IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Ingénierie des Systèmes d'Information
Vol. 30, No. 4, April, 2025, pp. 891-899
Journal homepage: http://iieta.org/journals/isi

Efficient and Robust Iris Localization Framework for Real-World Noisy Images

Dena Nadir George[1], Noor A. Yousif [2], Samar Amil Qassir[3*]

[1] Department of Computer Science, Collage of Education, Mustansiriyah University, Baghdad 10052, Iraq
[2] Control and Systems Eng. Dept., University of Technology-Iraq, Bagdad 10066, Iraq
[3] Department of Computer Science, College of Science, Mustansiriyah University, Baghdad 10052, Iraq

Corresponding Author Email: samarqassir@uomustansiriyah.edu.iq

https://doi.org/10.18280/isi.300406

Received: 9 January 2025
Revised: 12 April 2025
Accepted: 22 April 2025
Available online: 30 April 2025

Keywords:
biometrics, iris localization, Haar Cascade Classifier (HCC), AdaBoost, Hough Circle Transform (HCT)

ABSTRACT

The iris pattern is one of the most precise and dependable biometrics that is frequently used for user authentication systems because of its stability and uniqueness. Delineating the inner and outer boundaries of the actual iris in the eye's part is the goal of the iris localization. Dealing with less-than-ideal iris images can result in an inaccurate location, making this localization process difficult. To describe the pupillary boundaries in facial images with varying skin colors, eye colors, and eye sizes, the traditional methods can be noisy, antiquated, and possibly inaccurate. In order to solve this problem, this paper introduced a robust framework that uses the AdaBoost and Haar Cascade to localize iris in complex conditions. Five phases that the introduced framework goes through. It was evaluated on both standard and non-standard photos using three datasets: the Labeled Faces in the Wild (LFW), the MMU V1.0, and the Iris Super Resolution (ISR), from which images of entire faces and images of eyes only were chosen. According to the experiments, the introduced algorithm rate was 100% for 220 eye images in the ISR, 99.33% for 300 eye images in the MMU, and 98.88% for 180 face photos in the LFW.

1. INTRODUCTION

Secure human recognition techniques have come a long way, combining computer vision, biometrics, and artificial intelligence to guarantee high security, accuracy, and dependability in both public and private settings. Traditional security mechanisms like credit cards, passwords, and personal identification numbers have been shown to be unreliable due to security breaches that occur globally [1]. This fact is evident from the media, which frequently reports on hacking of these conventional techniques. The biometric application was used by the research community to address these problems [2, 3]. This technology recognizes people based on their physical and behavioral characteristics. Facial identification, iris and retinal scanning, fingerprint recognition, behavioral biometrics, and vein pattern recognition are some of the most potent techniques currently in use [4-6]. When compared to more conventional security methods, biometric technology has been shown to be a dependable security measure. It has enormous potential for use in covert applications, such as tracking terrorist activity, in addition to its overt ones [7]. Among these biometric applications is iris recognition. As seen in Figure 1, the iris is an annulus that lies between the pupil and the sclera. It is an externally visible internal organ that is protected by the cornea. The eye's iris is unique, as seen by the differences in iris patterns even between identical twins [8, 9].

Iris technology uses digital image processing and pattern recognition techniques to identify people based on the texture of their iris. This method is frequently employed in high-security applications for a variety of reasons, the most significant of which are: First off, these extremely intricate and distinct patterns which have over 200 distinctive points, as opposed to roughly 40 for fingerprints are what give the iris its uniqueness and high precision. Because of this, iris recognition is very dependable and impervious to false matches. The second is stability throughout time, and this characteristic Long-term accuracy is ensured by the iris pattern's consistency throughout a person's life, in contrast to facial characteristics or fingerprints, which might alter over time owing to wear or age. Thirdly, the speed and efficiency; Iris recognition systems are appropriate for high-throughput applications like public gatherings or airport security since they can confirm IDs in a matter of seconds. Lastly, this technology is hygienic because it only requires a fast look at a camera from humans, removing the need for physical touch [10, 11].

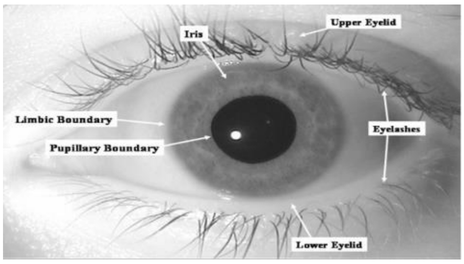


Figure 1. Illustration of the various components of the eye

Image capture, feature extraction, and encoding and matching are the three fundamental phases of the iris recognition system. Iris recognition is a popular option for many applications in a variety of industries [12]. The technique of determining the exact borders of the iris in an image is known as "iris localization". A few of these are described in Figure 2.

Border Control
Banking and Financial Services
Healthcare and Patient Identification
Mobile Authentication
Gaming and Entertainment

Figure 2. Some iris recognition application domains

Due to its great precision and uniqueness, iris localization is essential component of biometric recognition system. However, a number of issues, including motion blur, dim lighting, occlusions, limited resolution, and sensor noise, frequently result in lower-quality eye photos in real-world situations. A critical step before iris recognition, precisely iris localization, is made extremely difficult by these circumstances. In order to provide high reliability, speed up the automatic iris_ROI segmentation process and performance in imaging conditions, the main objective of this paper is to introduce a robust and automatic iris localization framework that combines the AdaBoost algorithm with a Haar Cascade Classifier (HCC) that can precisely identify the iris region in noisy, low-quality, and unconstrained real-world eye images. This iris_ROI is difficult, time-consuming, inconsistent, and prone to errors when extracted manually [13].

