Hindawi
Mobile Information Systems
Volume 2019, Article ID 4568929, 14 pages
https://doi.org/10.1155/2019/4568929

Research Article

An Efficient and Robust Iris Segmentation Algorithm Using Deep Learning

Yung-Hui Li, Po-Jen Huang, and Yun Juan

Department of Computer Science and Information Engineering, National Central University, Taoyuan 32001, Taiwan

Correspondence should be addressed to Yun Juan; nh60211@g.ncu.edu.tw

Received 28 August 2018; Revised 15 November 2018; Accepted 22 November 2018; Published 2 January 2019

Academic Editor: Marco Anisetti

Iris segmentation is a critical step in the entire iris recognition procedure. Most of the state-of-the-art iris segmentation algorithms are based on edge information. However, a large number of noisy edge points detected by a normal edge-based detector in an image with specular reflection or other obstacles will mislead the pupillary boundary and limbus boundary localization. In this paper, we present a combination method of learning-based and edge-based algorithms for iris segmentation. A well-designed Faster R-CNN with only six layers is built to locate and classify the eye. With the bounding box found by Faster R-CNN, the pupillary region is located using a Gaussian mixture model. Then, the circular boundary of the pupillary region is fit according to five key boundary points. A boundary point selection algorithm is used to find the boundary points of the limbus, and the circular boundary of the limbus is constructed using these boundary points. Experimental results showed that the proposed iris segmentation method achieved 95.49% accuracy on the challenging CASIA-Iris-Thousand database.

1. Introduction

In the 21st century, people use electronics (personal computers, laptops, smartphones, smart watches, etc.) to browse through web-based social platforms, store personal images or videos, chat with other people through text or video, and so on. The amount of personal information stored in electronics is increasing by the day. Thus, biometric authentication is required to prevent unauthorized users from stealing such information from personal electronics. Biometric authentication is also used in the access control systems to identify illegal persons and block them from entering private buildings [1].

Among all the biometric modalities, iris recognition is the one with the highest performance, in terms of false acceptance rate (FAR) and false rejection rate (FRR) [2, 3]. Iris as a biometric identification method has a large amount of the complex texture information available for identification. This paper focuses on an iris recognition system that uses the iris texture for biometric identification.

A common iris recognition system consists of six elementary steps: iris image acquisition, image preprocessing, iris boundary segmentation, iris image normalization, feature extraction, and feature matching [4, 5]. The iris boundary segmentation step is a critical step in the entire iris recognition system. In an iris image, most of the iris textures are concentrated in the iris region close to the pupillary boundary. If the boundary of the pupillary region is not accurately located, a large number of iris textures will be missed in the feature extraction step. In most cases, the limbus boundary is obscured by eyelashes, eyelids, and specular reflections, and thus, a number of noisy features will be extracted in the feature extraction step, if the limbus boundary is not accurately located in the iris segmentation step. These features will deteriorate the performance of the entire iris recognition system [5].

In this paper, we present a novel algorithm for iris boundary segmentation. The proposed algorithm breaks down the iris segmentation step into two actions: locating the eye and segmenting the iris region. Judging whether or not the target exists in the image and locating the target are two major challenges in the object detection technology. Firstly, a well-designed Faster R-CNN network model [6] is used to detect and locate eyes in the proposed algorithm.

Once the potential bounding boxes of the eye are obtained, a pretrained Gaussian mixture model (GMM) [7, 8] is used to fit the pupillary region. Secondly, an improved limbus boundary localization algorithm [9] is applied to find the limbus boundary points. Thirdly, the iris region is located by identifying the pupillary and limbus boundaries. Fourthly, we evaluate the accuracy of our algorithm with a newly proposed evaluation method in Section 4.3. Finally, we conclude our research by discussing result and the possibility of implementing the method to a mobile device.

2. Literature Review

2.1. Background of Object Detection. Object detection is a task of finding different objects in an image and classifying them. In 2014, Girshick et al. [10] showed dramatically high performance on the PASCAL VOC object detection challenge [11] using regions with the CNN feature model (R-CNN). Their method achieved a mean average precision (mAP) of 54% as compared to the 33% mAP of the HOG-based deformable part model (DPM) [12]. Although R-CNN works well, it runs really slow because each image has around 2,000 region proposals that need to be propagated through the CNN, and it has three different models that require training: the CNN to extract image features; the classifier, which is a support vector machine (SVM) to predict class; and the linear regression model to obtain a tighter bounding box similar to the true dimensions of an object.

To obtain the same dimensions of feature vectors for prediction, the traditional CNN [13] can only run with fixed-size (e.g., 224×224) input images. SPP-net [14] uses the new pooling strategy, spatial pyramid pooling, to eliminate the above requirement. It computes the feature maps from the entire image only once, and then, it uses the pooled features in the subregions to generate fixed-length representations. In 2015, Girshick, the first author of R-CNN, applied the ideas of SPP-net to develop an enhanced version of R-CNN called Fast R-CNN [15]. A region of interest (RoI) pooling layer is set in the CNN to share the forward pass for an image across its subregions. In Fast R-CNN, the CNN is jointly trained with the classifier and the bounding box regressor in a single model.

In the R-CNN, SPP-net, and Fast R-CNN models, the potential region proposals used to detect the locations of objects are created using selective search [16], which is a fairly slow process. Such a slow region proposal method becomes the bottleneck of the overall process. Zitnick and Dollar [17] used edge information to generate the object bounding box proposals. Szegedy et al. [18, 19] developed a learning-based proposal method called multiscale convolutional MultiBox (MSC-MultiBox). Redmon et al. [20] presented another solution that predicts the bounding boxes and the class probabilities directly from the full images in one evaluation.

In 2016, Ren et al. [6] proposed to automatically generate the region proposals using a region proposal network (RPN) that shared the convolutional weights with the CNN. Such method is named as Faster R-CNN. Faster R-CNN consists of two modules. The first module is a deep fully convolutional neural network (FCNN) that proposes regions of interest for object detection. The second module is a Fast R-CNN that uses the proposed regions in the first module to detect objects. Therefore, in Faster R-CNN, only one CNN had to be trained, and the results were used to carry out the region proposals and the classification. A simple summary of the aforementioned object detection methods is shown in Table 1.

The reason we choose Faster R-CNN is because its model size is small compared to other deep learning models for object detection, which make it fast enough to be possible to do real-time iris recognition on a mobile device (for example, smartphones or smart glasses).

2.2. Background of Iris Segmentation. The two typical algorithms for iris segmentation were proposed by Daugman and Wildes using integrodifferential operators [4] and Hough transforms [21], respectively. These methods are based on the idea of finding edge points in an iris image and then fitting them by using circular or elliptical models. For example, Tan et al. [22] presented a combination method of region clustering, semantic refinements, and well-designed integrodifferential operators. Betancourt and Silvente [23] obtained circular boundaries using QMA-OWA operators [24]. Ghodrati et al. [25] used a set of morphological operators, canny edge detector [26], and Hough transforms. Wang and Xiao [27] constructed a difference operator of radial directions. Some other groups used algorithms that rely on region growing instead of edge-based algorithms. They gradually merged the blocks with high correlation in an image to obtain the iris region. Liu et al. [28] used a K-means cluster for pupillary detection. Yan et al. [29] applied the watershed transform [30] and region merging on the structured eye images. Abate et al. [31] combined the watershed transform, region merging, and color quantization. The edge-based and region-growing algorithms estimated the iris region well, but they are not suitable for application to images with various light environments.

The active contour model [32] is another widely used solution for implementing iris segmentation. Jarjes et al. [33] used an angular integral projection function (AIPF) [34] and an active contour model. Bastos et al. [35] combined the pulling and pushing algorithm [36] and the active contour model. Boddeti et al. [37] built the seminal work on active contours without edges [38]. Krichen [39] used the Viterbi algorithm [40] to find a contour that maximizes the gradient value along a connect contour. These algorithms program dynamically and combine the solutions of multiple subproblems. Therefore, they require considerably long processing time in many iterations for achieving better accuracy.

Deep learning is a powerful machine learning tool that has recently exhibited outstanding performance in many fields. Many learning-based methods have been applied to iris segmentation. Tang and Weng [41] used an intensity operator to find the iris' inner border and border recognition with an SVM classifier for the iris outer border. Li et al. [42]

Table 1: Accuracy of the iris segmentation algorithm.

