Amal Hameed Khaleel*, Thekra H. Abbas and Abdul-Wahab Sami Ibrahim

Best low-cost methods for real-time detection of the eye and gaze tracking

https://doi.org/10.1515/icom-2023-0026
Received September 21, 2023; accepted December 20, 2023;
published online January 8, 2024

Abstract: The study of gaze tracking is a significant research area in computer vision. It focuses on real-world applications and the interface between humans and computers. Recently, new eye-tracking applications have boosted the need for low-cost methods. The eye region is a crucial aspect of tracking the direction of the gaze. In this paper, several new methods have been proposed for eye-tracking by using methods to determine the eye area as well as find the direction of gaze. Unmodified webcams can be used for eye-tracking without the need for specialized equipment or software. Two methods for determining the eye region were used: facial landmarks or the Haar cascade technique. Moreover, the direct method, based on the convolutional neural network model, and the engineering method, based on distances determining the iris region, were used to determine the eye's direction. The paper uses two engineering techniques: drawing perpendicular lines on the iris region to identify the gaze direction junction point and dividing the eye region into five regions, with the blackest region representing the gaze direction. The proposed network model has proven effective in determining the eye's gaze direction within limited mobility, while engineering methods improve their effectiveness in wide mobility.

Keywords: computer vision; human computer interaction; deep learning; eye gaze tracking; iris landmarks

1 Introduction

Eye detection and tracking are essential aspects of the interaction between humans and computers that have recently developed. The applications of virtual reality, augmented reality, consumer behavior detection, computer control, identification, medical treatment systems, and security programs have all benefited from research in this field (*1*).

Among all the characteristics of the face, the eyes are the most noticeable and stable (*2*). Eye detection remains challenging due to variations in eye appearance, occlusion, and external noise. Factors like iris color, size, and shape influence eye appearance. Occlusion, a white reflection in images, is caused by camera illumination. External noise, such as glasses and lighting, also affects eye features (*3*).

The eyes have the potential to be an effective nonverbal communication tool. Eye gaze monitoring may give more accurate information about user activities, such as the location of a person's gaze, the amount of time spent gazing, and so on (*4*). The practice of monitoring eye activity is known as eye gaze tracking (EGT). It is generally used to measure an individual's focus of attention. Eye gaze analysis may also aid in the comprehension of human behavior, attention, and other cognitive functions (*5*). EGT might also be used as an interface for people with impairments. It might let people control a computer gadget with their eyes. EGT technology has emerged as a replacement for numerous input devices such as the mouse and touch screen, among others (*6*).

Eye tracking research is a big challenge for academics interested in the computer vision field since the technique of eye detection requires a high-quality camera installed in specific equipment, which might be expensive. Unmodified cameras are more difficult to operate than infrared cameras since the pupil cannot be readily located all of the time. The first issue is that unaltered webcams only function in visible light. The second issue is that they often feature a wide-angle lens with few zoom options. As a result, the picture of the eyes is of poor quality and extremely reliant on the quality of lighting, therefore, varied lighting circumstances and head posture might make estimating the person's gaze difficult (*7*).

Remote tracking employs a computer display or screen to estimate gaze position, whereas eye gaze tracking uses cameras and infrared sensors to capture eye movements. Tobii EyeX is a commercially available EGT device; however, it is expensive and needs specialist hardware. These gadgets can cost up to 25,000 dollars, putting them out of reach for most people (*8*). Existing gaze-tracking systems include

***Corresponding author: Amal Hameed Khaleel**, Department of Computer Science, College of Computer Science and Information Technology, Basrah University, Barsh, 61004, Iraq, E-mail: amal_albahrany@yahoo.com. https://orcid.org/0000-0003-2759-2897
Thekra H. Abbas, Department of Computer Science, College of Science, Mustansiriyah University, Baghdad, Iraq. https://orcid.org/0000-0002-1202-2026
Abdul-Wahab Sami Ibrahim, Department of Computer Science, College of Education, Mustansiriya University, Baghdad, Iraq. https://orcid.org/0000-0001-5112-7640

significant setup complexity, costly components, rigid configuration, and time-consuming data-gathering processes (*9*). As a result, current research has concentrated on building appearance-based gaze estimate methods that anticipate eye gaze from pictures taken with standard cameras. This method is more accessible as well as cost-effective for recording eye movements (*10*).

