Signal Processing 90 (2010) 1290–1299

Contents lists available at ScienceDirect

Signal Processing

journal homepage: www.elsevier.com/locate/sigpro

A novel method for gaze tracking by local pattern model and support vector regressor

Hu-Chuan Lu [a,*], Guo-Liang Fang [a], Chao Wang [a], Yen-Wei Chen [a,b]

[a] Department of Electronic Engineering, Dalian University of Technology, Dalian, China
[b] College of Information Science and Engineering, Ritsumeikan University, Kusatsu, Japan

ARTICLE INFO

Article history:
Received 19 October 2008
Received in revised form
19 October 2009
Accepted 19 October 2009
Available online 27 October 2009

Keywords:
Gaze tracking
PPBTF
LPM
SVR

ABSTRACT

This paper presents a novel eye gaze tracking method with allowable head movement based on a local pattern model (LPM) and support vector regressor (SVR). The LPM, a combination of improved pixel-pattern-based texture feature (PPBTF) and local-binary-pattern texture feature (LBP), is employed to calculate texture features from the characteristics of the eyes and a new binocular vision scheme is adopted to detect the spatial coordinates of the eyes. The texture features from LPM and the spatial coordinates together are fed into support vector regressor (SVR) to match a gaze mapping function, and subsequently to track gaze direction under allowable head movement. The experimental results show that the proposed approach results in better accuracy in estimating the gaze direction than the state-of-the-art pupil center corneal reflection (PCCR) method.

1. Introduction

Eye gaze, referring to the direction of line of sight, reveals a person's focus of attention and interest. The majority of existing gaze tracking techniques are vision based, i.e., cameras are used to capture images of the eyes. Some of these camera-based techniques are intrusive since special equipments such as chin rests, electrodes [25], and head-mounted cameras [26] are required on users. The scheme proposed in this paper is non-intrusive, that is, users are not equipped with any devices.

Yu and Eizenman [1] developed a head-mounted methodology by using features extracted from a video sequence to determine the position of a head relative to objects in a scene. Since the relative eye positions to the position of the head are also provided, the data of eye and head positions can be integrated to determine the fixation behavior of objects in the scene. An appearance-based eye gaze estimation scheme was developed by Tan et al. [2] and an artificial neural network (ANN)-based gaze direction tracking approach was proposed in [3]. Zhu et al. [4] used pupil center corneal reflection (PCCR) to determine the gaze track direction and geometric relationships to compensate for the effects of head movement. Although all these schemes described above have made significant contributions to the eye gaze tracking technology, their accuracy and reliability need to be further improved.

In this paper, we propose a new method for eye gaze tracking under allowable head movement of ± 5 cm forward/backward, ± 2 cm on horizon, and ± 2 cm along vertical direction. This method first calculates the spatial coordinates of the eyes and the LPM texture features, which will then be fed into support vector regressor (SVR) [19,20,27] to predict the gaze direction. The adopted binocular vision method has less complexity in terms of camera calibration process and lower cost than traditional methods, without regard to the evaluation of accuracy. In order to calculate target coordinates accurately, the binocular vision method only requires the spatial coordinates of three points, while the camera calibration

* Corresponding author.
E-mail address: lhchuan@dlut.edu.cn (H.-C. Lu).

0165-1684/$ - see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.sigpro.2009.10.014

methods [22–24] require multiple views of a checkerboard to estimate the inner and outer parameters of the camera.

Oftentimes, traditional infrared gaze tracking methods calculate the pupil-glint vector using basic image processing algorithms [6]. However, the traditional approaches suffer from the fact that pupil fuzzy borders and shape changes may cause the pupil center to drift. In addition, the reflection point sometimes can be large enough to offset the actual position. Factors such as relatively large size of reflection point, pupil fuzzy borders and shape changes, may cause an inaccurate calculation of pupil-glint vector, and hence adversely affect experimental results. The two-dimensional PCCR vector has the advantages of simple calculation and short operation time, but it neglects the global characteristics of the eyes when a person frequently changes the direction of sight. Our study indicates that the LPM features of the captured eye images contain both the pupil-glint vector information and texture changes information. Therefore, there is no need to calculate the pupil-glint vector separately because the proposed method allows us to estimate the gaze direction more accurately by making use of the global features.

The rest of the paper is organized as follows. Section 2 introduces the adopted binocular vision method. Section 3 describes LPM, original and improved PPBTF algorithms. Section 4 presents the experimental results, and Section 5 concludes the paper with future work.

2. Binocular vision method

The main purpose of the binocular vision method is to determinate the spatial coordinates of the eyes. Before calculating the coordinates of the eyes, the location of eyes in the two camera images should be obtained (i.e., eye detection).

In this paper, the eye detection is carried out by using the method proposed in [15], which is the evolvement of a face detection method proposed by Viola and Jones [7]. Lu and Zhang [15] used some geometric characteristics of the eyes to reduce the rate of false detection. The left eye's center coordinates in image are used as the input parameters to calculate spatial coordinates of the left eye. Assuming that there is no distortion of camera, the spatial coordinates of the eye can be calculated by the following method.

The optical axes alignment is carried out after setting up two cameras. The coordinates of two cameras can be measured, respectively, which are defined as $C_1 = (x_1, y_1, z_1)$ and $C_2 = (x_2, y_2, z_2)$. There is a reference point $O = (x_0, y_0, z_0)$ which can be any spatial point in the two cameras' field of vision. As shown in Fig. 1, α is the horizontal angle of C_1 and O, β is the horizontal angle of C_2 and O, γ is the vertical angle of C_1 and O, and θ is the vertical angle of C_2 and O. C_1', C_2' and O' are projections of C_1, C_2 and O in the $x-z$ plane. Point B is the target point, whose coordinates should be determined. B' is the projection of B in the $x-z$ plane.