	R-CNN	Fast R-CNN	RPN + Fast R-CNN (Faster R-CNN)
Contributions	Improved performance of PASCAL VOC	Applied SPP-net to R-CNN	Added RPN to Fast R-CNN
Speed on a Tesla K40 GPU [6]	<<0.5 fps	0.5 fps	5–17 fps

built edge detectors based on a set of features including intensity, gradient, texture, and structure information to characterize the edge points and learned six class-specific boundary detectors with AdaBoost [43] for the localization of pupillary and limbic boundaries. Benboudjema et al. [44] presented an implementation of triplet Markov fields (TMF) [45] for segmentation. Happold [46] trained a fast-structured random forest [47] for learning generalized edge detectors. Learning-based algorithms have more complex computations than the other algorithms, and thus, a device requires sufficient computational storage space during training for implementing these algorithms.

More diverse iris segmentation algorithms are included in [48, 49]. Each method has its advantages and disadvantages. In this paper, we propose a combination method of edge-based and learning-based algorithms. The methodology is described in Section 3.

3. Proposed Method

The algorithm proposed in this paper consists of three key steps: eye detection, pupillary boundary estimation, and limbus boundary estimation. We used a Faster R-CNN model to detect the location of an eye in an image. Then, the pupillary and limbus boundaries were found using GMM, maximization of the intensity gradient along the radial emitting path (MIGREP), and boundary point selection algorithms. Thus, the iris region was accurately located.

3.1. Eye Detection. The first step to segment the iris region is to find (detect and locate) the eye in an image. As the task of detecting only two classes, eye or background, in an image is simple, the architecture of CNN in Faster R–CNN does not require very deep convolutional layers. In this study, the original CNN, Zeiler and Fergus (ZF) model [50] or Simonyan and Zisserman model (VGG-16) [51], presented in [6] was replaced with a newly designed network. As depicted in Figure 1, the network contained only six layers. The first convolution layer filtered the grayscale input image with 64 kernels of size $5 \times 5 \times 1$ with a stride of one pixel. It was followed by a rectified linear unit (ReLU) [52] layer and a local response normalization (CN) [13] layer, which ran over five adjacent kernel maps at the same spatial position. A max-pooling layer with a two-pixel gap between the centers of neighborhood pooling units of size 2×2 followed the normalization layer. The second, third, and fourth convolution layers had 64 kernels of size $3 \times 3 \times 64$. A batch normalization (BN) [53] layer and a ReLU layer were applied after the second, third, and fourth layers. The reason we use batch normalization and ReLU is because it reaches the same error rate faster compared to other activation functions such as Tanh function, which means we can train the neural network faster and acquire more neural network models with different parameters. Also, the speed of the model will be faster than other traditionally used activation functions. Interested readers can find more detailed explanation in [13].

The RoI pooling layer extracted a 1024-dimensional feature vector from the output feature maps of the final convolutional layer. The fully connected layer had 128 neurons, and its output that after passing through an ReLU layer was fed to a softmax layer to generate a distribution between the two class labels.

3.2. Gaussian Mixture Model. After generating the potential eye regions with Faster R-CNN, only one bounding box with the maximum score of the eye class and an appropriate aspect ratio was selected to fit the pupillary region. Originally, we planned to use another Faster R-CNN model trained specifically for detecting the pupillary region. However, the result is not as accurate as the model for eye region, and the execution time of two Faster R-CNN models is not fast enough for a real-time iris recognition system. Hence, we decided to use the Gaussian mixture model as our pupillary detection method.

The GMM was built using the expectation maximization (EM) algorithm [7] based on a set of features including the normalized coordinates of pixels, pixel values filtered by a local median of kernel size 5×5, and pixel values filtered using Gabor filters (see Figure 2). A GMM was parameterized by mixture component weights, component means, and covariance matrices. For a GMM with K components, the k^{th} component had the mean μ_k and the covariance matrix $\sum_k$. The posterior probability distribution of GMM can be expressed using the following equations:

$$p(\theta \mid x) = \sum_{i=1}^{K} w_i N\left(\mu_i, \sum_i \mid x\right), \tag{1}$$

$$N\left(\mu_i, \sum_i \mid x\right) = \frac{1}{\sqrt{(2\pi)^K \left|\sum_i\right|}} \cdot \exp\left(-\frac{1}{2}(x-\mu_i)^T \sum_i^{-1} (x-\mu_i)\right), \tag{2}$$

$$\sum_{i=1}^{K} w_i = 1, \tag{3}$$

where θ is the parameter set $\{w, \mu, \sum\}$. The mixture component weight was defined as w_i, and its total number of

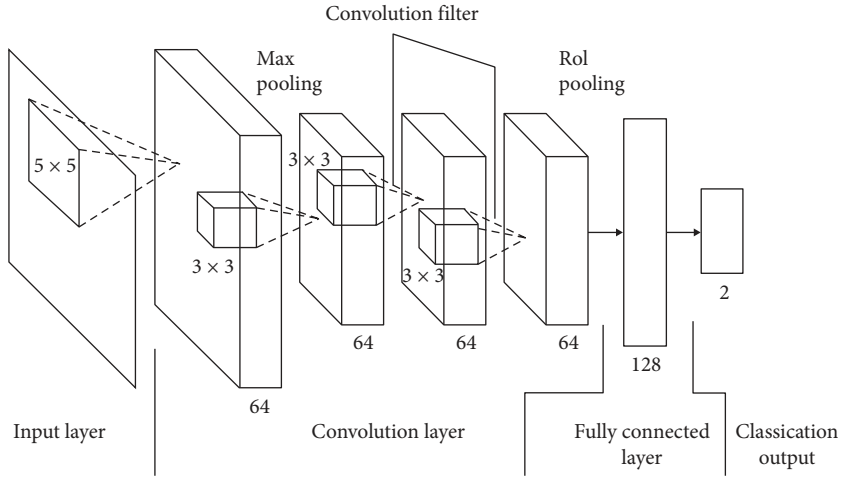


Figure 1: Architecture of proposed CNN.

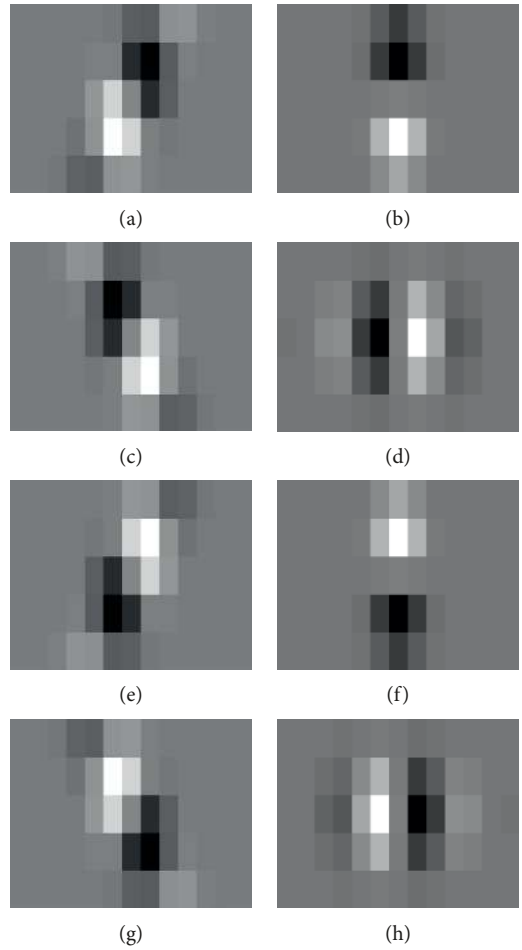


Figure 2: Eight types of Gabor filters used to extract features.

K components was normalized to one. μ_i and $\sum_i$ are mean and covariance matrix of the component i, with a total number of K. In the training stage, the model was trained using the EM algorithm, which is a type of maximum likelihood estimation techniques. The EM algorithm for GMM consists of two steps. The first step, known as the expectation step or E step, is to calculate the expectation of the component C_k for each datum $x_i \in X$, given the model parameters w_k, μ_k, and $\sum_k$. The second step is known as the maximization step or M step, which is needed to maximize

the expectations calculated in the E step with respect to the model parameters and to update the values w_k, μ_k, and $\sum_k$. The entire iterative process repeats steps 1 and 2 until the algorithm converges on the maximum likelihood estimation. As the number of components K is not a known priori parameter in this task, the method [8] is used to adjust the K value automatically during the training stage.