In appearance-based eye tracking, characteristics such as head posture, camera settings, and user distance to the screen or camera are required for gaze estimation. Most research, however, gathers data with a fixed head attitude and in a controlled setting, neglecting distance information between the user and the camera. As a result, the gaze estimate findings are imprecise, limiting its usability for precision location applications (*11–15*).

Scleral contact lenses, electro-oculography, and image/video-oculography are some of the techniques used for eye detection. The visible spectrum approach non-invasively localizes the iris location employing infrared illumination. These techniques, however, are less accurate in uncontrolled lighting situations and call for additional gear (*16*).

This paper presents an eye detection technique employing low-resolution face images taken with an inexpensive camera.

This study's primary contributions are as follows:

(a) We build a new dataset of eye images.
(b) We suggest two methods for determining the eye region: facial landmarks and the Haar cascade technique. In real-time, webcams can use direct methods (convolution neural networks [CNN]) and engineering methods based on distances to identify the eye's iris region. The proposed deep learning-based methodology is effective in determining the eye's gaze direction within a limited mobility range, while engineering methods improve their effectiveness in a wide mobility range.

The remaining sections of the paper's content are structured as follows. The second section highlights current studies for eye detection and eye-tracking methods, whereas the third section gives a full explanation of our novel dataset of eye regions and recommended methodology for eye detection and gaze tracking. The fourth section is devoted to the findings and discussion, whereas the fifth section is focused on concluding remarks.

2 Related work

This section describes the most current approaches, methodologies, and results of previous works in eye detection and gaze tracking. Manuscripts published from 2018 to 2023 were considered for reviewing related studies.

2.1 Eye landmark detection

The quality of the basic face area and facial landmark detection influences how accurate facial recognition algorithms are. The face recognition method was used by Cheng et al. (*17*) to extract face areas, extract single-eye regions, and carry out very accurate position estimates. Retina-Face which predicts face bounding boxes using extra-supervised as well as self-supervised multitask learning was suggested by Dent et al. (*18*). Occlusion is a problem in face detection, the Occlusion-aware Face Detector (AOFD) was developed by Chen et al. (*19*), which uses an adversarial training technique to identify faces with few visible facial landmarks. In other approaches. Putro et al. (*20*) employed a face area scaled to 128×128 pixels preceding a bounding-box estimate of eyesight, while some, such as Ahmed (*21*) and Leo et al. (*22*) utilized statistically facial-landmark data to crop off individual eyes preceding a real-time eye segmentation technique.

Regarding face-landmark-detection algorithms, Wang et al. (*23*) extracted essential points from face-bounding boxes that reflect facial landmarks. Early landmark detection algorithms relied heavily on statistical approaches to fit a deformable face mesh.

Eye appearance-based models rely on the photometric appearance of the eyes. Using a Semi-Circular Edge shape and Semi-Ellipse Edge Shape characteristics, a completely automated eye localization approach based on the geometric form of the iris and eyelid was suggested in (*24*). Manir and Rabul used also these characteristics to combine the (histogram of oriented gradients, Hit-or-miss transform, and Local binary pattern) features allowing for the detection of eye candidates in face pictures, which improves the process overall and improves performance during the verification step by extracting slope edges (*25*). Xia et al. (*26*) presented hybrid regression and isophote curvature methods for eye center localization, but their model struggles with specular reflection or closed eyes. Abbasi and Khosravi (*27*) proposed a particle filter-based pupil detection technique, but their model performs worse in real-world settings. Choi et al. (*28*) introduced a convolutional neural network-based eye pupil localization technique, while Liu et al. (*29*) suggested a weight binarization cascade convolution neural network for eye localization.

Naseem et al. (*30*) suggested Faster RCNN, and a rectangular-intensity-gradient technique for face, eye detection, openness identification, and eye center localization, with suggested AlexNet assisting in detecting eye state (opened or closed). On the other side, Ahmad and Laskar

(*2*) used support vector machines (SVMs), and convolutional neural networks (CNNs)for iris-shape feature extraction to propose an accurate eye center identification and localization model. It used iris shape characteristics from the upper face area to extract rough eye candidates by the CNN to extract high-level features, and SVM for classification.