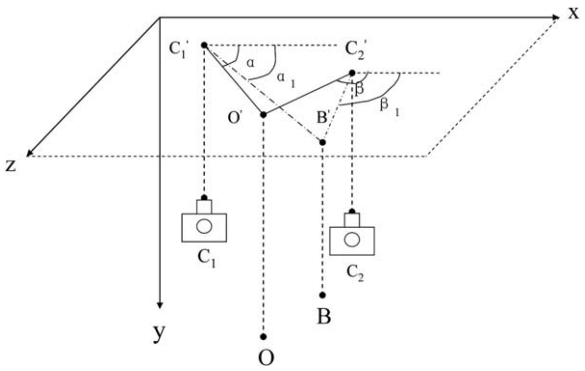

Fig. 1. Binocular vision method for eye's spatial coordinates x–z plane diagram.

The relationships between above-mentioned parameters can be expressed as

$$\alpha = \arctan((z_0 - z_1)/(x_0 - x_1)) \quad (1)$$

$$\beta = \arctan((z_0 - z_2)/(x_0 - x_2)) \quad (2)$$

$$\gamma = \arctan((y_0 - y_1)/(z_0 - z_1)) \quad (3)$$

$$\theta = \arctan((y_0 - y_2)/(z_0 - z_2)) \quad (4)$$

Let us assume that point B is the center of the left eye. The spatial coordinates of point B can be calculated as follows.

Before calculating the coordinates of point B, we should measure the horizontal and vertical angles of two cameras C_1 and C_2 (the cameras' foci are fixed). These two are inner parameters, and should be measured before setting up the cameras. There is a simple method to obtain the horizontal and vertical angles of the cameras. To begin with, lay a checkerboard on the ground level. Then, adjust the position of the camera to ensure that a defined rectangular region, consisting of several connected grids on the checkerboard, cover neither more nor less than the sight of the camera. At this moment, the camera is perpendicular to the checkerboard and the perpendicular foot lies at the center of the rectangular region, as shown in Fig. 2. The horizontal and vertical visual angles can be determined as $\varepsilon_1 = \arctan(W_1/h) \times 2$ and $\mu_1 = \arctan(H_1/h) \times 2$, respectively, where h is measured by plumb line. Parameters ε_2 and μ_2 of C_2 can be obtained in a similar way.

The horizontal and vertical angles of the vector $\overline{BC}_1$ are α_1 and γ_1, respectively. The horizontal angle of the vector $\overline{BC_2}$ is β_1, and the vertical angle is θ_1 (Fig. 1). The coordinates of point O in the image captured by camera C_1 are $O_1 = (x_{o1}, y_{o1})$. The coordinates of B in the image captured by C_1 are $B_1 = (x_{b1}, y_{b1})$. Define $\Delta x_1 = x_{o1} - x_{b1}$, and $\Delta y_1 = y_{o1} - y_{b1}$. Similarly, the coordinates of point O in the image captured by C_2 are $O_2 = (x_{o2}, y_{o2})$, and the coordinates of point B are $B_2 = (x_{b2}, y_{b2})$. Define $\Delta x_2 = x_{o2} - x_{b2}$, and $\Delta y_2 = y_{o2} - y_{b2}$. If the size of the image captured by C_1 is $w_1 \times h_1$, and the image captured by C_2 is $w_2 \times h_2$, then α_1, γ_1, β_1 and θ_1 can be expressed as

$$\alpha_1 = \alpha - \arctan\frac{\Delta x_1 \tan 0.5\varepsilon_1}{0.5 w_1} \quad (5)$$

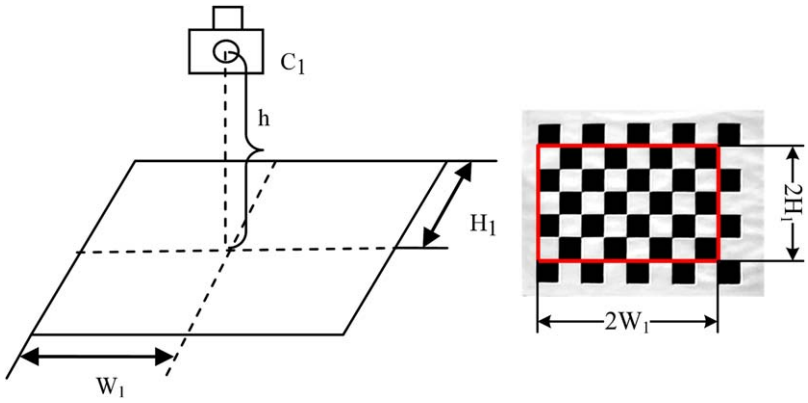

Fig. 2. Camera angle measurement diagram.