3.3. Pupillary Boundary Estimation. A well-trained GMM can fit the pupillary region inside the region proposal. In general, the result shows a unique candidate pupillary region in each image. However, in some situations, the GMM fits multiple regions consisting of the pupillary region, eyelashes, eyelids, specular reflections, and noisy points. We used a three-step process with three image processing methods (grouping, filling, and morphology opening) to discard the noisy regions and were left with only one candidate pupillary region. As shown in Figure 3, each row presents one test eye image. The left column presents the region proposals produced from Faster R-CNN. The median column presents the candidate pupillary regions predicted by the GMM. Further, the right column presents the final smooth region after applying the image processing methods.

The GMM calculated the probability scores with the eye and the background classes of each pixel in the image. According to this score, several candidate points in the pupillary region could be obtained to smoothen the candidate pupillary region and remove the noisy region. The first step involves grouping regions on the candidate pixels predicted from the GMM using an eight-connected neighborhood algorithm. Then, each subregion was checked for whether it contained more than 250 pixels, and the longer axis of its area was less than 1.15 times the shorter axis. The largest subregion that met the above requirements was considered the pupillary region. If all the regions were outside the specification, the largest region was selected as the pupillary region. Filling the empty space inside the region was the second step. Finally, a morphological opening operator based on a structuring square element of size four was applied to smoothen the region. In the mathematical morphology, the opening operator eroded objects that were smaller than the structuring element and dilated the shape of the remaining region. When empty spaces occurred on the edge of the region, as shown in the bottom row in Figure 3, the filling step prevented the region passing through the opening operator from generating new cracks. More importantly, the opening operator not only smoothened the pupillary region but also eliminated the noisy points, as shown in the top row in Figure 3.

When the pupillary region was drawn up, the coordinates of its center point were easily obtained. To precisely recover the pupillary boundary, a pixel scan of the column and the row was performed at the center point to select the lower, left, and right end points. Because the top end point might be obscured by the upper eyelid, the top point found by the pixel scan was probably different from the actual pupillary boundary point. Instead, two points selected from a new scan performed at the location with the same distance to the center point, and the upper end point was collected. We obtained five key boundary points through the pixel scan methods. The full procedure is shown in Figure 4. After obtaining the five boundary points, each point was denoted according to its coordinates as $(x_i,\ y_i)$. It completely collected five pairs of coordinates of pupillary boundary points. The parameters of an approximate circle are computed using Equation (4). Moreover, the circle with the computed circle parameters could be accurately located on the pupillary boundary, as shown in Figure 5.

$$\{x,\ y,\ r\} = \arg\min_{x,y,r} \sum_{i=1}^{N} (x_i - x)^2 + (y_i - y)^2 - (r_i - r)^2. \tag{4}$$

3.4. Limbus Boundary Estimation. The limbus boundary was estimated after the pupillary region, and its boundary was located. The enhanced version of MIGREP [9] was applied for estimating the coarse limbus boundary. Its required work was to design a few radially emitting paths that went outward from the pupillary center. Hence, the parameters of two distances had to be defined in advance. One, called s_1, was the distance between the starting points of the emitting paths and the pupillary center. The other was defined as s_2 and represented the distance from the pupillary center to the end points of the emitting paths. In [9], these two parameters were predefined and cannot adapt to various input images during runtime. In this work, $(s_1,\ s_2)$ were dynamically adjusted according to the size of the bounding box found by Faster R-CNN. We compared the distances from the edge of the pupillary region to the left and the right sides of the bounding box and selected the shortest one as the basic length, as shown in Figure 6. Then, s_1 and s_2 were assigned the values of the pupillary radius and further incremented by 0.4 and 1.2 times the basic length, respectively. As the bounding box was located by the learning-based algorithm, the basic length associated with it was robust and adjusted automatically during runtime for each image. Thus, most of the emitting paths were supposed to start from somewhere inside the iris region and stop somewhere in the sclera region.

By keeping record of the pixel intensity values along the emitting path, the position that exhibited the maximal variation of pixel intensity was located. This position had to correspond with the intersection between the emitting path and the limbus boundary. Thus, multiple boundary points were successfully estimated when multiple emitting paths were used. Depending on the parameter θ and the shape of the eyelids and the eyelashes, the position showing the maximal value of the intensity gradient was probably not located on the limbus boundary. To solve this problem, we had to consider a set of candidate points where the local maximum gradient occurred, rather than considering only a single point where the global maximum gradient occurred. As depicted in Figure 7, the gradient value of the red point could be higher than that of the blue one, which denoted incorrect boundary point estimation. Therefore, we had to consider a set of candidate points consisting of red and blue points and then, select the point with the highest likelihood from the set.

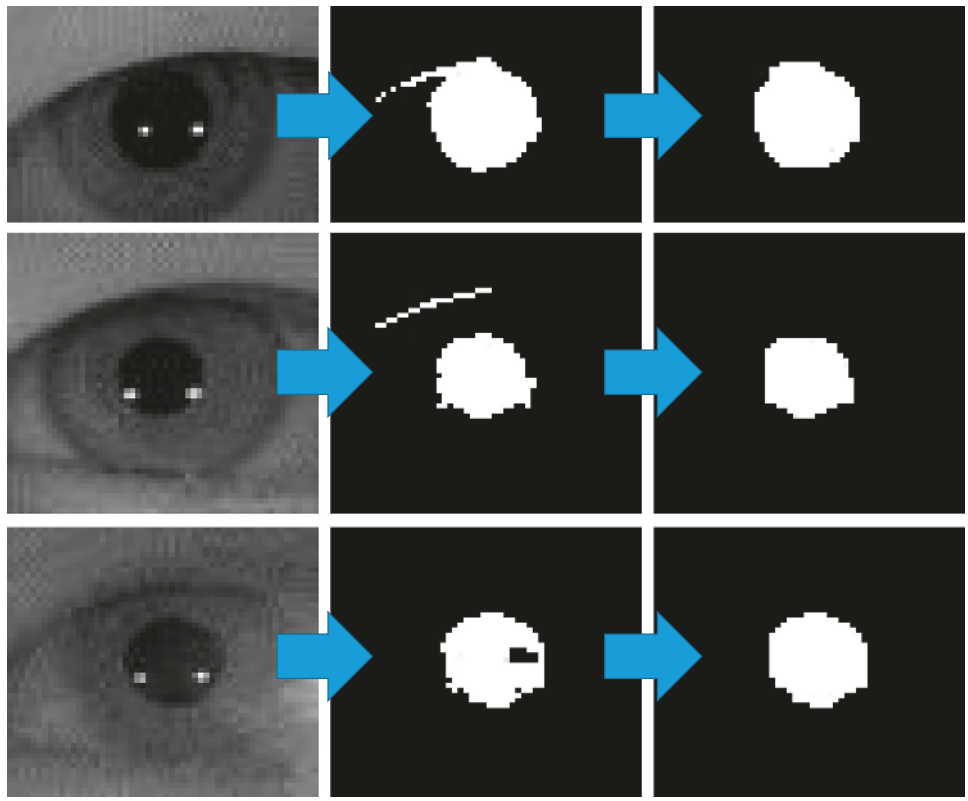
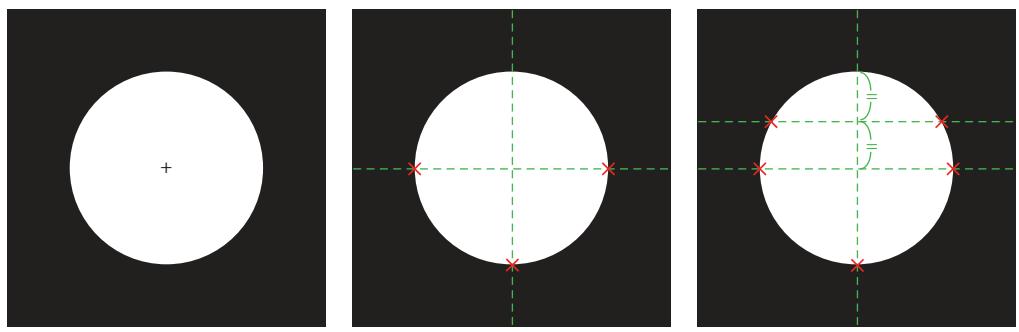
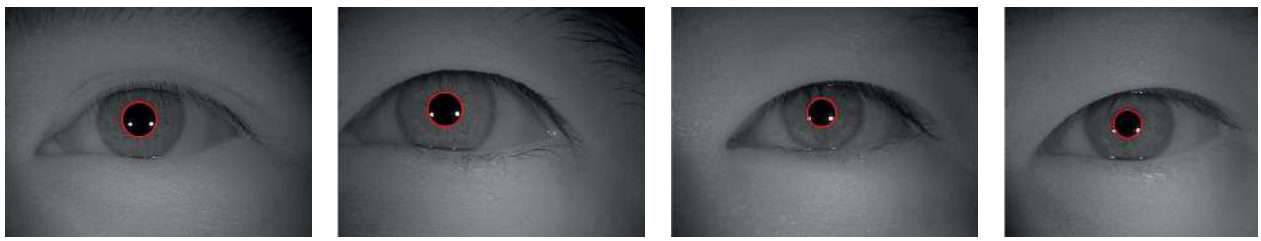


Figure 3: The process of pupillary region prediction.