The main advantage of the introduced framework stems from the features of the hierarchical detection framework-based HCC. Quick removal of irrelevant areas, such as the background, eyelashes, and eyebrows, is the main goal of the early stages. To guarantee precise iris_ROI localization, additional thorough tests are carried out in later phases. This speeds up localization and lowers computational overhead. Real-time or large-scale applications benefit from the computational efficiency of calculating Haar features, which are based on rectangular intensity contrasts, utilizing integral summed-area tables. AdaBoost highlights unique iris patterns that set the eye or face apart from other areas. To improve localization accuracy, it iteratively modifies weights to concentrate on regions that are challenging to classify. The framework can adapt to changes in eye size and image resolution since the introduced algorithm can withstand lighting variations and detect irises at different scales.

This paper is divided into the following sections: Section 2 provides an overview of the literature on various iris localization and segmentation research. Section 3 describes the techniques used in this paper, and Section 4 provides specifics on the introduced framework. The results and conclusions are presented in Sections 5 and 6.

2. LITERATURE REVIEW

In order to identify and locate irises for iris recognition systems, a wide variety of research investigations have been carried out. The goal is to make identification more accurate in a range of situations and lighting conditions.

An effective iris recognition system based on Integrated Wavelet Transform (IWT) has been proposed [14]. Iris localization and segmentation phases were conducted using the Circular Hough Transform (CHT) and Total Variation Model approaches (TVM). In order to extract features, the segmented image was broken down into sub-band images using IWT. The matching criterion was the normalized Hamming distance. The results of experiments show that the suggested algorithm can improve performance in controlled conditions and efficiently distinguish individuals by recognizing their irises.

An iris segmentation technique for photos taken with near-infrared (NIR) illuminators is introduced, detailing how the scheme operates [15]. To lessen the impact of the input eye image's abrupt changes in grayscale intensity, it first applies a rank filter. It then marks the pupillary boundary with a circle approximation using an improved coarse-to-fine approach that includes adaptive thresholding, histograms, and 2D geometry. In order to avoid this interference, it then uses the inverted gray-level intensity of the pupil region to identify the low-intensity areas of an image. Lastly, eyelashes and reflections in the polar shape of the iris are also marked by the presented system. Three public databases CASIA-Iris-Interval V3.0, IITD V1.0, and MMU V1.0 are used to validate the presented technique.

An iris recognition method for human identification is proposed, employing two-dimensional principal component analysis (2DPCA) and genetic algorithms (GA) to reduce the dimensionality of iris features [16]. The Back Propagation Neural Network (BPNN), which employs Levenberg-Marquardt's learning rule, was employed for the classification stage. They tested the proposed approach on the CASIA iris picture database, which consists of 2655 iris photos from 249 individuals. The 2DPCA-GA achieves a 96.40% classification accuracy.

Iris localization scheme [17] involves preprocessing the input eye image with an order statistic filter and the bilinear interpolation scheme. Using the image's histogram to extract an adaptive threshold. Processing the binary image using morphological operators, determining the center and radius of the pupil using centroid and geometry concepts. Using the CHT to mark the iris outer boundary, and using the Fourier series to refine coarse iris boundaries. They used three iris image datasets to test the proposed scheme: MMU V1.0, IITD V1.0, and CASIA. The scheme that is being given achieves an average accuracy of 99.34.

In this study [18], a hybrid biometric model that uses convolutional neural networks (CNNs) and Hamming distances (HD) for feature extraction and classification along with edge detection and segmentation has been presented. The CASIA-Iris-Interval V4 dataset, IITD dataset, and MMU dataset are the three datasets to which the developed model is applied. The biometric model that was presented attained accuracy levels of 94.88% when using HD on CASIA, 96.56% when using CNN on IITD, and 98.01% when using CNN on MMU.

3. RELATED METHODOLOGIES

The background theories for the main techniques employed in this paper are illustrated in this section.

3.1 Image normalization

Use the histogram equalization (HE) technique to enhance lighting and contrast. This is a method of image processing that involves shifting the image's intensity levels in order to increase contrast. HE can be applied to each color channel separately for color images, which usually contain three color channels (red, green, and blue). Stretching the pixel intensity range is intended to create a more even distribution of pixel intensities in the image. The Cumulative Distribution Function (CDF) is calculated in order to implement it. With CDF, the probability intensity value of a pixel in the image that is less than or equal to a threshold value is calculated, the histogram of pixel intensities is normalized, and the cumulative sum of the probability intensity values is then calculated. The CDF is used to map the pixel intensities to new values after it has been calculated. The contrast is improved by applying the modification in a way that makes the pixel intensity values more dispersed. Eqs. (1) and (2) provide the mathematical definition of the CDF [19-21]:

$$CDF(r_k) = \sum_{i=0}^{k} h(ri) \tag{1}$$

$$r_k = \frac{CDF\ (r_k) - CDF_{min}}{N - CDF_{min}} \times (\text{L-1}) \tag{2}$$

where,

$h(ri)$: It is a histogram of the pixel intensity.

r_k: It is a specific intensity level.

CDF_{min} : It is the minimum value of CDF.

N: It is the image's total amount of pixels.

L-1: is the maximum pixel's intensity for an 8-bit image, *L*=256.

3.2 Noise reduction

One of the most popular filters for image processing noise reduction is the Gaussian Filter (GF). It is favored because of a number of significant characteristics and benefits that make it useful and efficient for this task: Isotropic filtering, smooth and continuous weight distribution, parameter control using σ, edge preservation in comparison to simple averaging, in addition to mathematical simplicity and efficiency, are the concluding points. It is a linear filter that gives weights to nearby pixels in a picture using the Gaussian function. The degree to which each nearby pixel will affect the core pixel throughout the filtering process is determined by the Gaussian function. The weights of this function are the same in all directions since it is symmetric around the origin. No matter where, a pixel is in relation to the filter's center, this symmetry guarantees that it is weighted equally. For the majority of image sizes, the GF is computationally efficient and reasonably simple to use. Eq. (3) provides the mathematical definition of the Gaussian function, from which the filter utilized in the GF is generated [22, 23]:

$$G(n,m) = \frac{1}{2\Pi\sigma^2} \exp\left(-\frac{n^2+m^2}{2\sigma^2}\right) \tag{3}$$

where,

G(*n*,*m*): In relation to the center, it is the value of the Gaussian function at the point (*n*,*m*).

