained for tight error limits. This is important because the studies in [14][15][18] show that accurate eye localization is crucial for an acceptable face recognition performance. In particular, for an RMS eye localization error $\epsilon_{RMS}$ of 5% the probability of misclassification already increases from 3% to 28% for LDA and 5% to 17% for EBGM [14]. This also makes the results obtained for the 10% and 25% $\epsilon_{max}$ error limit frequently used for testing (Table 1) practically less relevant.

For an error (2) limit of $\epsilon_{max} = 2\%$, we still have a 40.2% performance. For $\epsilon_{max} = 5\%$ we are at 91.6% and for $\epsilon_{max} = 7\%$, we achieve 95.4%. With an average eye distance of 54.9 pixels, a 2% error is on average 1.10 pixel, which is close to pixel accuracy.

The FRGC dataset [13] (Table 2) is not of direct interest to us since it contains high resolution digital camera images. Our results are however close to state-of-the-art without optimizing parameters for the additional image quality.

4.6. Evaluation of the method complexity

All the results in the evaluation are based on a C++ implementation of the algorithm. On an AMD Athlon 64 at 2.2 GHz we need on average 0.50 s to process an image. This high number has to be considered as an upper bound on the complexity for many reasons. First, new low-end computers are already faster than our machine. Second we did not try to optimize the code and did not exploit trivial parallelism in calculating (13) which could be easily used by modern multicore machines. Third, we work with a large search region $(w_S, h_S)$, and also scan it completely. The complexity can be significantly reduced by limiting the search region and thresholding on the lower scales. We also noticed that the response images $L(x, y)$ are always smooth. This allows for scanning on a coarse grid first, or even for applying a
particle filter to avoid exhaustive search in $L(x, y)$.

If we adopt the processing time per image as a complexity indicator, we can show the effect of different parameters on computational complexity. Starting from the optimal point as explained in section 4.4, increasing the number of scales to four adds only 7% to the complexity. This is because the first scale ($k = 0$) already accounts for 75% of the complexity. Reducing the LBP window size from 21 × 21 to 17 × 17 pixels saves 18%. Reducing the window size from 21 × 21 to 11 × 11 pixels saves 74%. In this case, we only need 97 ms per image and still achieve a 31% result for $P(\epsilon_{\text{max}} < 2\%)$ and a 85% result for $P(\epsilon_{\text{max}} < 5\%)$.

The memory consumption of our implementation is mainly in the histograms and response images and is about 1 MB when using single-precision floating point operations.

5. Discussion

In this paper we described a method for eye localization based on multiscale local binary patterns. The features we used, namely multiscale LBPs, have a compact representation, are simple to compute and provide a good description of the spatial texture repartition. Our primary contribution is our model which is able to achieve both high spatial accuracy and robustness by using spatial multiscale LBP histograms on the entire eye region. It shows to be particularly robust for noise typical of targeted content, while being computationally efficient. In the experimental part we showed that the method outperforms the state-of-the-art on the standard dataset BioID. We also showed that the method works well on our own collection of movie and webcam videos.

We emphasize that the reported performance of our method was obtained by training the LBP model based on the YaleB dataset in order to allow for reproduction. This dataset is however not necessarily the optimal training set for our method. This provides room for further improvement of the performance. Furthermore, the performance can also be improved by incorporating the additional attributes supplied by a multiview face detector to achieve better estimates of the prior eye positions.

References