(a) (b) (c)

Figure 4: Procedure for finding the pupillary boundary points.



(a) (b) (c) (d)

Figure 5: Results of pupillary boundary estimation.

A more sophisticated boundary point selection algorithm was used for this problem. Figure 8 illustrates the idea. First, 11 emitting paths were drawn with the parameter $\theta \in [180^\circ, 210^\circ]$. For paths with such angles, it was highly likely that the maximal gradient occurred on the limbus boundary, as shown in Figure 8(a). Thus, the median value r_m of the distances from these points to the pupillary center was recorded as a reference value. Second, a new emitting

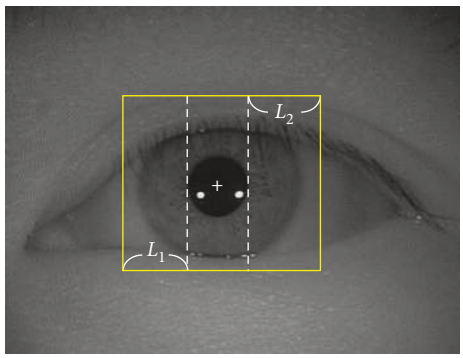


Figure 6: The definition of L_1 and L_2, which are derived based on the localization results from Faster R-CNN. The shortest distance between L_1 and L_2 will be set as the basic length.

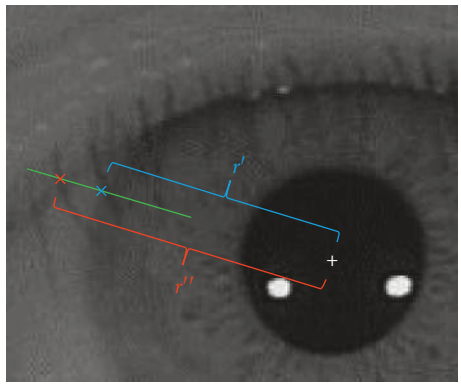


Figure 7: Illustration of boundary point selection problem.

path (with $\theta \in [130^\circ, 170^\circ]$) was drawn, for which an incorrect boundary point might have the maximal gradient, as shown in Figures 8(b)–8(d). In such a case, the corresponding distance values r from the pupillary center to all the points where the local maximal gradient g occurred were respectively recorded. The point that had the larger local maximal gradient value and whose distance value was within $\varepsilon = 2$ of the reference value was selected. Taking Figure 8(b) as an example, assuming that the reference value is r_m in this runtime, the blue point on the new path will be selected on the basis of Equation (5), instead of the red point.

$$k = \arg\max_i \left(g_i \mid |r_m - r_i| < \varepsilon\right). \quad (5)$$

Third, after the best candidate point was selected, the reference value r_m was updated with r_k, which served as the new approximate value of the radius for the boundary points close to it. By repeating the above mechanism for the boundary point selection and the distance updating on the next emitting path with a new θ value ranging from $[130^\circ, 240^\circ]$ to $[-60^\circ, 50^\circ]$, we gradually adjusted the coarse limbus boundary points to more precise locations, as shown in Figure 8(e). Sometimes, the reflection point of light occurred on the limbus boundary and might cause the iteration of the mechanism to go into a bad evolution. Therefore, the distance updating might not apply to the pixels whose pixel values were larger than those of the normal pixels that had 95% probability in the normal distribution established using the pixel values of the complete image.

4. Experimental Results and Discussion

4.1. Database. The database used to train Faster R-CNN and GMM was the CASIA-Iris-Thousand database [54]. This database contains 1,000 subjects with a total of 20,000 iris images, which were collected using the IKEMB-100 camera. As a large number of subjects wore glasses during image capturing, many images have glass frames and specular reflection. These types of obstructions were obstacles to the iris segmentation.

4.2. Detection Model Training. Faster R-CNN and GMM used the full CASIA-Iris-Thousand database for the training and the test. The training set had 6,000 right-eye and 6,000 left-eye images, and the test set had 4,000 right-eye and 4,000 left-eye images. Each image had the region information of the iris that was manually labeled, as shown in Figure 9. To build the proposed algorithm in a mobile device or an embedded system, the model had to occupy less storage space and has lower computational complexity. The model was trained using the training images, previously reduced to the specified size. However, in the test stage, the test images were resized in runtime to pass through the model. The results of the detection were mapped onto the original test images to ensure that there were sufficient iris textures inside the bounding boxes for use in the other iris recognition steps.

To share the convolutional weights between CNN and RPN in Faster R-CNN, the model had to be trained in four steps. The first step consisted of training a region proposal network. For the convolutional feature map of size $W \times H$ outputted from the fourth convolution layer of the proposed model, RPN found the $W \times H \times k$ potential regions. Using of the last convolution layer as feature map has been applied and proven very effectively by other object detecting convolution neural network such as R–CNN and Faster R–CNN. Interested readers can find more details of why using the last layer for feature by reading [6, 10, 15].

However, only 2,000 regions with the higher intersection over union (IoU) value were assigned to positive samples for training the CNN. In the second step, a separate detection network by Fast R–CNN was trained using the region proposals generated from the RPN built in Step 1. At this stage, the two networks did not yet share the convolutional weights. In the third step, the detection network was used to initialize RPN training. It frozed the weights of the shared convolution layers and fine-tuned the layers that belonged only to the RPN during training. The final step was to fine-tune (with the same operation) the layers that only belonged to the CNN. Hence, the networks shared the same convolution layers and merged into a single network.

For the purpose of finding the best architecture of the RPN and CNN model, we trained multiple models with different fine-tuned parameter sets using the right-eye images of the CASIA-Iris-Thousand database, as shown in Tables 2 and 3. The new architecture of the CNN model was

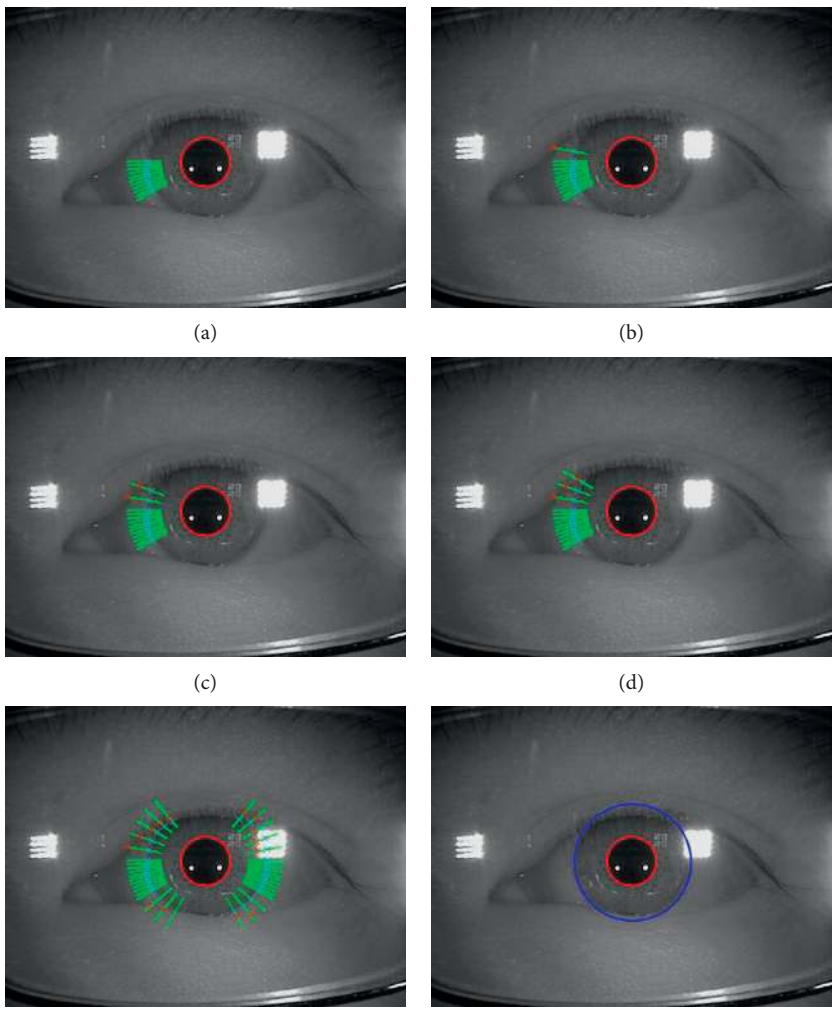


Figure 8: Illustration of boundary point selection algorithm. (a) 11 emitting paths are drawn with the parameter $\theta \in [180^{\circ}, 210^{\circ}]$ (for the other side $\theta \in [-30^{\circ}, 0^{\circ}]$), and the points corresponding to the maximal gradient are recorded. The reference value r_{m} is determined by taking the median of the distances from these points to the pupillary center. (b–e) Repeatedly drawing new emitting paths and applying the boundary point selection algorithm result in the correct location of many boundary points. (f) Final limbus boundary is recovered by fitting a circle on all candidate points based on Equation (4).