σ: It is the standard deviation of the Gaussian distribution, which determines the width of the Gaussian bell curve. A larger σ results in a wider kernel and more smoothing, while a smaller *σ* results in a sharper filter with less smoothing.

x and *y*: are the pixel positions relative to the center of the filter.

The choice of the standard deviation σ (sigma) is important since it determines how much smoothing (blurring) is done when employing a Gaussian filter to replicate or counteract motion blur in the context of iris localization in real-world noisy images. When using a Gaussian filter to handle motion blur in iris detection tasks, the value of *σ*=2 provides a suitable trade-off between feature preservation and noise reduction. It is a reasonable and useful default setting for preprocessing in such applications since it is computationally effective, robust under a variety of real-world image situations, and supported by empirical evidence.

3.3 HCC

HCC is a feature-based approach to object recognition in photos. Numerous positive and negative photos are used to train a cascade function for detection. This method doesn't need a lot of computing and can work in real time. For bespoke items like automobiles, bikes, animals, etc., it can be trained on particular cascade functions. As illustrated in Figure 3, Haar Cascade makes use of the cascade function and cascading window. It attempts to categorize good and negative qualities for each window. Positive if the window is part of an item; negative otherwise. A sizable collection of both positive and negative face data is used to train the classifier. Weak classifiers are combined into a strong classifier using the AdaBoost method. It works well for object detection in static or controlled situations, is quick and simple to develop, and can operate in real-time [24, 25].

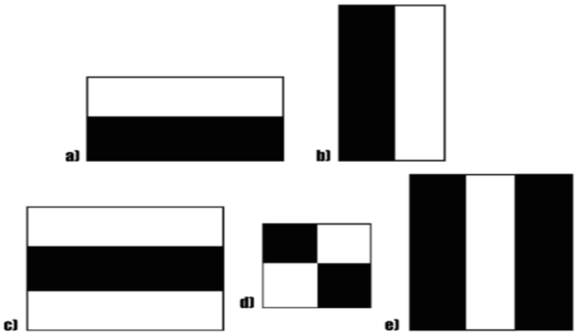


Figure 3. Different windows of HCC

4. THE INTRODUCED FRAMEWORK

The iris localization algorithm introduced in this paper comprises these subsequent phases:

4.1 Preprocessing phase

This phase's goal is to improve the input image by reducing noise and increasing contrast. It involves the following two steps:

4.1.1 Image normalization

The HE process is apply to each color channel separately for color images, which normally contain three color channels (Red, Green, and Blue). Three sub-steps precede this step: The input color image should first be separated into its R, G, and B color channels. In essence, each channel is a grayscale image with pixel intensities that match the corresponding color component. Second, provide each channel its own application of the HE process. This entails separately calculating the R, G, and B channels' histogram, CDF, and mapping. Lastly, merge the R, G, and B channels once more to create a single color image, as shown in Figure 4's implementation example and Algorithm 1.

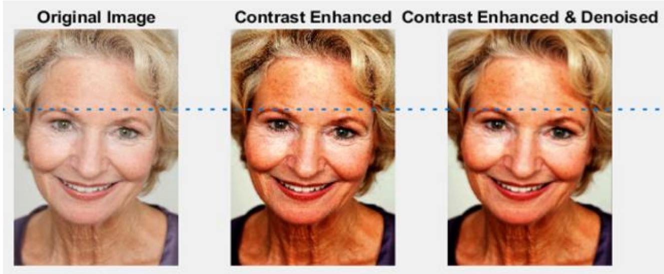


Figure 4. Preprocessing phase

```
Algorithm 1. Image normalization
  Input: color_image (3D matrix: Height×Width×Channels)
  Output: equalized_ color image
  sub-step1: Separate the Red, Green, and Blue (R, G, and B) channels from the color image.
  sub-step2: For each channel (R, G, B):
      a). Flatten the channel into a 1D array.
      b). Calculate the histogram of the flattened channel.
      c). Compute the cumulative distribution function (CDF) from the histogram.
      d). Normalize the CDF to map it to the range [0, 255].
      e). Use the normalized CDF to map the original pixel values to new equalized values.
      f). Reshape the equalized channel back to its original shape.
  sub-step3: Combine the equalized R, G, and B channels into a single color image.
```

4.1.2 Noise reduction

In order to smooth the equalized_image result from the previous step and eliminate noise, the GF is applied in this phase. It uses a GF to convolve the image. By slightly blurring the image, the filter lowers high-frequency noise. Usually, the GF is applied separately to the R, G, and B color channels for the color images. Three sub-steps can be used to summarize the overall process: The color image should first be separated into its R, G, and B. Second, apply the GF separately for R, G, and B color channels. A smoothed version of each channel is the end result. Lastly, merge the transformed channels back into a single image after filtering each one independently. A color image with less noise will be the end outcome. The smoothing level was controlled by setting the sigma to (2). The Algorithm 2 explains these three sub-steps, and Figure 4 shows the outcome of the implementation.

```
Algorithm 2. Noise reduction
  Input: equalized_color image
  Output: denoised_image
  sub-step1: Define GF for a chosen filter size (3×3).
  For a 3×3 Gaussian kernel:
  G=[1/16, 1/8, 1/16]
     [1/8,   1/4,   1/8]
     [1/16, 1/8, 1/16]
  sub-step2: For each channel (R, G, B) in equalized_image apply the GF to the channel:
      a). For each pixel (i, j) in the channel:
          (1). Extract the surrounding neighborhood of size kernel_size (3×3).
          (2). Multiply the pixel values in the neighborhood by the corresponding values in the GF.
          (3). Sum the weighted values and assign this value to the new pixel at position (i, j).
      b). Repeat for all pixels in the channel.
  sub-step3: Combine the denoised R, G, and B channels into a single denoised color image.
```