In-the-nature facial-landmark identification has improved significantly with CNN-based landmark detectors. The approaches (*31–35*) are commonly assessed about 68 points employing datasets of labeled visible-light pictures.

Many iris-landmark identification techniques (*36–38*) leverage cropped single-eye areas from facial-landmark recognition findings. Choi et al. (*35*) presented a segmentation-based eye position estimate by cutting off a rectangle area employing landmarks of the eye socket and eye corner using 68 points while Ablabatski et al. (*38*) discovered five iris features in a 64×64 single-eye area. A quick eye-identification method that utilizes a Siamese network was created for NIR partial face images (*39*). Moreover, Bazarevsky et al. (*40*) recognized the left- and right-eye centers using the facial landmark recognition technique with reasonable accuracy.

2.2 Eye-tracking methods

Eye movements are critical in eye tracking and also influence human-computer interaction. The widespread technology to track eyes has broadened its use. Eye-tracking technologies, for example, are increasingly utilized to detect abnormalities linked to a variety of psychological conditions (*41*). Maurage et al. (*42*) described in detail how they used a gaze-tracking device to identify alcoholism. Data on gaze-tracking technologies in neurological disorders and their potential therapeutic effect on brain function were provided by Bueno et al. (*43*). Eye tracking has also been used by Robertson et al. (*44*) to suggest a method for mild speech comprehension difficulties in dyslexic children.

Eye tracking is frequently used to improve learning. In order to show how well a C programming course is taught, Sun et al. (*45*) used eye-tracking equipment to identify learners' difficulties, offer them recommendations, and help them improve their self-efficacy for learning the C language. While Molina et al. (*46*) presented an empirical analysis combining eye-tracking and student-subjective evaluation.

Using real-time eye tracking, Kerr et al. (*47*) presented a corrective method that uses closest neighbor calibration locations to calculate anticipated drift at the user's attention after analyzing calibration information that has been gathered from the user. Pavlas et al. (*48*) provided helpful guidance on how to establish low-cost tracking. They used ITU Gaze Tracker and Eye Writer software in their system. Lee et al. (*49*) developed a method for assessing 3D gaze position using light reflections (Purkinje images) on the surface of the cornea and lens while accounting for the 3D optical framework of a human eye system. Borsato et al. (*50*) created a stroboscopic catadioptric eye tracking (SCET) software using roller shutter cameras and stroboscopic structured infrared light. Krafka et al. (*51*) introduced the Gaze Capture software, which used a similar technique to track eyeballs in restricted spaces, using an infrared LED for labeling and the eye's inner space for training. In (*52*), the authors suggested a computationally efficient approach for extracting iris regions from pictures, with an emphasis on segmentation for iris localization. To determine the iris error-free location, the authors (*53*) employed rough entropy, a soft-computing technique, and Circle Sector Analysis (CSA). The approach fared well when contrasted with the most advanced technologies, but it is extremely costly in real-time applications. Park et al. (*54*) presented a network-based stacked-hourglass approach of learning for localizing eye area features localization, which they tested on the Columbia Gaze, EYEDIAP, MPIIGaze, and UT Multiview datasets. These models can help human-computer interactions in real-time gaze detection applications. Deep learning is a sophisticated technology that has rapidly evolved and is frequently employed in various gaze estimation applications (*55*).