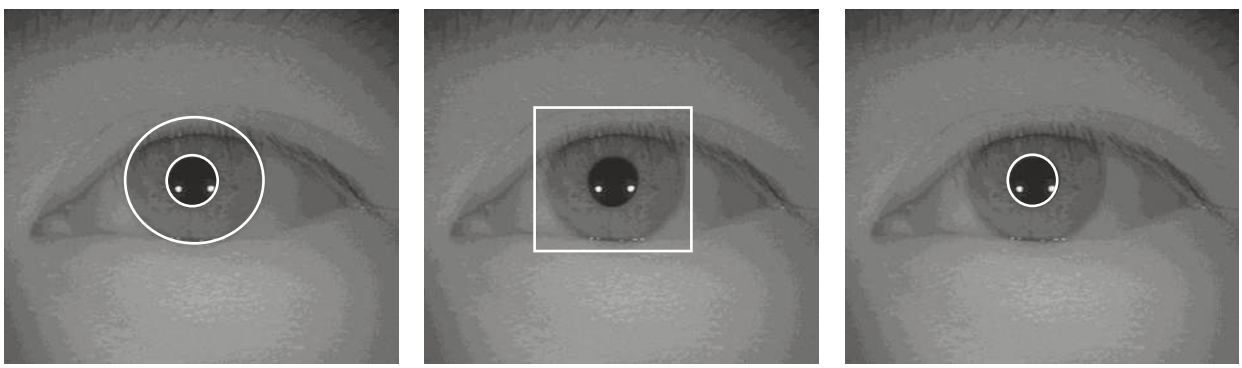


Figure 9: Manually labeled region information of the iris. (a) The manually labeled iris region. (b) The bounding box computed from the circle parameters of the limbus boundary is the region label for training Faster R-CNN. (c) The information of the pupillary region used for training GMM.

Table 2: Experimental results of finding the best architecture of CNN. Note that the "conv_1" and "conv_2" layers shown in the table correspond to the sequences Conv-ReLU-CN and Conv-BN-ReLU, respectively. Further, all the convolution layers run with stride one.

	Test time	Input image size	Layers	RoI pooling grid size	Fully connected layer
A	0.191 s	480 × 640	(3 × 3 × 64) × 3 conv_2 (2 × 2) max pool (3 × 3 × 64) × 3 conv_2	120 × 120	128 ⟶ 2
B	0.197 s	480 × 640	(3 × 3 × 64) × 2 conv_2 (2 × 2) max pool (3 × 3 × 64) × 3 conv_2	120 × 120	256 ⟶ 2
C	0.286 s	480 × 640	(7 × 7 × 128) × 1 conv_1 (2 × 2) max pool (3 × 3 × 128) × 3 conv_2	120 × 120	128 ⟶ 2
D	0.295 s	480 × 640	(7 × 7 × 128) × 1 conv_1 (2 × 2) average pool (3 × 3 × 128) × 3 conv_2	120 × 120	128 ⟶ 2
E	0.286 s	240 × 320	(7 × 7 × 128) × 1 conv_1 (2 × 2) max pool (3 × 3 × 128) × 3 conv_2	64 × 64	128 ⟶ 2
F	0.042 s	120 × 160	(3 × 3 × 128) × 1 conv_1 (2 × 2) max pool (3 × 3 × 128) × 3 conv_2	32 × 32	128 ⟶ 2
G	0.036 s	120 × 160	(5 × 5 × 128) × 1 conv_1 (2 × 2) max pool (3 × 3 × 128) × 3 conv_2	32 × 32	128 ⟶ 2
H	0.037 s	120 × 160	(5 × 5 × 64) × 1 conv_1 (2 × 2) max pool (3 × 3 × 64) × 3 conv_2	32 × 32	128 ⟶ 2
I	0.037 s	120 × 160	(5 × 5 × 64) × 1 conv_1 (2 × 2) max pool (3 × 3 × 64) × 3 conv_2	32 × 32	128 ⟶ 2
J	0.033 s	60 × 80	(5 × 5 × 64) × 1 conv_1 (2 × 2) max pool (3 × 3 × 64) × 3 conv_2	16 × 16	128 ⟶ 2

designed on the basis of VGG-16. As the detection task in this study was simple, we reduced the number of convolution layers of VGG-16. Precision and recall were used to measure the performance of the detector. Precision is the fraction of retrieved objects relevant to the detection, and recall is the fraction of relevant objects successfully retrieved. Here, we set an overlap threshold of IoU = 0.8 to select effective detection, which was a strict condition.

The initial version of the new network architectures was labeled Model A and Model B, which had only six and five convolutional layers, respectively. The experimental results showed that the performance of Model B was considerably worse than that of Model A, even when the number of neurons in the fully connected layer was increased. Next, we attempted to replace the first three layers of the network with a convolutional layer of a larger kernel size, which resulted in Models C and D. The use of multiple kernel sizes in a network helped the network to obtain more diverse features in an image. The difference between these two models was the different pooling strategies used, namely, max pooling for Model C and average pooling for Model D. Irrespective of the pooling strategy, their performance was almost 100% precision and recall. Although the models performed well, they used a large number of computed parameters in the networks and thus required a long processing time of approximately 0.3 s to complete the detection. Therefore, we reduced the size of the training set by 2×, 4×, and 8× to generate Models E, I, and J, respectively. The smaller was the size of the images used for training, the less was the time required for the model training and testing and the lower was the detection accuracy. According to the experimental results, the performance of Model J was the worst of all the models trained using images of different sizes. This might be attributed to the fact that the images used for training had very few features for the detection when they were shrunk considerably. We finally used the architecture of Model I to implement the algorithm proposed in this paper. Models F, G, and H were the parameter-adjusted results of Model I. Among them, Model I exhibited better performance and sufficiently low time consumption for the detection.

The GMM was trained using the images with the information of the pupillary region. We used the GMM to fit the potential pupillary region inside the bounding box found by Faster R–CNN. Each pixel in an image was represented by a nine-dimensional feature vector used for the training and the testing. The features consisted of the normalized coordinates of pixels, pixel values filtered by a local median of kernel size 5 × 5, and pixel values filtered using Gabor filters. The Gabor filters of size 5 × 13 were parameterized as follows: $\sigma = 2$, $\theta = [45^\circ, 360^\circ]$, $\lambda = 5$, $\psi = 1.5$, and $\gamma = 2.5$. In

Table 3: Experimental results of finding the best architecture of RPN.