4.2 Face localization phase

Finding the face's location and size within the image is the goal of this phase. These sub-steps are used to carry out this phase: To begin with, the HCC is trained to recognize face. Secondly, the input image converted to grayscale. Thirdly, face in the image is found using the detectMultiScale method. Lastly, a bounding box containing *x* and *y* for the width and height is used to represent the discovered face. The Algorithm 3 explains these processes, and Figure 5 shows the outcome of the implementation.

```
Algorithm 3. Face localization phase
  Input: denoised_image
  Output: face_detected (bounding box)
  sub-step1: Load the pre-trained HCC for face detection:
  haar_cascade_path='haarcascade_frontalface_default.xml'
  face_cascade=LoadHaarCascade(haar_cascade_path)
  sub-step2: Convert the input denoised_image to grayscale:
  grayscale_image=ConvertToGrayscale(image)
  sub-step3: Detect face in the grayscale image using the HCC:
  face=face_cascade.detectMultiScale(grayscale_image, scaleFactor=1.1, minNeighbors=5, minSize=(30, 30))
  sub-step4: Initialize an empty list to store the bounding box of detected face: face_detected=[]
  sub-step5: For detected face (bounding box):
  Extract the coordinates of the bounding box (x, y, width, height).
```

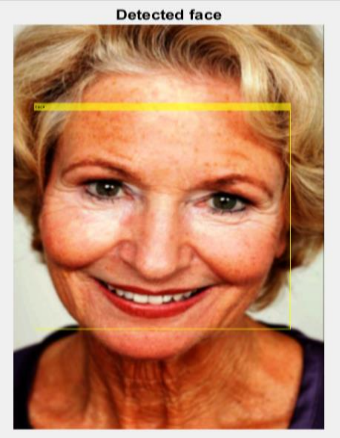


Figure 5. Face localization phase

4.3 Eyes localization phase

The objective of this phase is to identify the eyes' location and size within the face image. These sub-steps are used to carry out this phase: First, training the HCC for detect eyes. Second, apply the detectMultiScale method once again inside the cropped facial region to detect the eyeballs. A bounding box with the coordinates (x_eye, y_eye, width_eye, height_eye) represents each detected eye. Lastly, a list of bounding boxes for each identified eye is produced after the eye coordinates have been modified in relation to the face's location in the original image. The Algorithm 4 explains these processes, and Figure 6 shows the outcome of the implementation.

Algorithm 4. Eyes localization phase

```
    Input: face_detected (bounding box)
    Output: eyes_detected (List of bounding boxes)
    sub-step1: Trained HCC for eye detection:
    haar_cascade_path='haarcascade_eye.xml'
    eye_cascade=LoadHaarCascade(haar_cascade_path)
    sub-step2: apply the detectMultiScale method in the face bounding box:
    roi_gray=grayscale_image[y:y+height, x:x+width]
    eyes=eye_cascade.detectMultiScale(roi_gray, scaleFactor=1.1, minNeighbors=3, minSize=(20, 20))
    sub-step3: For each detected eye in the ROI:
    1). Extract the coordinates of the eye bounding box (x_eye, y_eye, width_eye, height_eye).
    2). Adjust the coordinates relative to the original face bounding box.
    3). Append the adjusted eye bounding box (x_eye, y_eye, width_eye, height_eye) to the eyes_detected list.
```

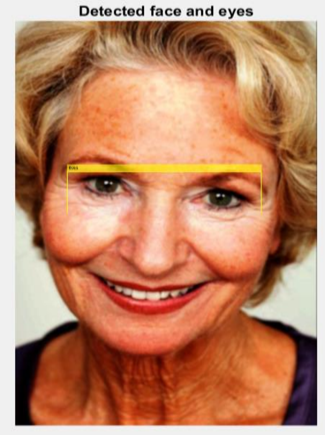


Figure 6. Eyes localization phase

4.4 Eyes segmentation phase

This phase's goal is to cut off the eyes_ROI from the rest of the face by using morphological procedures and edge detection thresholding. Last but not least, restore the ocular ROI to its initial color and binary state. The Algorithm 5 explains these processes, and Figure 7 shows the outcome of the implementation.

Algorithm 5. Eyes segmentation phase

```
    Input: eyes_detected (List of bounding boxes)
    Output: binary_eye_image (2D matrix: Height x Width)
    sub-step1: Threshold the eye region to create a binary image:
    binary_eye_region=ApplyThreshold(eye_roi)
    sub-step2: Apply morphological operations (dilation/erosion) to clean the binary image:
    binary_eye_region=MorphologicalOperations(binary_eye_region)
    sub-step3: Replace the eye region in the original image with the binary_eye_region:
    binary_eye_image=ReplaceRegionWithBinary(image, eye_roi, binary_eye_region)
```

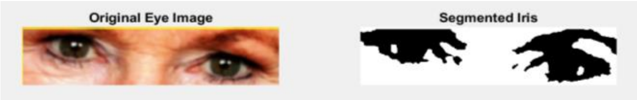


Figure 7. Iris segmentation phase

4.5 Iris and pupil localization phase

Finding the location of each iris and pupil region in an eye image is the goal of this phase. The algorithm goes through the following sub-steps to accomplish this goal: Firstly, the first step in identifying the iris and pupil is determining the boundary. To find the circles in the image, the Hough Circle Transform (HCT) technique is used. Secondly, determine the minimum and maximum radius values, which are determined by the anticipated sizes of the image's iris and pupil. Thirdly, determining the pupil by assuming that the pupil is represented by the smallest detected circle. The pupil is usually smaller than the iris, which explains this. Lastly, since the iris is typically larger than the pupil, identify it by finding the second-smallest circle, which is thought to be the iris. The Algorithm 6 explains these processes, and Figure 8 shows the outcome of the implementation.