Several methods for estimating gaze based on appearance have been presented. For example, the authors of (*56*) presented a multimodal convolutional neural network; however, they first utilized the SURF cascade technique (*57*) and the limited local mode architecture (*58*) to distinguish faces and landmarks of the face before feeding this network data. The researchers (*51*, *59*) utilized the Gaze Capture dataset to train an iTracker based on a CNN network; the network gathers eye pictures and face images with their positions associated in a face grid-like structure. In (*60*) face-based two-dimensional and three-dimensional gaze detection was proposed. The researchers used a weight mechanism that takes information from multiple parts of the face and trains the pattern utilizing a standard CNN network. The researchers (*61*) employed a recurrent convolution network with distant cameras to consider the person and head posture as a specific issue for 3D gaze prediction. They used a many-to-one recurring model to predict gaze coordinates and fed the eyes region, face, as well as face landmark points into the CNN network as distinct inputs. Manir and Rabul (*15*) developed the gaze estimation method, which makes use of the Iris Center and Eye Corner to estimate gaze in low-resolution face pictures. A circular gradient-intensity-based

operator is used for a rough estimate, a CNN model is used for the correct Iris Center, and an Explicit Shape Regression approach (ESR) is used for Eye Corner localization. The experimental findings indicate that the suggested technique may be utilized to estimate gaze with more accuracy in both still pictures and movies. In (*62*), the authors introduced a Gazemap for gazing calculation, which is a graphical illustration of the eyeball, iris, and pupil at its exact center. The study's authors generated an intermediary graphic representation rather than returning both eyeball angles, which improves the gaze estimate procedure. A capsule network for analyzing gaze information from an image's ocular region was provided by (*63*). The authors created a pose-aware Gazing-Net architecture for gaze estimation utilizing the concept of capsules, which transforms the intensity of pixels to feature initialization characteristics that aggregate into a greater number of features as the network grows in depth.

Some researchers have used intrusive devices for gaze tracking such as sophisticated electrodes, contact lenses, and head-mounted devices (*64–68*). The main downside of such methods is that they necessitate physical contact with the users and cannot be utilized without their permission. Furthermore, such gadgets are difficult and costly to use.

Although there is a wealth of research on gaze estimation, only a few have examined gaze prediction via unaltered cameras. The study by Xia et al. (*69*), for example, proposed a two-phase gaze prediction strategy based on neural network algorithms and logistic regression analysis. Their proposed technique may be employed on a range of mobile devices without the requirement for additional hardware or extensive prior expertise. On the other hand, the suggested gaze tracking system in (*70*) used a single on-camera allows for unrestricted head movement, and operates in real-time with high accuracy. It was used to recognize faces, extract the eye area, and calculate the user-screen distance.

3 Methodology

This section describes the suggested model in detail. First, the overview of the different approaches utilized in this study is given and describes the proposed datasets, followed by the suggested model.

3.1 Overview of the different techniques

In this paper, two different techniques are used as explained in the following:

3.1.1 Haar cascade classifier: Several researchers (*71–73*) suggest employing the Haar cascade to determine feature coordinates and detect faces. Viola and Jones (*74*) proposed the Haar cascade as a machine-learning method for detecting objects in pictures. Through a drawn rectangle, a trained Haar cascade checks a picture to determine if it contains the required item. The Haar method is speedier because it performs high-speed calculations that depend on the number of pixels within the rectangle feature instead of the value of each pixel in the picture. The item is detected in four stages: Haar-like feature, integral picture, AdaBoost learning, and Cascade Classifier (*75*, *76*).

The Haar Cascade Classifier for face detection is built around Haar-like characteristics. Haar features are employed to detect whether or not a feature exists in a given picture. Every characteristic produces just one value, which is computed by summing the pixels beneath the black rectangle.

The Haar-like component is a rectangular property that adds a distinctive hint to a picture to help it swiftly distinguish faces (*77*). Figure 1 depicts instances of common Haar-like characteristics (*75*).

The Haar Cascade Classifier has the disadvantage of discovering an additional feature in the picture that is falsely comparable to the eye characteristic (*6*).

3.1.2 Facial landmark extraction: There are various techniques for the extraction of facial landmarks; the essential techniques in this study are as follows:

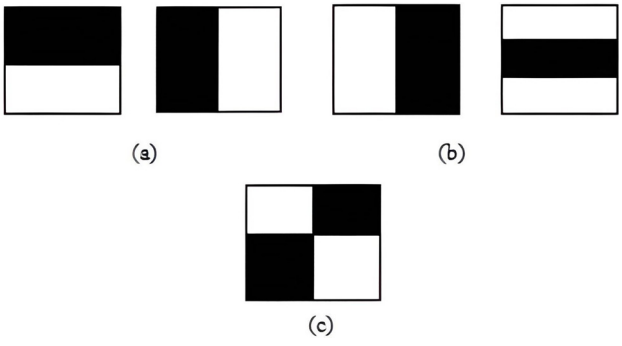


Figure 1: Type of feature (a) edge, (b) line, and (c) four-triangle.