	Overlap range	MinBox sizes	Box pyramid scale	NumBox pyramid levels	Precision	Recall
A	[0, 0.3; 0.7, 1]	[125, 125]	1.1	9	0.9810	0.9793
B	[0, 0.3; 0.7, 1]	[125, 125]	1.1	9	0.9489	0.9568
C	[0, 0.3; 0.7, 1]	[125, 125]	1.1	9	0.9922	0.9918
D	[0, 0.3; 0.7, 1]	[125, 125]	1.1	9	0.9900	0.9888
E	[0, 0.3; 0.7, 1]	[64, 64]	1.1	9	0.9825	0.9825
F	[0, 0.3; 0.65, 1]	[32, 32; 48, 32]	1.2	5	0.9737	0.9800
G	[0, 0.3; 0.65, 1]	[32, 32; 48, 32]	1.2	5	0.9762	0.9778
H	[0, 0.3; 0.65, 1]	[32, 32; 48, 32]	1.2	5	0.9604	0.9640
I	[0, 0.3; 0.65, 1]	[32, 32; 32, 48]	1.2	5	0.9778	0.9805
J	[0, 0.3; 0.6, 1]	[16, 16; 24, 16; 16, 24; 24, 24]	1.1	9	0.9400	0.9398

"Overlap Range" is the bounding box overlap ratios for selecting negative and positive samples. "MinBox Sizes" is the minimum anchor box sizes used to build the anchor box pyramid. "Box Pyramid Scale" is the anchor box pyramid scale factor used to successively upscale anchor box sizes. "NumBox Pyramid Levels" is the number of levels in an anchor box pyramid.

the training stage, the pixels inside the pupillary region were taken as the positive samples. A normal distribution built from the pixel values of the entire region was used to remove the positive samples located in the region of the reflection points. The same number of samples as in the positive sample was selected from the pixels out of the pupillary region to form a negative sample. We also attempted to use SVM instead of GMM to predict the potential pupillary region. However, it did not perform as well as GMM, as it took more than three days for training, which was considerably much longer than GMM which only takes 5 min. Furthermore, its accuracy of region prediction was poor, as shown in Figure 10.

We implemented our algorithm with MATLAB R2018a and run it on a personal computer with 3.4-GHz CPUs and GTX 1080 GPU. The average time cost per eye of iris segmentation was approximately 0.06 s, which indicated that the proposed algorithm is a fast iris segmentation algorithm.

4.3. Performance Evaluation for Iris Segmentation. Traditionally, most researchers have evaluated the results of iris segmentation with subjective methods, for example, by reading the iris segmentation results on the plotted image and manually giving the judgment [9, 27, 29, 33, 35, 36, 39, 41]. To quantitatively estimate the performance of pupillary boundary localization and limbus boundary localization, we propose a new method based on the integration of the radial difference. For each image, we used the region information of the manually labeled iris region to generate two separate binary maps containing the pupillary region and the iris region, respectively. We assumed a segmentation S that was parameterized by the coordinates of the circle's center and its radius, denoted as a triple set (x_c, y_c, r). Then, we created a dilated version S_d^+ and an eroded version S_d^- of S, which was parameterized as $(x_c, y_c, r+d)$ and $(x_c, y_c, r-d)$, respectively. As such, every point of S had its corresponding points on S_d^+ and S_d^-. By collecting N pairs of corresponding points on S_d^+ and S_d^-, denoted as (p_i^+, p_i^-), we evaluated the performance of S by using the q value computed using Equation (6). Figure 11 illustrates the procedure for the performance evaluation.

$$q = \frac{\sum_{i=1}^{N} (p_i^- - p_i^+)}{N}, \quad i \in [1, N]. \tag{6}$$

We compared our algorithm with [9], which was proved to be very robust and efficient for iris images captured on wearable devices. The proposed performance evaluation method was used with parameters $d = 10\,(15)$ and $N = 36\,(36)$ for evaluating the performance of the pupillary (limbus) boundary localization. With such a d value, it ensured that the results of the proposed segmentation algorithm had at least a 0.5 IoU value with the ground truth. By selecting the aforementioned parameters, we make the proposed algorithm to be fast enough for a real-time iris recognition system (above 15 frames per second) while maintaining the accuracy of iris segmentation. It set $q = 0.9$ as the threshold to select effective segmentation and computed the accuracy of segmentation with this threshold. Figure 12 illustrates the histogram of the q

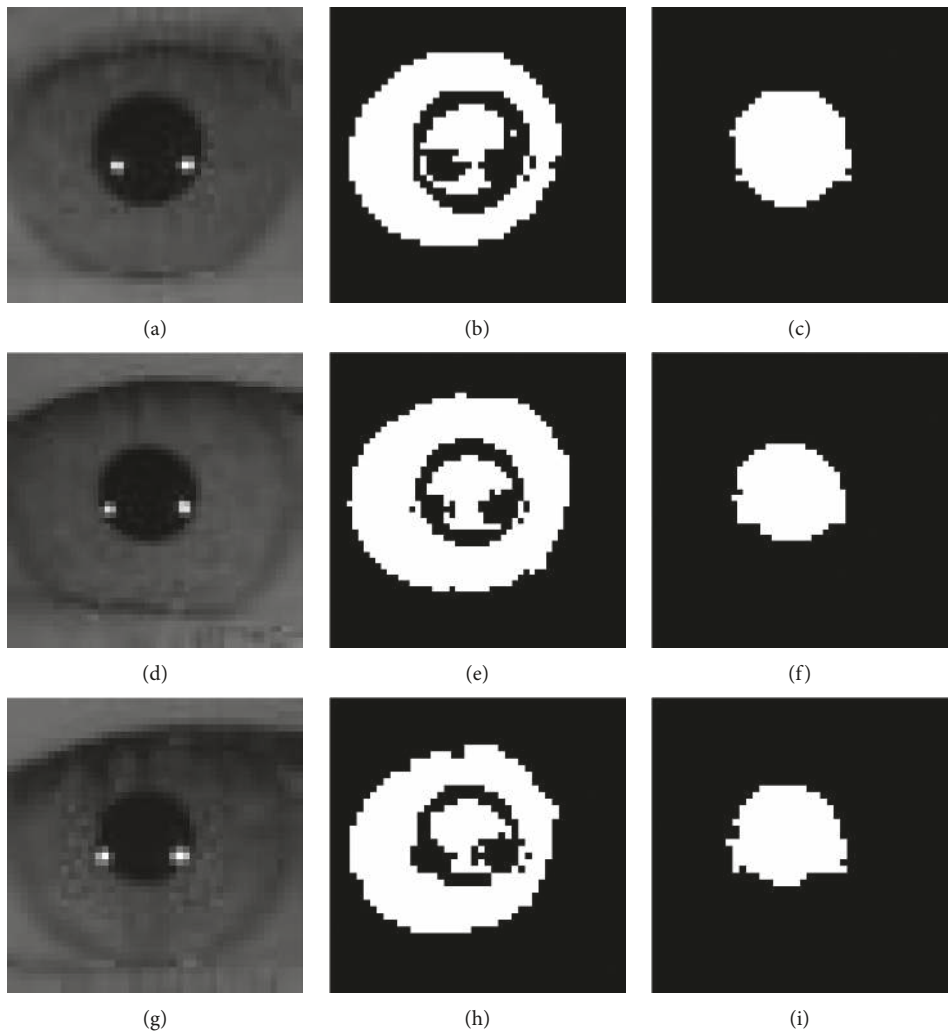


Figure 10: Performance comparison between SVM and GMM. The second column presents the region fit by SVM and the third column shows that fit by GMM.

values for evaluating the segmentation performance on the full CASIA-Iris-Thousand database. The accuracy of segmentation is shown in Table 4. As depicted in Table 4, the proposed algorithm showed a dramatic increase from 47.84% to 95.49%. This could be attributed to the fact that the method used for the localization of the eye was changed to a learning-based algorithm. As such, the parameters used to find the boundary points of the pupillary region and the iris were robustly adjusted automatically during runtime for different input images.

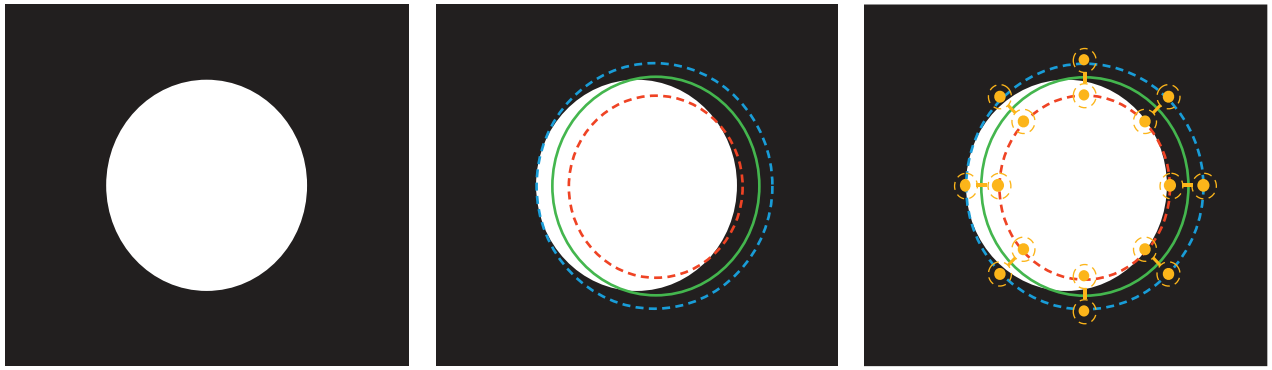
4.4. Difference between the Proposed Method and Other Published Methods. There are many iris segmentation methods based on deeply learned neural networks. In this section, we discuss the difference between two state-of-the-art methods, IrisDenseNet [55] and the model proposed by He et al. in [56].