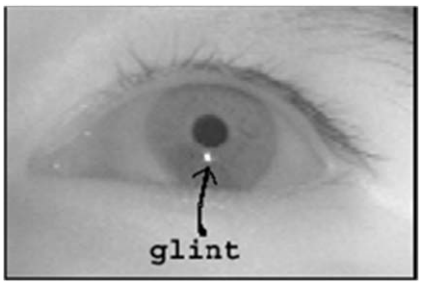

Fig. 3. Eye image with glint.

$$\gamma_1 = \gamma - \arctan\frac{\Delta y_1 \tan 0.5\mu_1}{0.5h_1} \tag{6}$$

$$\beta_1 = \beta - \arctan\frac{\Delta x_2 \tan 0.5\varepsilon_2}{0.5w_2} \tag{7}$$

$$\theta_1 = \theta - \arctan\frac{\Delta y_2 \tan 0.5\mu_2}{0.5h_2} \tag{8}$$

The formulas to determine the coordinates of point B are as follows:

$$x = \frac{(x_1 \tan\alpha_1 - x_2 \tan\beta_1 + z_2 - z_1)}{\tan\alpha_1 - \tan\beta_1} \tag{9}$$

$$z = \frac{(x_1 - x_2)\tan\alpha_1 \tan\beta_1 + z_2 \tan\alpha_1 - z_1 \tan\beta_1}{\tan\alpha_1 - \tan\beta_1} \tag{10}$$

$$y = \tfrac{1}{2}[y_1 + (z - z_1)\tan\gamma_1 + y_2 + (z - z_2)\tan\theta_1] \tag{11}$$

It is critical that the images captured from the two cameras must be synchronous; otherwise, computational errors will increase because of the discrepancy. For this purpose, our application uses the cvcam class of OpenCV [9], which can obtain a pair of images simultaneously through its callback function.

3. Features of eyes

As shown in Fig. 3, corneal reflection (CR) can be easily observed in the dark pupil effect eye image. The center of the pupil and the CR defines a vector in the image. This vector can be mapped to screen coordinates on a computer monitor after a calibration procedure. In general, during the calibration procedure, the user should look at several different points on the computer screen, one point at a time, and presses a button to capture images. In Morimoto et al. [8], the authors used nine points to calibrate the parameters of a second order polynomial function. Duchowski [21] included more details about corneal reflection tracker and the mapping screen coordinates to the 2D image.

A major assumption in the classical algorithms [16–18] is that the user cannot change the position of the head, but this is not quite feasible in real world applications. In order to accommodate head movement, researchers have put forward various methods. For example, Zhu and Ji [4] took advantage of simple geometrical relationships. Later, these authors combined the spatial coordinates of the eye and pupil-glint vector to achieve the tracking process [10]. In [11], Zhu and Ji estimated the 3D gaze direction of the users. Methods proposed in [4,10,11] are all based on PCCR. The accuracy of calculating the vector in methods proposed by Zhu and Ji is negatively affected by the factors such as pupil fuzzy borders and shape changes. These factors cause the pupil center migration, thereby reducing the accuracy of calculation. Moreover, the reflection point sometimes can become large enough to offset its actual position. This factor may also cause inaccurate calculation of the pupil-glint vector and therefore affects experimental results.

The local pattern model (LPM) features of the eye images contain the "pupil-glint" vector information by obtaining the texture information. Taking into account this fact, calculation of the pupil-glint vector becomes redundant, while LPM improves accuracy of estimation the gaze under allowable head movement.

3.1. Improved PPBTF algorithm

PPBTF is a type of texture feature extraction algorithm, which was first proposed in [5]. This algorithm can be effectively applied to expression recognition [12], and is briefly described as follows.

3.1.1. PPBTF

PPBTF is constructed from one pattern map. A gray scale image is transformed into a pattern map in which edge and background pixels are classified by pattern matching with a given set of M pattern templates $\{W_i\}$ that reflects the spatial features of images. For each pixel (x, y) in a gray scale image $\mathbf{I}$, let z_i be the inner product of its $S \times S$ neighbor block $\mathbf{b}$ with the ith pattern templates

$$z_i = |b \cdot W_i| \tag{12}$$

The pixel (x, y) in the pattern map $\mathbf{P}$ is assigned a number k such that $z_k = \max(z_1, z_2, \ldots, z_M)$. Therefore, the pixel values in a pattern map represent the pattern classes of pixels in the original gray scale image.

The feature model comes as follows. Let us assume that the number of patterns is M, and then the pixel value $\mathbf{P}(x, y)$ in the pattern map $\mathbf{P}$ is in a range of $[1, M]$. For each pixel (i, j), the features in a window $S1 \times S1$ can be generated by

$$f_l(i,j) = \sum_{x=i-(S1-1)/2}^{i+(S1-1)/2} \sum_{y=j-(S1-1)/2}^{j+(S1-1)/2} h_l(x,y), l = 1, \ldots, M \tag{13}$$

where h is a binary function defined as

$$h_l(x,y) = \begin{cases} 1 & \text{if } P(x,y) = l \\ 0 & \text{otherwise} \end{cases} \tag{14}$$

Thus, the feature f_l gives the number of the pixels belonging to the lth pattern, and the feature vector can be constructed with f_l as components.

$$F_{PPBTF} = (f_1, f_2, \ldots, f_M) \tag{15}$$

3.1.2. Principal component analysis (PCA) templates

Pattern templates represent the spatial features in an image and reflect how the value of each pixel relates to its neighbors. The process of obtaining the PCA templates is as follows:

- Denote an $S \times S$ image block as a neighbor vector.
- Randomly create N $S \times S$ image blocks on each given image.
- Suppose there are K images, then matrix A with the size of $(S \times S) \times (K \times N)$ is composed of all neighbor vectors.
- Calculate PCA using matrix A.
- Select some eigenvectors from the PCA analysis as the PCA templates.

For instance, let us assume the block-size is 5×5. We randomly generate on each eye image 100 block samples by the size of 5×5. Use 811 eye images of the size 64×120 as template training samples. Analyze a matrix of $25 \times 81\,100$ using PCA. Sort the basis functions in decreasing order of their eigenvalues and select the first 25 basis functions. The first basis function, corresponding to the largest eigenvalue is a Gaussian low-pass filter, and the others are derivative filters. With the exception of the first basis function, the rest can be used as gradient filters for pattern matching. In this paper, excluding the first basis function, the subsequent eight basis functions are chosen as pattern templates.