Algorithm 6. Iris and pupil localization phase

```
    Input: binary-eye-image
    Output: pupil_center (x_pupil, y_pupil), iris_center (x_iris, y_iris), pupil_radius, iris_radius
    sub-step1: Detect the outer boundary of the iris and pupil using the HCT to detect circles in the binary image (which should correspond to the eye contour):
    circles = HoughCircleDetect(binary_image, min_radius=20, max_radius=50)
    The `min_radius` and `max_radius` are based on expected sizes of the pupil and iris.
    sub-step2: Identify the pupil:
    Sort the detected circles by radius and identify the smallest circle, which corresponds to the pupil.
    Assign the center and radius of the smallest circle as:
    (x_pupil, y_pupil), pupil_radius=smallest_circle
    sub-step3: Identify the iris:
    From the remaining detected circles, choose the next smallest circle, which corresponds to the iris.
    Assign the center and radius of this circle as:
    (x_iris, y_iris), iris_radius=second_smallest_circle
    sub-step3: Post-process the results:
    If no valid circles are detected, return error message.
```

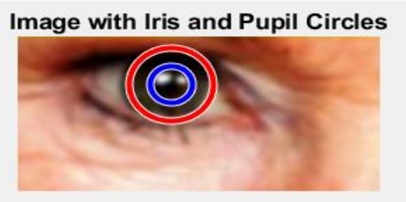


Figure 8. Iris and pupil localization phase

5. EXPERIMENTS' OUTCOMES

With MATLAB Version (R2023b), the introduced framework has been put into practice. Windows 10 was utilized, along with an Intel Core i9 CPU and 12GB of RAM. Three datasets have been used to validate the work. The first dataset consists of 180 contactless color face photos were gathered from Labeled Faces in the Wild (LFW). Some face images have a black background, while others have a gray, sunny, or white background. More than 13,000 color photos of faces that were gathered from the internet make up this collection. This set includes distracting elements like contact lenses, low contrast, uneven lighting, eyelids, eyebrows, eyelashes, and specular reflections. The second is the MMU iris database, which was made by Malaysia's Multimedia University (MMU) mainly for iris recognition system research. MMU includes 460 grayscale iris images for 46 people, with 5 images for each of the left and right eyes. The photos were taken under controlled lighting. Thirdly, the Iris Super Resolution (ISR) dataset, which contains 4,320 color iris photos from 704 subjects, 392 of them are female and 312 of whom are male university students. It offers high-quality photos taken in controlled settings; each student's left and right eyes were used for more than six photos. The factors described in Table 1 were taken into account when determining each of the metrics for the evaluation of the introduced framework, as demonstrated in Eqs. (4)-(6): specificity, which measures the proportion of actual negatives that are correctly identified. Accuracy, which measures the framework's overall correctness, and sensitivities, which measure the proportion of actual positives that are correctly identified [26-28].

Table 1. Definition used in evaluation the introduced framework

Evaluation Metric Name	Evaluation Mean
TP	Iris exist with correct-localization
TN	Iris not exist with non-localize
FP	Non- iris exist with localized it
FN	Iris exist with non-localize

$$\text{Sensitivity} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)+False Negative (FN)}} \times 100\% \tag{4}$$

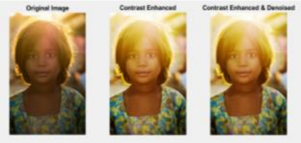
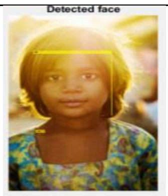
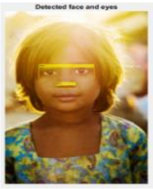
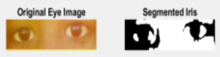
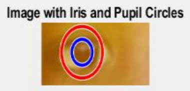
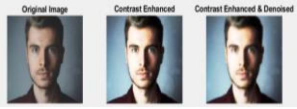
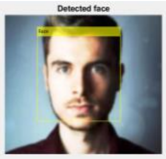
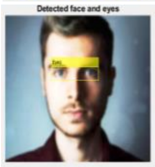
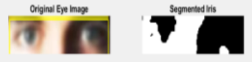
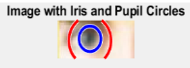
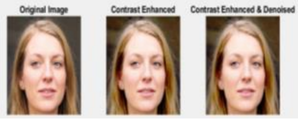
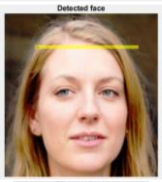
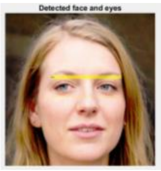
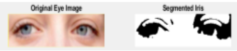
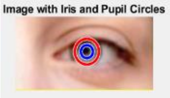
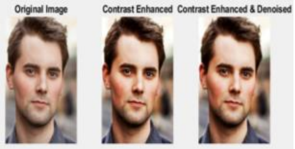
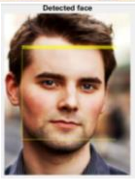
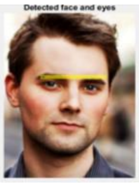
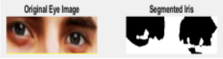
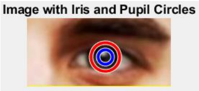
$$\text{Specificity} = \frac{\text{True Negatives (TN)}}{\text{True Negatives (TN)+False Positive (FP)}} \times 100\% \tag{5}$$

$$\text{Accuracy} = \frac{\text{True Positives (TP)+True Negatives (TN)}}{\text{Total (TP+TN+FP+FN)}} \times 100\% \tag{6}$$