IrisDenseNet uses a 13-layered VGG-16 network [51] as its core to detect actual iris area (excluding area such as eyelid and eyelashes). However, it only performs segmentation for the iris area without a proper method to normalize it. As we can see in [2–4], the iris normalization is a key stage for high-performance iris recognition. If this stage is missing, there is no guarantee that the final accuracy of their iris recognition system still remains the desired precision. Also, due to its deep layers, the computation complexity of training and using it is extremely costing compared to our proposed method.

Model proposed in [56] also employs VGG-16 network but with some changes. Its execution time for one image is 0.112 second on a 2.6 GHz CPU and GTX970 M GPU which, again, is not fast enough for a real-time iris recognition system on the embedded system. Our proposed method, on the contrary, can perform iris localization within 0.06 seconds, which is 1.87x faster.

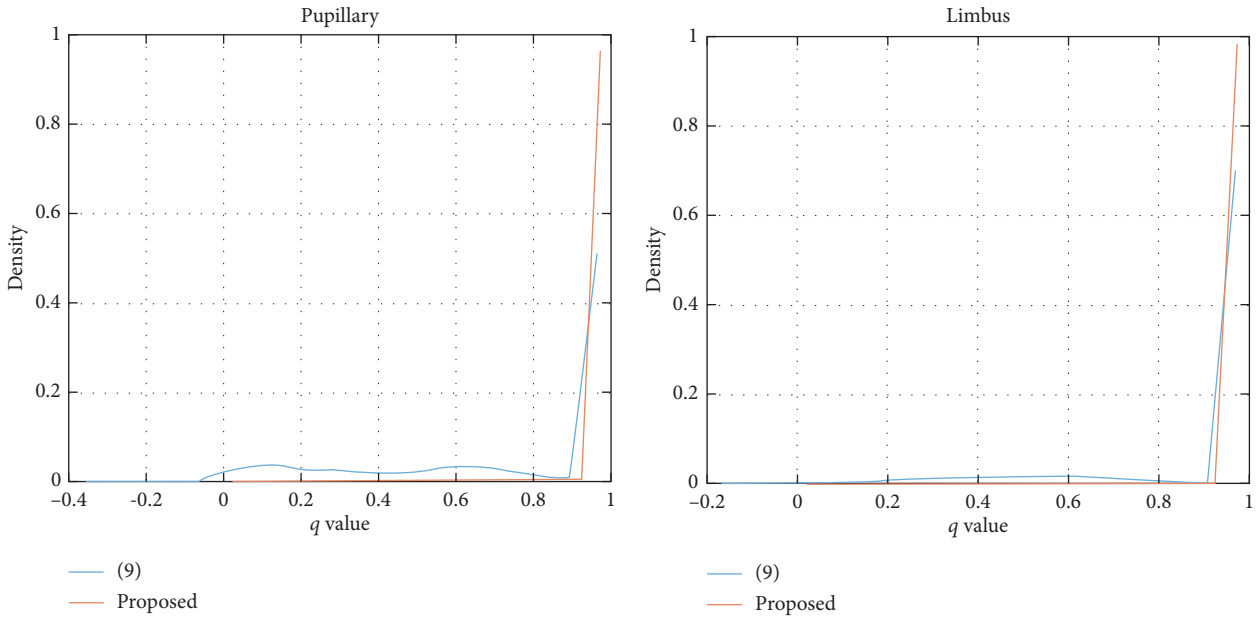
5. Conclusion

In this paper, we presented a robust and fast iris segmentation algorithm based on Faster R-CNN. We reconstructed the CNN architecture of Faster R-CNN. This new model with only six layers could generate precisely located region proposals of the



(a) (b) (c)

Figure 11: Performance evaluation of segmentation. The first image illustrates an example of a binary map. S (green circle), S_d^+ (blue-dotted circle), and S_d^- (red-dotted circle) are drawn on the binary map in the second image. The q value is the mean of the integration of the difference between the yellow point pairs in the third image used to evaluate the segmentation performance.



(a) (b)

Figure 12: Histogram of q values for evaluating the segmentation performance.

Table 4: Accuracy of the iris segmentation algorithm.

	Proposed	[9]
Pupillary boundary	96.77%	51.60%
Iris boundary	98.32%	70.17%
Both boundaries	95.49%	47.84%

eye in the images. We then extracted the feature vectors with specific dimensions to train a GMM for fitting the potential pupillary regions. Then, the pupillary boundary was recovered through five key boundary points found by pixel scans of the rows and columns. An enhanced version of MIGREP and the boundary point selection algorithm were used to find some boundary points of the limbus region, and the limbus boundary was located by using these boundary points. To evaluate the performance of iris segmentation, we developed an evaluation method based on the integration of the radial difference. Experimental results showed the effectiveness and efficiency of the proposed iris segmentation method on the CASIA-Iris-Thousand database. The segmentation accuracy of the proposed method was 95.49%, which was higher than the accuracy of 47.84% achieved in the previous work, and the time cost of the proposed iris segmentation procedure was only approximately 0.06 s. The results on the challenging CASIA-Iris-Thousand database showed that the proposed method is a fast and accurate iris segmentation algorithm.

The main advantage of the proposed algorithm over most of the state-of-the-art iris segmentation algorithms based on neural networks such as IrisDenseNet [55] and the model proposed by He et al. [56] is that it has a smaller

model size which make it faster to segment iris images, which is crucial for a real-time iris recognition system or even implement it on a mobile device.

For the future work, we want to further improve the speed of the algorithm by creating heterogeneous models that combining the power of CNN and the speed of traditional computer vision methods. Another direction is to try to use the semantic segmentation method and combine it with the proposed algorithm. The semantic segmentation algorithm has a high sensitivity of predicting the reflection points in the iris region, which can improve the overall accuracy of the algorithm. After the algorithm is improved, we will attempt to build the algorithm on the mobile devices, by using more concise deep learning models, such as XNOR-NET [57]. The ultimate goal is to implement a fast and accurate portable iris recognition system.

Data Availability

The CASIA database [54] used to support the research is provided by the Institute of Automation, Chinese Academy of Science.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the Ministry of Science and Technology, Taiwan (grant number 106-2221-E-008-102-).

References

[1] *Security and Data Protection in a Google Data Center*, https://www.youtube.com/watch?v=cLory3qLoY8.

[2] J. Daugman, "Probing the uniqueness and randomness of IrisCodes: results from 200 billion iris pair comparisons," *Proceedings of the IEEE*, vol. 94, no. 11, pp. 1927–1935, 2006.

[3] J. Daugman, "Information theory and the IrisCode," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 2, pp. 400–409, 2016.

[4] J. G. Daugman, "High confidence visual recognition of persons by a test of statistical independence," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 15, no. 11, pp. 1148–1161, 1993.

[5] N. Othman, B. Dorizzi, and S. Garcia-Salicetti, "OSIRIS: an open source iris recognition software," *Pattern Recognition Letters*, vol. 82, pp. 124–131, 2016.

[6] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, 2016.

[7] J. A. Bilmes, "A gentle tutorial of the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden markov models," University of Berkeley, Technical Report ICSI-TR-97-021, 1998.

[8] M. A. T. Figueiredo and A. K. Jain, "Unsupervised learning of finite mixture models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 3, pp. 381–396, 2002.

[9] Y.-H. Li and P.-J. Huang, "An accurate and efficient user authentication mechanism on smart glasses based on Iris recognition," *Mobile Information Systems*, vol. 2017, Article ID 1281020, 14 pages, 2017.

[10] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of 2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 580–587, Columbus, OH, USA, June 2014.

[11] M. Everingham, L. V. Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL visual object classes (VOC) challenge," *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, 2010.

[12] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramona, "Object detection with discriminatively trained part-based models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1627–1645, 2010.

[13] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems*, vol. 25, pp. 1097–1105, 2012.

[14] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 9, pp. 1904–1916, 2015.

[15] R. Girshick, "Fast R-CNN," in *Proceedings of 2015 IEEE International Conference on Computer Vision*, pp. 1440–1448, Santiago, Chile, 2015.