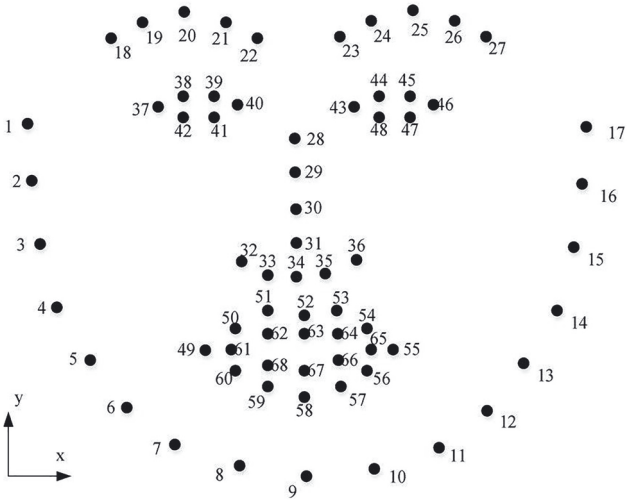


Figure 2: The map for dlib facial landmarks.

3.1.2.1 ***Using the dlib facial landmark detector:*** Dlib library indicated landmark detection technique is an implementation of Kazemi and Sullivan's Regression Tree Ensemble (ERT) (*78*). According to MultiPIE (*79*), this approach extracts 68 face landmarks using a simple and rapid algorithm as shown in Figure 2 (*77*).

These predicted locations are then revised through a continuous procedure including a cascade of regression coefficients. Regressors make another estimate based on the one before it at each iteration, seeking to decrease the variance of misalignment of the predicted points (*1*). To determine the shape of the eye, we chose to use the dlib package in our work. In the initial phase, the facial contour is calculated via the dlib.get frontal face detector () method. Next, we use the dlib program to input facial features for estimating eye form (shape predictor 68 face landmarks.dat) (*78*).

3.1.2.2 ***Using MediaPipe face mesh:*** The media pipe (*80*) is a powerful library for recognizing faces and facial landmarks. Pictures of the eyes are obtained from the library. To extract 478 face landmarks as shown in Figure 3 (*81*), MediaPipe Face Mesh employs a residual neural network architecture (*82*).

The landmarks used in this paper to detect the face and eyes are as follows in Table 1;

The MediaPipe face nets (*83*) are utilized to produce a three-dimensional representation of the face and retrieve three-dimensional nose values to serve as a reference for calculating head position. The X and Y output coordinates of the face mesh solution are normalized according to the frame length. The z vector represents the depth of the face wire mesh, which reflects the head's distance from the camera.

We use the media pipe command map_face_mesh = mp. solutions.face_mesh to specify face attributes for calculating eye shape (*78*).

The procedure begins by reading through each frame of the video and passing every frame to the library to earn facial landmark recognition. Images of both the left and right eyes are saved separately (*1*).

In terms of test reliability, we find that results extracted using MediaPipe metrics are better than using dlib metrics; therefore, this paper used MediaPipe metrics for eye detection.

3.1.2.3 ***EAR feature:*** The EAR function works by measuring the distance from the eyes' landmarks. In general, the EAR measure calculates a ratio based on the vertical as well as horizontal distances of six ocular landmark locations, as demonstrated in Figure 4 (*79*).

Beginning with p1 and concluding with p6, the coordinates in question are identified by numbers clockwise starting at the left-eye corner. Rosebrock (*84*) claims that each of the six coordinates from p1 to p6 is two-dimensional. When the eyes are open, the EAR value remains essentially constant, according to (*85*). When the eyes remain closed, however, the gap between coordinates p3 and p5 and p2 and p6 disappears, and the EAR ratio falls to 0 (*84*). In this paper, the EAR function is used to predict whether the eye is closed or not.