3.1.3. Improved PPBTF features

Like Gabor features, the number of original PPBTF features is larger than the number of pixels. To be specific, it generates eight features for every pixel under the condition of eight existing PCA templates. The number of features is too large to fit for gaze tracking. This calls for methods to reduce the dimension. We employ a method as follows. The pattern map of an image is divided into a few regions of equal size to form a histogram directly rather than calculate the feature for every pixel. Then the numbers of all patterns are counted for each region. It is easy to see that using this method the number of extracted features is reduced from $8 \times m$, where m is the size of the image, to $8 \times N$, where N is the number of regions in an image.

However, there are two drawbacks of this method:

- The dimension of the features is too small.
- For every pixel of the pattern map, the quantitative level is 8, which is not enough to represent the texture pattern.

For the first one, we can overcome it by increasing the number of regions and refining the divisions. Therefore, the essence of improved PPBTF is an application of higher quantitative level to represent the value in a pattern map. Specifically, it can be described as follows:

- In the pattern map, we define max_index=max $(z_1, z_2, \ldots, z_M)$ and min_index=min $(z_1, z_2, \ldots, z_M)$. Therefore, there are $P_8^2 = 56$ pattern permutations for different max_index and min_index. If we assign each permutation a number ranging from one to 56, the quantitative level will increase from eight to 56, and this is the reason for changing the original pattern map, while the number of PCA templates is still eight (Fig. 4).
- This improvement is effective, because, although max_index represents the major pattern of the region around the pixel, min_index also represents the major pattern of the region in the opposite direction. Hence, the combination pattern of max_index and min_index outperforms the single max_index pattern in a pattern map.
- The same as $8 \times N$, the number of extracted features is $56 \times N$, where N is the number of regions.

The top left corner of Fig. 4 represents the original pattern map with its inner data matrix underneath. From the matrix, we can see that many blocks have the same value. For instance, numbers in the left red rectangle are of the same value 2, which means pattern 2 is the major pattern of the neighbor region around the pixels.

The top right corner of Fig. 4 is the improved pattern map with its inner data matrix underneath. The blocks that have the same value in original pattern map now may have different values in the improved pattern map.

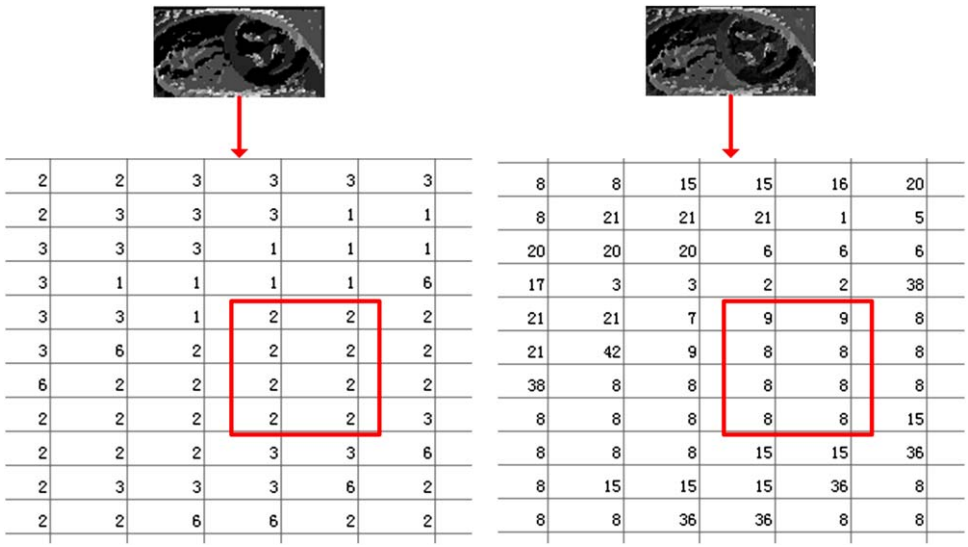

Fig. 4. The original pattern map and the improved pattern map.

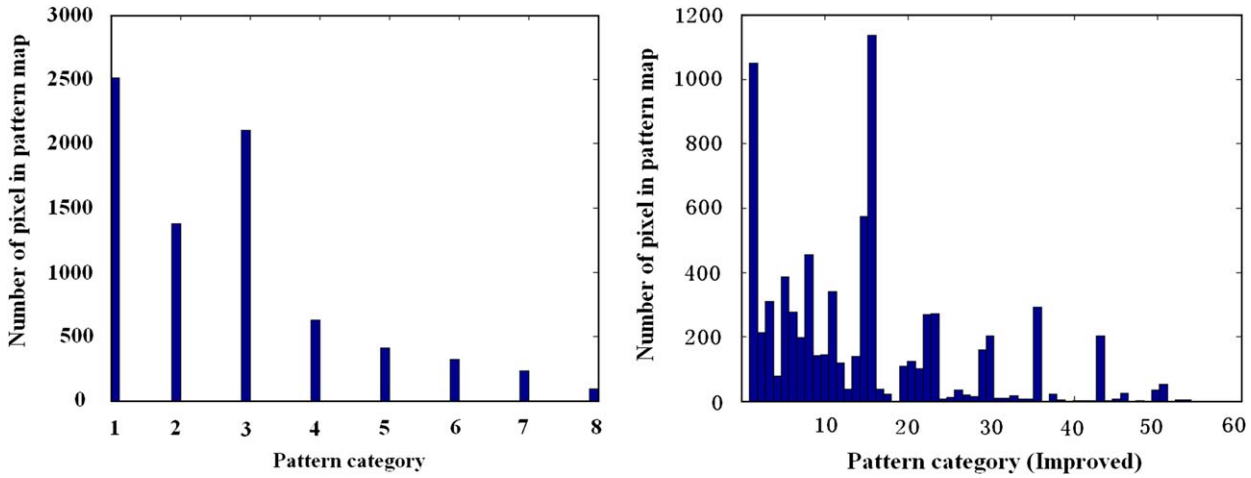

Fig. 5. The dimensional histogram statistic of the original pattern map and the improved pattern map.