Every image in LFW has a fixed size of 250×250 pixels and is in the JPEG format; some of the images are not completely focused or have a small amount of motion blur. Since LFW was gathered in unrestricted settings, it inherently includes noise and variability from the real world, which is useful for robustness testing. The three main categories of noise in LFW are expression variability, background clutter, and illumination variations. Gaussian and salt-and-pepper noise are examples of synthetic noise that are not specifically included in the LFW dataset. However, in order to assess the resilience of introduced framework, it is frequently utilized in tests with additional synthetic noise. For the MMU Iris dataset, the image is in grayscale BMP format and has a resolution of 320×240 pixels. By default, MMU does not include artificial noise such as Gaussian or salt-and-pepper. In experiments, they are usually manually added to replicate difficult conditions. Images used in the ISR dataset are usually 320 by 280 pixels in size. There may have been motion blur and natural differences because the photos were taken in a variety of settings, such as with varying lighting and subject movement.

Indeed, the performance of an iris localization framework using AdaBoost with HCC can be impacted by the color of the eyes; for instance, dark brown irises are easier to identify with circular dark features. For light brown, its detection reliability is rated as average. Insufficient edge strength leads to poorer blue, green, and gray detection. Due to that, we employ contrast enhancement throughout the preprocessing phase.

Table 2. Some instances from the LFW dataset along with their outcomes after applying the introduced algorithm

Preprocessing Phase	Face Localization Phase	Eyes Localization Phase	Segmentation Phase	Iris and Pupil Localization Phase
 Original Image Contrast Enhanced Contrast Enhanced & Denoised				
 Original Image Contrast Enhanced Contrast Enhanced & Denoised				
 Original Image Contrast Enhanced Contrast Enhanced & Denoised				
 Original Image Contrast Enhanced Contrast Enhanced & Denoised				

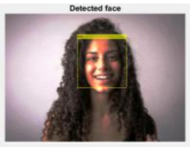
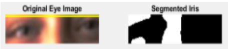
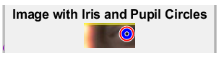
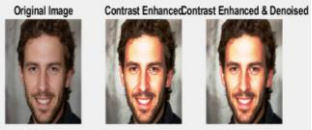
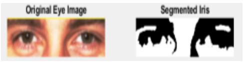
Original Image Contrast Enhanced Contrast Enhanced & Denoised				
 Original Image Contrast Enhanced Contrast Enhanced & Denoised				

Table 3. Some instances from MMU dataset along with their outcomes after applying the introduced algorithm

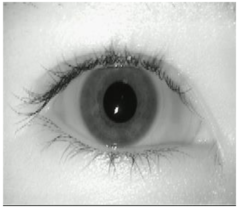
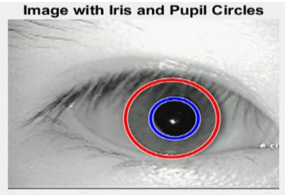
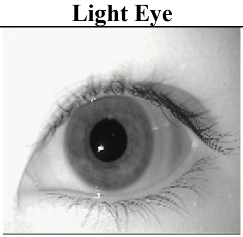
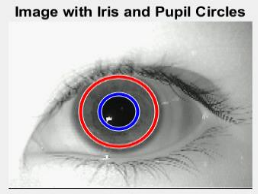
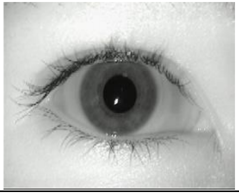
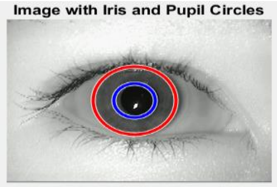
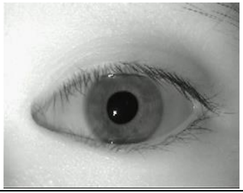
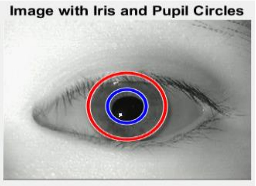
Right Eye	Iris and Pupil Localization Phase	Light Eye	Iris and Pupil Localization Phase
			
			

Table 4. Some instances from the ISR dataset along with their outcomes after applying the introduced algorithm

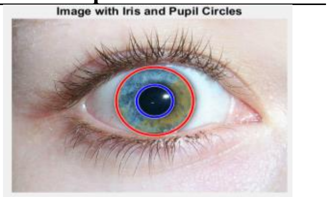
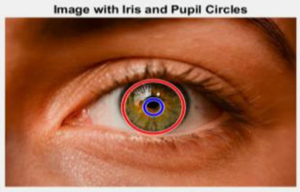
Eye Instance	Iris and Pupil Localization Phase
	
	

Table 5. The Evaluation of introduced algorithm

Dataset	Total Images Tested	SN	TP	TN	FP	FN	Sensitivity	Specificity	Accuracy
LFW	180	40%	176	2	0	2	98.87%	100%	98.88%
MMU	300	30%	298	0	0	2	99.33%	0	99.33%
ISR	220	40%	220	0	0	0	100%	0	100%