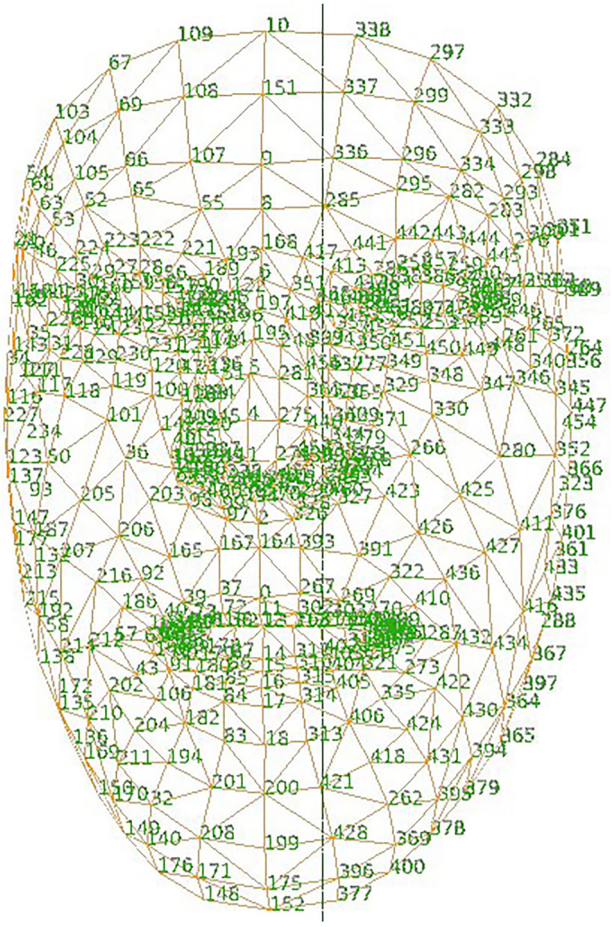


Figure 3: Media pipe face mesh solution map.

To determine the EAR value, the numerator computes the distance that exists across horizontal landmarks, while the denominator computes the distance that exists between both vertical landmarks and multiplies it by two to equalize, as shown in Equation (1) (*86*).

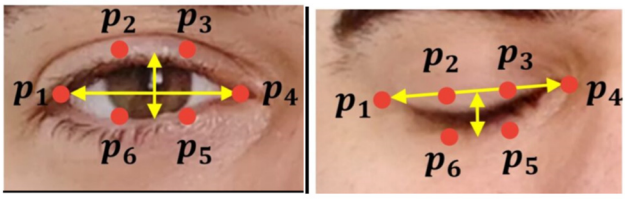
$$\mathrm{EAR} = \frac{||p_2 - p_6| + |p_3 - p_5||}{2|p_1 - p_4|} \tag{1}$$

3.2 Datasets

The dataset of the suggested system has eye images of people from different races and geographic areas. The global eye dataset included images of people's eyes from European countries, whereas the local eye dataset included images of people's eyes from Asian countries.

Table 1: Indices of the landmark.

Parts of facial	Indices of landmarks
Face border	[10, 338, 297, 332, 284, 251, 389, 356, 454, 323, 361, 288, 397, 365, 379, 378, 400, 377, 152, 148, 176, 149, 150, 136, 172, 58, 132, 93, 234, 127, 162, 21, 54, 103,67, 109]
Left eye	[362, 382, 381, 380, 374, 373, 390, 249, 263, 466, 388, 387, 386, 385,384, 398]
Right eye	[33, 7, 163, 144, 145, 153, 154, 155, 133, 173, 157, 158, 159, 160, 161, 246]



a) open-eye b) close-eye

Figure 4: EAR features.

Four different types of datasets were employed in this work, including two for training models and another two for results testing.

There are two different datasets used for training as follows:

3.2.1 First type: global eye dataset: The SBVPI (Sclera Blood Vessels, Periocular, and Iris) comprises 1800 images of 55 participants, categorized by gender, eye class, and view/gaze-direction labels. The dataset includes 450 images in each gaze direction, with a maximum resolution of 1700–3000 px. During a single recording session, a camera (DSLR @ Canon 60D) was used to capture RGB-colored images, as shown in the samples in Figure 5 (*87–90*).