For example, in the right red rectangle, that data have changed from 8 to 9 confirms our hypothesis. Fig. 5 also shows that the values of the improved pattern map have been dispersed from eight to 56 bins.

Experiment in Section 4 shows that the fitting error of SVR is greatly reduced due to the improvement. On the other hand, the improvement does not increase the computing time significantly.

3.2. Local pattern model

The local pattern model comes from the feature fusion strategy proposed in [13]. Given two different kinds of features, there are two strategies to integrate them into a single feature. One strategy is called the serial strategy, which combines two kinds of features in serial mode, thus the dimension of the new features is the sum of the two. The other strategy is called the parallel strategy, which combines two kinds of features into a complex vector. Obviously, the dimension stays constant. Yang and Yang [13] showed that the parallel strategy usually outperforms the serial strategy.

The algorithm for the local pattern model is described as follows:

- Given one gray image of the captured eyes, use the improved PPBTF algorithm to calculate the features based on six equal blocks. This operation generates $6 \times 56=336$ features for the image.
- Given the same gray image, use uniform LBP algorithm [14] to calculate the features in the same division. The number of generated features is $6 \times 59=354$.
- Suppose α is the improved PPBTF features of the image and β is the uniform LBP features of the image. Add three zero numbers to the α of each block, because it is three digits shorter than β.

- The local pattern model of an image is $(\alpha + i\beta)$. The sequence of α and β should be meticulous. Experimental results in Section 4 show that the fitting effect of α are better than the fitting effect of β.
- Results in Section 4 show that the complex features generated by local pattern model are well applicable for SVR. The advanced version of SVR was named as generalized support vector regression (GSVR). GSVR is a reasonably extension of SVR in complex-number fields.

4. Experimental results

4.1. Experimental setup

Our system consists of two CCD cameras, as shown in Fig. 6. To reduce the errors in z plane, camera C_1 should be close to camera C_2, while the horizontal axis of the two cameras should be parallel to the ground. The spatial coordinates of the eye (x, y, and z) can be calculated by using the binocular vision method proposed in Section 2. The 640×480 images captured by C_2 are used to detect the raw eyes (Fig. 7) by using the eye detection method proposed in [7]. All detected eye images are normalized to 64×120. Then, the LPM algorithm is used to extract features from the eye images.

Finally, the LPM features combined with the spatial coordinates are fed into SVR [19,20,27], which is a generalized support vector regressor (GSVR). GSVR uses complex vectors as its input parameters. In our experiments, because of high nonlinearity of the gaze mapping function we use the kernel of Gaussian radial basis function (RBF) for SVR. The flow chart of the gaze-tracking algorithm is shown in Fig. 8.

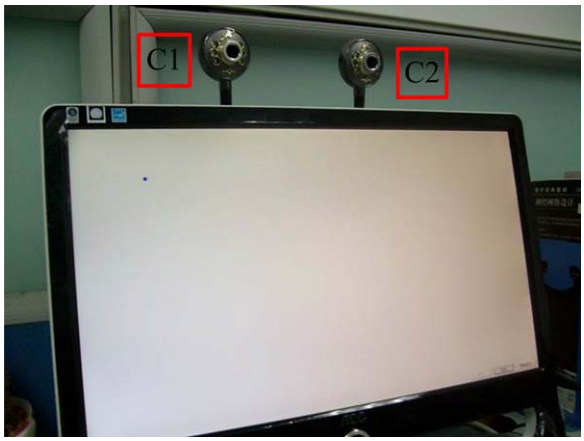

Fig. 6. Cameras C1, C2 are used to measure the spatial coordinates of the observer's eyes; the image captured by C2 is also used to compute the LPM features. The screen dimensions are 1024×786.

In the training procedure, we asked five users to gaze at a blue point moving on the screen over nine different positions (Fig. 9). Each picture was taken by C_1 and C_2 simultaneously. During the task, each camera took 1014 pictures. From 1014 raw eyes (left eye), detected from the pictures, 811 raw eyes were used to build up the training set, while the other 203 for testing. In fact, two SVRs were chosen for training and testing in horizontal and vertical directions, respectively. During the training procedure, we tried to simulate allowable head movements, meaning that users could move their heads ± 5 cm forward/backward, ± 2 cm along horizontal direction, and ± 2 cm along vertical direction.

4.2. Parameters analysis

There are three main parameters, which affect the final performance: the block-size, the number of regions on extracting PPBTF features, and the parameter of RBF kernel on SVR.

In order to optimize the block-size, which could lead to the best performance, we assume that all images are divided into 2×3=6 regions, each with a size of 32×40. In our experiment, we find that the results become more and more accurate with the increasing of the block-size. However, when computing PCA templates, with the increasing of block-size, the correlation among different blocks increases too. As a result, the number of eigenvectors could be insufficient to generate eight PCA templates. This indicates that the increase of block-size is confined. We have adopted the foursquare blocks, as presented in Table 1. For foursquare blocks, 15×15 is the largest block-size that we can choose under the restriction to compute the PCA templates. In other words, when the foursquare block-size becomes bigger than 15×15, the PCA templates cannot be obtained. Subsequently, we extend the foursquare blocks to quadrate blocks in order to obtain the largest block-size under the restriction. The results of extension are presented in Table 2. For quadrate blocks, for a similar reason, 11×23 is the largest block-size we can choose to compute the PCA templates.