Table 6. Comparison of the introduced algorithm with other works

Related Works	Method	Accuracy
Singh et al. [14]	Using characteristics taken from the Integer Wavelet Transform Level (IWT).	98.9%
Jan and Min-Allah [15]	The pupillary boundary is marked with a circle approximation using an improved coarse-to-fine approach that incorporates adaptive thresholding, histogram, and 2D geometry, along with a rank filter.	97.86%
Garg et al. [16]	The dimensionality of iris characteristics has been decreased by the application of feature extraction and selection techniques such as GA (Genetic Algorithm) and 2DPCA (two-dimensional Principal Component Analysis). Levenberg-Marquardt's learning rule is used to create the Back Propagation Neural Network (BPNN).	96.40%
Jan et al. [17]	Convolutional neural network layers are used to extract features, and a softmax layer is used to categorize these features into N groups so that CNN can be trained. The weights and learning rate are updated using the Adam optimization technique and the backpropagation algorithm.	97.32%

[29-31]	AdaBoost based on Haar is the ideal method for low-resolution images for three reasons: it is quick, lightweight, and interpretable. Deep learning techniques like CNN or U-Net are the best for high-resolution photos because they offer superior generalization.	~91-98% (with enough training data)
Introduced Framework	Using LFW dataset	98.88%
	Using MMU dataset	99.33%
	Using ISR dataset	100%

The outcomes of three dataset instances after applying the introduced algorithm are shown in Tables 2-4. Table 5 provides an explanation of the sensitivity, specificity, and accuracy of the tested images for each dataset. The sample counts per dataset with synthetic noise was represented by (SN). The introduced algorithm outperforms the previous works that were compared with it, as Table 6 illustrates.

6. CONCLUSIONS

In conclusion, this paper introduced a robust iris localization and extraction algorithm. Accurately defining the iris outlines and detecting and eliminating noise from the legitimate iris portion are critical to the overall effectiveness of any iris recognition system. The precision and speed of an iris localization system are crucial. The following is how the introduced algorithm locates iris boundaries. First, use the HE and GF preprocessing steps to enhance the image's contrast and minimize noise. Second, it used the AdaBoost algorithm and the HCC to locate the face and eyes in the image. Localizing the face's position and size is the goal of this step. The HCC is then used to localize the eyeballs in the face image. Through the segmentation process, it uses edge detection thresholding to eliminate all pixels that are part of the ocular object's interior and/or exterior for speedy processing. The entire boundary localization procedure is significantly enhanced by this action. Lastly, it uses the HCT to extract the actual pupil and iris contours, binarizing the segmented portion and identifying the contours of the largest and smallest circular regions. AdaBoost and Haar features work together to create a quick and lightweight method that is appropriate for real-time applications. In addition to low and high image resolutions, it works well with both light and dark images. The described framework exhibits great generalization across a wide range of eye forms, sizes, colors, and orientations following appropriate training. Three datasets of face and eye images-the LFW, MMU, and ISR-were used to validate the introduced framework. When compared to manystate-of-the-art iris localization techniques, experimental findings from these databases are comparatively superior. We intend to use our framework for autofocusing iris cameras and integrate with raspberry Pi for edge deployment and evaluate latency.

ACKNOWLEDGMENT

The authors are grateful to Mustansiriyah University in Baghdad, Iraq (www.uomustansiriyah.edu.iq) for their cooperation with this work.

REFERENCES

[1] Butt, M.A., Qayyum, A., Ali, H., Al-Fuqaha, A., Qadir, J. (2023). Towards secure private and trustworthy human-centric embedded machine learning: An emotion-Aware facial recognition case study. Computers & Security, 125: 103058. https://doi.org/10.1016/j.cose.2022.103058

[2] Hussain, S.A.K., Al-Nayyef, H., Al Kindy, B., Qassir, S.A. (2023). Human earprint detection based on ant colony algorithm. International Journal of Intelligent Systems and Applications in Engineering, 11(2): 513-517.

[3] Yousif, N.A., Qassir, S.A., George, D.N. (2025). Robust and automatic algorithm for palmprint ROI extraction. JOIV: International Journal on Informatics Visualization, 9(1): 433-438. https://doi.org/10.62527/joiv.9.1.2801

[4] Naser, Z.S., Khalid, H.N., Ahmed, A.S., Taha, M.S., Hashim, M.M. (2023). Artificial neural network-based fingerprint classification and recognition. Revue d'Intelligence Artificielle, 37(1). https://doi.org/10.18280/ria.370116

[5] Alrawili, R., AlQahtani, A.A.S., Khan, M.K. (2024). Comprehensive survey: Biometric user authentication application, evaluation, and discussion. Computers and Electrical Engineering, 119: 109485. https://doi.org/10.1016/j.compeleceng.2024.109485

[6] Stylios, I., Kokolakis, S., Thanou, O., Chatzis, S. (2021). Behavioral biometrics & continuous user authentication on mobile devices: A survey. Information Fusion, 66: 76-99. https://doi.org/10.1016/j.inffus.2020.08.021

[7] Ahmed, A.S., Abdullatif, F.A., Hasan, T.M. (2019). Generating and validating DSA private keys from online face images for digital signatures. International Journal on Advanced Science Engineering Information Technology, 9(3): 993-998.

[8] Azam, M.S., Rana, H.K. (2020). Iris recognition using convolutional neural network. International Journal of Computer Applications, 175(12): 24-28. https://doi.org/10.5120/ijca2020920602

[9] Bhatt, S., Sehrawat, J.S., Gupta, V. (2025). A systematic review of iris biometrics in forensic science: Applications and challenges. Egyptian Journal of Forensic Sciences, 15: 12. https://doi.org/10.1186/s41935-025-00431-7

[10] Sharma, G., Tandon, A., Jaswal, G., Nigam, A., Ramachandra, R. (2024). Impact of iris pigmentation on performance bias in visible iris verification systems: A comparative study. arXiv Preprint arXiv: 2411.08490. https://doi.org/10.48550/arXiv.2411.08490