3.2.2 Second type: local eye dataset: In datasets created by the proposed method, the sole equipment needed was a camera, ideally integrated into the top portion of the screen of the laptop, as demonstrated in Figure 6 (*6*).

The assumptions are that a person is seated directly facing the screen and the embedded camera at the highest point of the horizontal

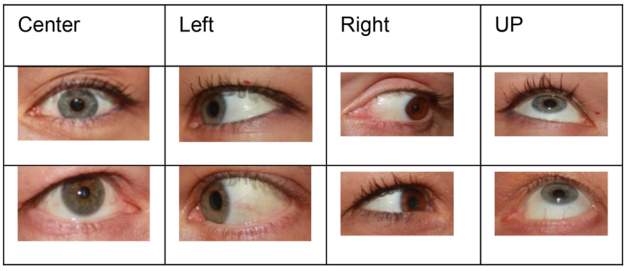


Figure 5: Global dataset.

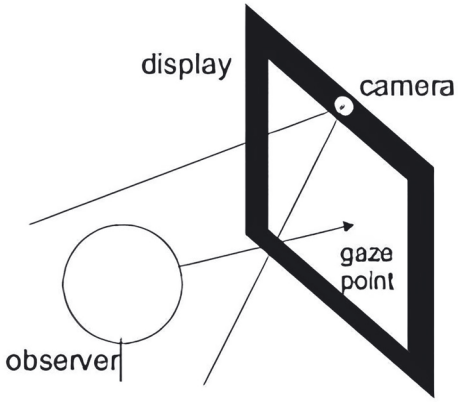


Figure 6: The experimental setup.

axis of their gaze with excellent lighting. This assumption allows for picture capture in real-world settings rather than traditional laboratory settings.

The proposed system can correctly predict the direction of eye gaze by steps of the algorithm following:

1. The computer screen should be placed at the level of the nose of the person sitting in front of it.
2. The head should be straight while sitting and facing toward the webcam and it not move or with slight head movements.
3. The computer should be close to the person at a distance of roughly 35 cm–50 cm away, the proxy distance between the user and the screen can be calculate by Equation (2)

$$D = \sqrt{(P_{Ux} - P_{Sx})^2 + (P_{Uy} - P_{Sy})^2} \quad (2)$$

where D is denoted by proxy distance, (P_{Ux}, P_{Uy}) is denoted by the position of the user, and (P_{Sx}, P_{Sy}) is denoted by the position of the screen computer.

There are used two different techniques to create a Local eye dataset as follows:

3.2.2.1 ***Haar cascade technique:*** The study utilized a local collection of human eye images to analyze eye movement in four directions. The images were collected from 15 individuals, with each having a different skin color. The Haar cascade technique was employed to determine the eye region in the images. The method focused on the upward movement of the eye in normal cases, as the iris does not rise significantly, and the downward movement is similar to the closing movement. The study involved 5000 images, captured under a different illumination (Figure 7).

3.2.2.2 ***Landmarks technique utilizing the MediaPipe library:*** The study utilized a local collection of human eye images captured from a webcam to analyze eye movement in four directions captured under indoor illumination and different skin colors. The MediaPipe Face Mesh technique was employed to determine the eye region in the images. The whole dataset is from 15 individuals: 1000 images for each of the five directions of eyes – 1000 images for the left, 1000 images for the right, 1000 images for the center, 1000 images for the top, and 1000 images for the down (Figure 8).

The global dataset has been used in the first experiments of the research. However, in the final research experiments, it was discarded and the focus was only on local datasets, as they are more diverse and realistic.

For the test in real-time, two approaches were used to get test data.

– The first approach uses the webcam to capture real-time images
– The second approach uses YouTube to download a video of eye movements. As it is necessary to extract the image of the eye, the videos used are face-focused, and Table 2 shows examples of the specifics of the videos.

3.3 Proposed method

The detection of the eye and gaze tracking are critical in the age of interactive technologies. The proposed system model consists of two parts: a built system and real-time run, the proposed system algorithm is shown in Figure 9.