In the following tables, "horizontal kernel" represents the best RBF kernel parameter of SVR, which is used to predict the position on horizon, similar to "vertical kernel". "Horizontal $(\mu \pm \sigma)$" represents the horizontal average accuracy and the standard deviation of horizontal

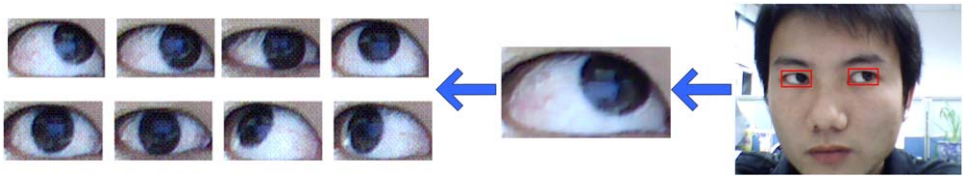

Fig. 7. Some of the raw eye images.

accuracy, similar to "Vertical ($\mu \pm \sigma$)", where μ is the average of the estimation errors, while σ is the standard deviation.

The results in Tables 1 and 2 present that the largest quadrate block 11×23 generates the best performance. These results are generated on the condition that each pattern map of PPBTF and each eye image of LBP are divided into 2×3 regions. Actually, even with other division scheme (for example 2×4, 4×3, 4×4), the best performance still comes from the block-size 11×23.

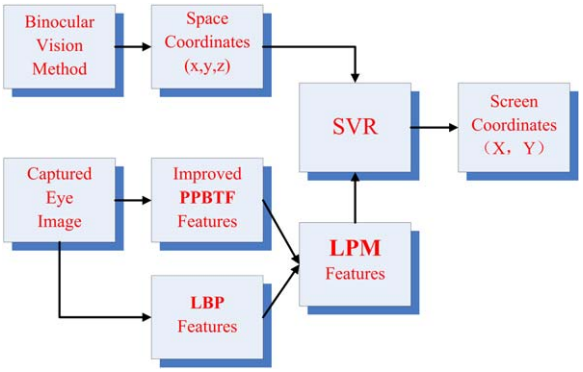

Fig. 8. Flow chart of the gaze-tracking algorithm.

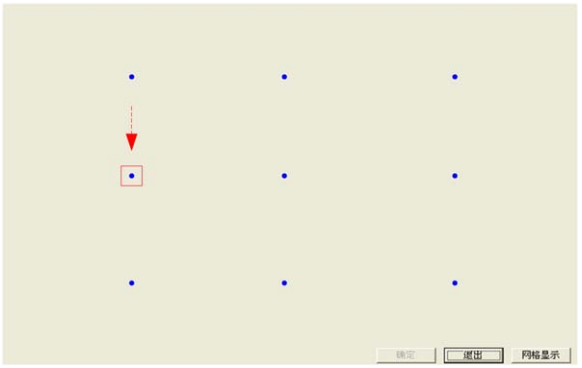

Fig. 9. The subject is asked to look at nine positions on the screen following the blue point. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 3 presents the results based on different division schemes with the same block-size 11×23. In Table 3, division 2×3 means to divide each pattern map of PPBTF and each eye image of LBP into six regions. Therefore, the size of each region is 32×40, and the number of features is 6×59=354. The meaning of other division scheme is similar. The results show that the performance is barely improved with increasing number of regions, while the training procedure becomes time consuming with larger number of features. Considering this, the division 2×3 is optimal to obtain the expected result in our experiments.

In our experiments, the RBF kernel parameter is adjusted to obtain the best average accuracy. The horizontal and vertical kernels in all tables are the kernel parameters, which result in the best average accuracy. Fig. 10 is the plot of adjusted kernel parameters when the condition that the division is 2×3 and the block-size is 11×23.

4.3. Comparative experiments

To verify the advantage of our proposed method, some comparative experiments conducted and results are shown in the following tables.

Table 4 compares four different algorithms based on SVR. First, the division 2×3 and the block-size 11×23 are adopted. The results show that improved PPBTF outperforms PPBTF and LBP with 1.86 ± 1.59 mm in horizontal accuracy and 2.23 ± 1.99 mm in vertical accuracy. As we can see from the table, the LPM features present the best accuracy.

The results from comparative experiments in Table 4 can be concluded as:

- First, the experiment shows that the improvement of PPBTF can be justified by the histogram of block statistics, because it enriches the texture information

Table 1
Tracking results using LPM on foursquare blocks using SVR.

Block size	3×3	7×7	11×11	15×15
Horizontal ($\mu \pm \sigma$)	3.63 ± 3.11 mm	2.82 ± 2.32 mm	1.96 ± 1.83 mm	1.73 ± 1.84 mm
Horizontal kernel	1000	1200	1200	1600
Vertical ($\mu \pm \sigma$)	3.50 ± 2.96 mm	2.67 ± 2.27 mm	2.48 ± 2.10 mm	2.03 ± 1.68 mm
Vertical kernel	700	800	900	1200

Table 2
Tracking results using LPM on quadrate blocks using SVR.

Block size	5×11	7×15	9×19	11×23
Horizontal ($\mu \pm \sigma$)	2.25 ± 2.20 mm	1.90 ± 1.83 mm	1.44 ± 1.39 mm	1.30 ± 1.21 mm
Horizontal kernel	400	700	600	800
Vertical ($\mu \pm \sigma$)	2.29 ± 1.87 mm	2.24 ± 1.79 mm	1.92 ± 1.60 mm	1.50 ± 1.49 mm
Vertical kernel	600	300	700	400

Table 3
Tracking results using LPM on different divisions using SVR.