[16] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders, "Selective search for object recognition," *International Journal of Computer Vision*, vol. 104, no. 2, pp. 154–171, 2013.

[17] C. L. Zitnick and P. Dollar, "Edge boxes: locating object proposals from edges," in *Proceedings of European Conference on Computer Vision*, pp. 391–405, Zurich, Switzerland, 2014.

[18] D. Erhan, C. Szegedy, A. Toshev, and D. Anguelov, "Scalable object detection using deep neural networks," in *Proceedings of 2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2155–2162, Columbus, OH, USA, June 2014.

[19] C. Szegedy, S. Reed, D. Erhan, D. Anguelov, and S. Ioffe, "Scalable, "High-Quality object detection," arXiv 1412.1441v3, 2015.

[20] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: unified, real-time object detection," in *Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 779–788, Las Vegas, NV, USA, 2016.

[21] R. Wildes, "Iris recognition: an emerging biometric technology," *Proceedings of the IEEE*, vol. 85, no. 9, pp. 1348–1363, 1997.

[22] T. Tan, Z. He, and Z. Sun, "Efficient and robust segmentation of noisy Iris images for non-cooperative Iris recognition," *Image and Vision Computing*, vol. 28, no. 2, pp. 223–230, 2010.

[23] Y. A. Betancourt and M. G. Silvente, "A fast Iris location based on aggregating gradient approximation using QMA-OWA operator," *International Conference on Fuzzy Systems*, pp. 1–8, 2010.

[24] J. I. Pelaez and J. M. Dona, "A majority model in group decision making using QMA-OWA operators," *International Journal of Intelligent Systems*, vol. 21, no. 2, pp. 193–208, 2006.

[25] H. Ghodrati, M. J. Dehghani, M. S. Helfroush, and K. Kazemi, "Localization of noncircular Iris boundaries using morphology and arched Hough transform," in *Proceedings of 2010 2nd International Conference on Image Processing Theory, Tools and Applications*, pp. 458–463, Paris, France, 2010.

[26] J. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 6, pp. 679–698, 1986.

[27] X.-C. Wang and X.-M. Xiao, "An Iris segmentation method based on difference operator of radial directions," in *Proceedings of 2010 6th International Conference on Natural Computation*, pp. 135–138, Yantai, China, August 2010.

[28] J. Liu, X. Fu, and H. Wang, "Iris image segmentation based on K-means cluster," in *Proceedings of 2010 IEEE International Conference on Intelligent Computing and Intelligent Systems*, pp. 194–198, Xiamen, China, October 2010.

[29] F. Yan, Y. Tian, H. Wu, Y. Zhou, L. Cao, and C. Zhou, "Iris segmentation using watershed and region merging," in *Proceedings of 2014 9th IEEE Conference on Industrial Electronics and Applications*, pp. 835–840, Hangzhou, China, June 2014.

[30] J. B. T. M. Roerdink and A. Meijster, "The watershed transform: definitions, algorithms and parallelization strategies," *Fundamenta Informaticae*, vol. 41, no. 1-2, pp. 187–228, 2000.

[31] A. F. Abate, M. Frucci, C. Galdi, and D. Riccio, "BIRD: watershed based Iris detection for mobile devices," *Pattern Recognition Letters*, vol. 57, pp. 41–49, 2015.

[32] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: active contour models," *International Journal of Computer Vision*, vol. 1, no. 4, pp. 321–331, 1988.

[33] A. A. Jarjes, K. Wang, and G. J. Mohammed, "Iris localization: detecting accurate pupil contour and localizing limbus boundary," in *Proceedings of 2010 2nd International Asia Conference on Informatics in Control, Automation and Robotics*, pp. 349–352, Wuhan, China, March 2010.

[34] G. J. Mohammed, B.-R. Hong, and A. A. Jarjes, "Accurate pupil features extraction based on new projection function," *Computing and Informatics*, vol. 29, no. 4, pp. 663–680, 2009.

[35] C. A. C. M. Bastos, I. R. Tsang, and G. D. C. Calvalcanti, "A combined pulling & pushing and active contour method for pupil segmentation," in *Proceedings of 2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 850–853, Dallas, TX, USA, March 2010.

[36] Z. He, T. Tan, and Z. Sun, "Iris localization via pulling and pushing," in *Proceedings of 18th International Conference on Pattern Recognition*, pp. 366–369, Hong Kong, China, August 2006.

[37] V. N. Boddeti, B. V. K. V. Kumar, and K. Ramkumar, "Improved Iris segmentation based on local texture statistics," in *Proceedings of 2011 Conference Record of the Forty Fifth Asilomar Conference on Signals, Systems and Computers*, pp. 2147–2151, Pacific Grove, CA, USA, 2011.

[38] T. F. Chan and L. A. Vese, "Active contours without edges," *IEEE Transactions on Image Processing*, vol. 10, no. 2, pp. 266–277, 2001.

[39] E. Krichen, "Lef3a: pupil segmentation using Viterbi search algorithm," in *Proceedings of 2012 5th IAPR International Conference on Biometrics*, pp. 323–329, New Dehli, India, 2012.

[40] A. J. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Transactions on Information Theory*, vol. 13, no. 2, pp. 260–269, 1967.

[41] R. Tang and S. Weng, "Improving Iris segmentation performance via borders recognition," in *Proceedings of 2011 4th International Conference on Intelligent Computation Technology and Automation*, pp. 580–583, Shenzhen, China, August 2011.

[42] H. Li, Z. Sun, and T. Tan, "Robust Iris segmentation based on learned boundary detectors," in *Proceedings of 2012 5th IAPR International Conference on Biometrics*, pp. 317–322, New Dehli, India, 2012.

[43] J. Friedman, T. Hastie, and R. Tibshirani, "Additive logistic regression: a statistical view of boosting," *The Annals of Statistics*, vol. 28, no. 2, pp. 337–407, 2000.

[44] D. Benboudjema, N. Othman, B. Dorizzi, and W. Pieczynski, "Challenging eye segmentation using triplet markov spatial models," in *Proceedings of 2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 1927–1931, Vancouver, BC, Canada, May 2013.

[45] W. Pieczynski, D. Benboudjema, and P. Lanchantin, "Statistical image segmentation using triplet markov fields," in *Proceedings of Image and Signal Processing for Remote Sensing*, Agia Pelagia, Crete, Greece, 2002.

[46] M. Happold, "Structured forest edge detectors for improved eyelid and Iris segmentation," in *Proceedings of 2015 International Conference of the Biometrics Special Interest Group*, pp. 28–33, Darmstadt, Germany, September 2015.

[47] P. Dollar and C. L. Zitnick, "Structured forests for fast edge detection," in *Proceedings of 2013 IEEE International Conference on Computer Vision*, pp. 1841–1848, Sydney, NSW, Australia, December 2013.

[48] K. W. Bowyer, K. P. Hollingsworth, and P. J. Flynn, "A survey of Iris biometrics research: 2008-2010," in *Handbook of Iris Recognition*, Springer, London, UK, 2013.

[49] M. R. Rajput and G. S. Sable, "IRIS biometrics survey 2010-2015," in *Proceedings of 2016 IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology*, pp. 2028–2033, Bangalore, India, May 2016.

[50] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proceedings of European Conference on Computer Vision*, pp. 818–833, Zurich, Switzerland, September 2014.

[51] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proceedings of International Conference on Learning Representations*, San Diego, CA, USA, May 2015.

[52] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proceedings of the 27th International Conference on Machine Learning*, pp. 807–814, Haifa, Israel, June 2010.

[53] S. Ioffe and C. Szegedy, "Batch normalization: accelerating deep network training by reducing internal covariate shift," in *Proceedings of the 32nd International Conference on Machine Learning*, vol. 37, pp. 448–456, Lille, France, July 2015.

[54] *CASIA Iris Image Database*, http://biometrics.idealtest.org/.

[55] M. Arsalan, R. A. Naqvi, D. S. Kim et al., "Robust Iris segmentation using densely connected fully convolutional networks in the images by visible light and near-infrared light camera sensors," *Sensors*, vol. 18, no. 5, 2018.

[56] Y. He, S. Wang, K. Pei, M. Liu, and J. Lai, "Visible spectral Iris segmentation via deep convolutional network," *Biometric Recognition*, pp. 428–435, 2017.

[57] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi, "XNOR-net: ImageNet classification using binary convolutional neural networks," in *Proceedings of European Conference on Computer Vision*, pp. 525–542, Amsterdam, The Netherlands, October 2016.