[11] Harikrishnan D., Sunilkumar, N., Shelby, J., Kishor, N., Remya, G. (2023). An effective authentication scheme for a secured IRIS recognition system based on a novel encoding technique. Measurement: Sensors, 25: 100626. https://doi.org/10.1016/j.measen.2022.100626

[12] El-Sofany, H., Bouallegue, B., Abd El-Latif, Y.M. (2024). A proposed biometric authentication hybrid approach using iris recognition for improving cloud security. Heliyon, 10(16). https://doi.org/10.1016/j.heliyon.2024.e36390

[13] Benalcazar, D.P., Tapia, J.E., Vasquez, M., Causa, L., Droguett, E.L., Busch, C. (2023). Toward an efficient iris

recognition system on embedded devices. IEEE Access, 11: 133577-133590. https://doi.org/10.1109/ACCESS.2023.3337033

[14] Singh, G., Singh, R.K., Saha, R., Agarwal, N. (2020). IWT based iris recognition for image authentication. Procedia Computer Science, 171: 1868-1876. https://doi.org/10.1016/j.procs.2020.04.200

[15] Jan, F., Min-Allah, N. (2020). An effective iris segmentation scheme for noisy images. Biocybernetics and Biomedical Engineering, 40(3): 1064-1080. https://doi.org/10.1016/j.bbe.2020.06.002

[16] Garg, M., Arora, A., Gupta, S. (2021). An efficient human identification through iris recognition system. Journal of Signal Processing Systems, 93(6): 701-708. https://doi.org/10.1007/s11265-021-01646-2

[17] Jan, F., Min-Allah, N., Agha, S., Usman, I., Khan, I. (2021). A robust iris localization scheme for the iris recognition. Multimedia Tools and Applications, 80: 4579-4605. https://doi.org/10.1007/s11042-020-09814-5

[18] Farouk, R.H., Mohsen, H., El-Latif, Y.M.A. (2022). A proposed biometric technique for improving iris recognition. International Journal of Computational Intelligence Systems, 15(1): 79. https://doi.org/10.1007/s44196-022-00135-z

[19] Rahman, H., Paul, G.C. (2023). Tripartite sub-image histogram equalization for slightly low contrast gray-tone image enhancement. Pattern Recognition, 134: 109043. https://doi.org/10.1016/j.patcog.2022.109043

[20] Jebadass, J.R., Balasubramaniam, P. (2024). Color image enhancement technique based on interval-valued intuitionistic fuzzy set. Information Sciences, 653: 119811. https://doi.org/10.1016/j.ins.2023.119811

[21] Abbas, A.H., Mirza, N.M., Qassir, S.A., Abbas, L.H. (2020). Maize leaf images segmentation using color threshold and K-Means clustering methods to identify the percentage of the affected areas. IOP Conference Series: Materials Science and Engineering, 745(1): 012048. https://doi.org/10.1088/1757-899X/745/1/012048

[22] Goceri, E. (2023). Evaluation of denoising techniques to remove speckle and Gaussian noise from dermoscopy images. Computers in Biology and Medicine, 152: 106474. https://doi.org/10.1016/j.compbiomed.2022.106474

[23] Abuhayi, B.M., Mossa, A.A. (2023). Coffee disease classification using convolutional neural network based on feature concatenation. Informatics in Medicine Unlocked, 39: 101245. https://doi.org/10.1016/j.imu.2023.101245

[24] Bhargava, N., Rathore, P.S., Jha, M., Goswami, A. (2023). Enhancing facial recognition and tracking for security applications using Haar cascade classifier. In 2023 2nd International Conference on Automation, Computing and Renewable Systems (ICACRS), Pudukkottai, India, pp. 601-606. https://doi.org/10.1109/ICACRS58579.2023.10404668

[25] Meddeb, H., Abdellaoui, Z., Houaidi, F. (2023). Development of surveillance robot based on face recognition using Raspberry-PI and IOT. Microprocessors and Microsystems, 96: 104728. https://doi.org/10.1016/j.micpro.2022.104728

[26] Shan, W., Li, D., Liu, S., Song, M., Xiao, S., Zhang, H. (2024). A random feature mapping method based on the AdaBoost algorithm and results fusion for enhancing classification performance. Expert Systems with Applications, 256: 124902. https://doi.org/10.1016/j.eswa.2024.124902

[27] Yaacob, R., Ooi, C.D., Ibrahim, H., Nik Hassan, N.F., Othman, P.J., Hadi, H. (2019). Automatic extraction of two regions of creases from palmprint images for biometric identification. Journal of Sensors, 2019(1): 5128062. https://doi.org/10.1155/2019/5128062

[28] Van Stralen, K.J., Stel, V.S., Reitsma, J.B., Dekker, F.W., Zoccali, C., Jager, K.J. (2009). Diagnostic methods I: Sensitivity, specificity, and other measures of accuracy. Kidney International, 75(12): 1257-1263. https://doi.org/10.1038/ki.2009.92

[29] Lin, Y.N., Hsieh, T.Y., Huang, J.J., Yang, C.Y., Shen, V.R., Bui, H.H. (2020). Fast Iris localization using Haar-Like features and AdaBoost algorithm. Multimedia Tools and Applications, 79: 34339-34362. https://doi.org/10.1007/s11042-020-08907-5

[30] Boyd, A., Czajka, A., Bowyer, K. (2019). Deep learning-Based feature extraction in iris recognition: Use existing models, fine-Tune or train from scratch? In 2019 IEEE 10th International Conference on Biometrics Theory, Applications and Systems (BTAS), Tampa, FL, USA, pp. 1-9. https://doi.org/10.1109/BTAS46853.2019.9185978

[31] Nguyen, K., Proença, H., Alonso-Fernandez, F. (2024). Deep learning for iris recognition: A survey. ACM Computing Surveys, 56(9): 223. https://doi.org/10.1145/3651306