Division	2 × 3	2 × 4	4 × 3	4 × 4
Features number	354	472	708	944
Horizontal ($\mu \pm \sigma$)	1.30 ± 1.21 mm	1.52 ± 1.31 mm	1.34 ± 1.57 mm	1.61 ± 1.50 mm
Horizontal kernel	800	1000	1700	2300
Vertical ($\mu \pm \sigma$)	1.50 ± 1.49 mm	1.55 ± 1.59 mm	1.60 ± 1.45 mm	1.65 ± 1.63 mm
Vertical kernel	400	700	1200	1400

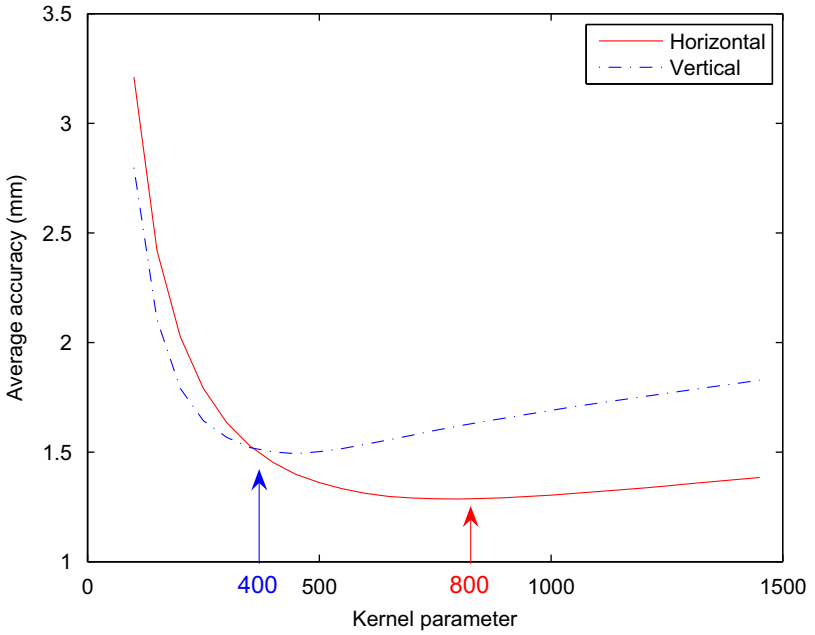

Fig. 10. The plot of adjusted RBF kernel parameters.

Table 4
Tracking results on different algorithms of feature algorithms using SVR.

Algorithm	LBP	PPBTF	Improved PPBTF	LPM
Horizontal ($\mu \pm \sigma$)	6.44 ± 5.28 mm	5.48 ± 4.56 mm	1.86 ± 1.59 mm	1.30 ± 1.21 mm
Horizontal kernel	14 800	13 500	6500	800
Vertical ($\mu \pm \sigma$)	5.49 ± 4.29 mm	4.37 ± 3.48 mm	2.23 ± 1.99 mm	1.50 ± 1.49 mm
Vertical kernel	30 800	23 800	5500	400

Table 5
Comparison with other system.

Algorithm	Horizontal accuracy ($\mu \pm \sigma$)	Vertical accuracy ($\mu \pm \sigma$)
LPM+SVR	1.30 ± 1.21 mm	1.50 ± 1.49 mm
[11]	5.02 ± 2.03 mm	6.40 ± 2.35 mm

represented by the neighbor region around each pixel in the pattern map, which enhances the fitting effect.

- Second, the experimental results prove that the construction of local pattern model is meaningful. LBP features and the improved PPBTF features, although representing different kinds of texture information of the eye image, can be combined to form LPM features in order to achieve high-precision results. We believe that LPM will influence the direction of our further researches.

Table 5 presents that the method based on LPM features with the use of SVR can approximate the gaze mapping function more precisely than the method used by Zhu and Ji [11]. As presented in the table, the average horizontal and vertical errors of LPM+SVR algorithm are approximately 1.30 and 1.50 mm, respectively, while the horizontal and vertical errors of algorithm presented in [11] are 5.02 and 6.40 mm, respectively.

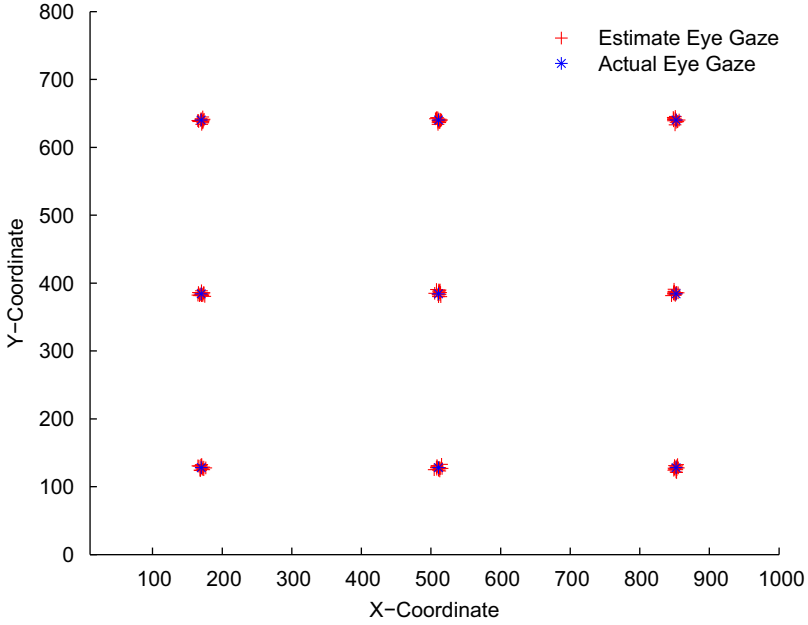

Fig. 11. The plot of the estimated and the actual gaze points, where "+" represents the estimated gaze point and "*" represents the actual gaze point.

Fig. 11 illustrates errors between the estimated and actual gaze points. The average horizontal error in the screen is around 1.30 mm. The average vertical error is around 1.50 mm.

5. Conclusion

This paper presents a novel eye gaze-tracking scheme based on local pattern model and support vector regressor. The binocular vision method is adopted to calculate the spatial coordinates of the eyes and the LPM algorithm is utilized to describe the features of the captured eyes. With the combination of the spatial coordinates and LPM features as the input to SVR, the mapping function of gaze direction and screen coordinates can be predicted. The experimental results demonstrate the effectiveness of the proposed eye gaze tracking approach when compared with the state-of-the-art schemes. As part of future work, the proposed scheme will be extended to research such as increasing the number of estimated points and the range of the allowable head movement, and will be applied to human–computer interaction.

References

[1] L.H. Yu, M. Eizenman, A new methodology for determining point-of-gaze in head-mounted eye tracking systems, IEEE Transactions on Biomedical Engineering 51 (10) (2004) 1765–1773.
[2] K. Tan, D. Kriegman, H. Ahuja, Appearance based eye gaze estimation, in: Proceedings of the IEEE Workshop on Applications of Computer Vision—WACV02, 2002, pp. 191–195.
[3] S. Baluja, D. Pomerleau, Non-intrusive gaze tracking using artificial neural networks, Advances in Neural Information Processing Systems 6 (1994) 753–760.
[4] Z. Zhu, Q. Ji, Eye gaze tracking under natural head movements, in: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), San Diego, CA, June 2005, pp. 918–923.
[5] X.Y. Zeng, Y.W. Chen, Z. Nakao, H. Lu, Texture representations based on pattern map, Signal Processing 84 (3) (2004) 589–599.
[6] J.-G. Wang, E. Sung, R. Venkateswarlu, Eye gaze estimation from a single image of one eye, IEEE International Conference on Computer Vision 45 (2003) 136–143.
[7] P. Viola, M.J. Jones, Robust real-time face detection, International Journal of Computer 57 (2) (2004) 137–154.
[8] C. Morimoto, D. Koons, A. Amir, M. Flickner, Pupil detection and tracking using multiple light sources, Image and Vision Computing 18 (4) (2000) 331–336.
[9] 〈http://www.opencv.org.cn/index.php/〉.
[10] Z. Zhu, Q. Ji, Nonlinear eye gaze mapping function estimation via support vector regression, in: The 18th International Conference on Pattern Recognition (ICPR), 2006, pp. 1132–1135.
[11] Z. Zhu, Q. Ji, Novel eye gaze tracking techniques under natural head movement, IEEE Transactions on Biomedical Engineering 54 (12) (2007) 2246–2260.
[12] H. Lu, Y. Huang, Y.-W. Chen, D. Yang, Real-time facial expression recognition based on pixel-pattern-based texture feature, Electronic Letters 43 (17) (2007) 916–918.
[13] J. Yang, J.-Y. Yang, Generalized K–L transform based combined feature extraction, Pattern Recognition 35 (1) (2002) 295–297.
[14] T. Ojala, M. Pietikinen, T. Menp, Multiresolution grayscale and rotation invariant texture classification with local binary patterns, IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (7) (2002) 971–987.
[15] H. Lu, W. Zhang, Eye detection based on rectangle features and pixel-pattern-based texture features, in: International Symposium on Intelligent Signal Processing and Communication System, 2007, pp. 265–268.
[16] T.E. Hutchinson, K.P. White Jr., K.C. Reichert, L.A. Frey, Human–computer interaction using eye-gaze input, IEEE Transactions on Systems, Man, and Cybernetics (1989) 1527–1533.
[17] R.J.K. Jacob, Eye-movement-based human–computer interaction techniques: towards non-command interfaces, Ablex Publishing Corporation, Norwood, NJ, 1993, pp. 151–190.
[18] C.H. Morimoto, D. Koons, A. Amir, M. Flickner, Frame-rate pupil detector and gaze tracker, in: IEEE ICCV'99 Frame-rate Workshop, 1999.
[19] A.J. Smola, B. Scholkopf, A tutorial on support vector regression, NeuroCOLT Technical Report NC-TR-98-030, Royal Holloway College, University of London, UK, 1998.

[20] V.N. Vapnik, The Nature of Statistical Learning Theory, Springer, New York, 1995.
[21] A.T. Duchowski, Eye Tracking Methodology: Theory and Practice, Springer, Berlin, 2003.
[22] J. Heikkilä, O. Silvén, A four-step camera calibration procedure with implicit image correction, in: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'97), 1997, pp. 1106–1112.
[23] Z. Zhang, A flexible new technique for camera calibration, IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (2000) 1330–1334.
[24] Z. Zhang, Flexible camera calibration by viewing a plane from unknown orientations, in: ICCV, 1999, pp. 666–673.
[25] M.J. Coughlin, T.R. Cutmore, T.J. Hine, Automated eye tracking system calibration using artificial neural networks, Computer Methods and Programs in Biomedicine 76 (2004) 207–220.
[26] L.E. Hong, M.T. Avila, I. Wonodi, R.P. McMahon, G.K. Thaker, Reliability of a portable head-mounted eye tracking instrument for schizophrenia research, Behavioural Research Methods 37 (2005) 133–138.
[27] N. Kabaoglu, H.A. Clrpan, Wideband target tracking by using SVR-based sequential Monte Carlo method, Signal Processing 83 (11) (2008) 2804–2816.