The proposed model consists of two stages built as follows:

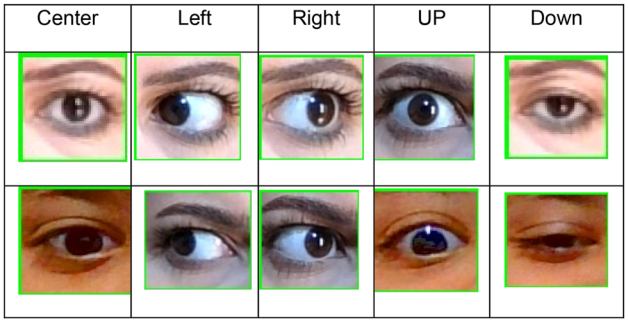


Figure 7: Haar – local dataset.

Table 2: Details of the video dataset.

Source	YouTube
Type	Mp4
Gender	Female
Size	8.87 MBs
Resolution	1920 × 1072
Frames per second	30
Duration (s)	20
Total images	500
Included images	453

3.3.1 Pre-processing: There are seven steps of the pre-processing as follows:

Step (1) uses a webcam to capture an image.

Step (2) improves image illumination and makes use of the HSV Color Space, which consists of a trio of channels color (H), saturation (S), and value (V). The first two are descriptions of colors, while the final one is a description of brightness (*91*).

HSV is preferred for geometric coordinate systems, which is usually more natural than HSL, which allows better color hue manipulation (*92*).

The images inputted by the webcam of the proposed system are in the format (RGB), but it converted to the format (BGR) because the programming language used is Python.

In order to improve the illumination of the input images to obtain high-accuracy prediction, the color format is converted to format (HSV) to control the saturation and color value degree, and then it returns the color formula to (BGR) color space.

To authentically recover the colors buried in the darkness and to simplify the brightness enhancement model, we developed a way to enhance the lighting while maintaining all the colors.

The suggested algorithm to enhance the lighting (Split-HSV) is as follows:

- Change from BGR to HSV color space.
- Isolate the s (saturation) and v (value) channels and then enhance their values such that the saturation equals $(1.5 \times S)$ and the value equals $(60 + V)$, as long as their values do not exceed 255.
- Re-merge the channels
- Return to BGR space

Step (3) image value normalization, which is the process of normalizing an image's pixel values. By dividing the data by 255, the process is normalized. It is strongly recommended that every input pixel in the data for the image contains an assigned value between zero and 255.

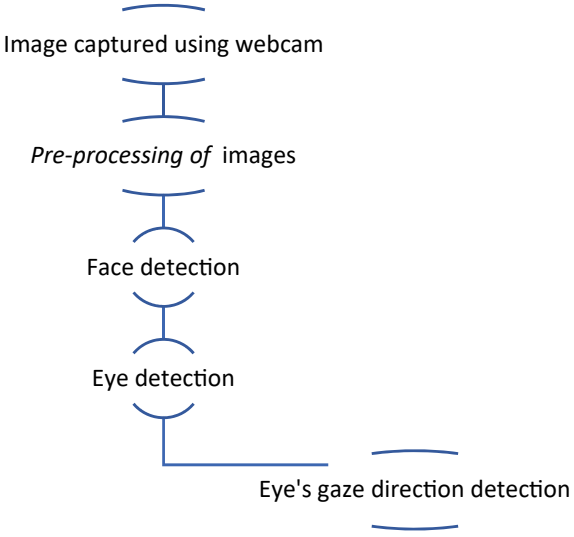


Figure 9: Proposed system algorithm.

Step (4) Change the image to gray to decrease complexity and noise.

Step (5) Clip off the eye region precisely using either the MediaPipe facial landmarks model to foresee 478 facial markers or the Haar cascade face bounding boxes.

Step (6) Rescale the images to 64×64 input shape; this is necessary to aid the models during training and requires that the images be scaled suitably to equal size.

Step (7) Divide the dataset into segments for training and testing. In this study, we split the data into 90/10 training and testing sets.

3.3.2 Processing: The most frequent method for tracking the direction of the eye's gaze is to track the eye's iris. To find the position of the eye's iris, first, determine the region of the eye.

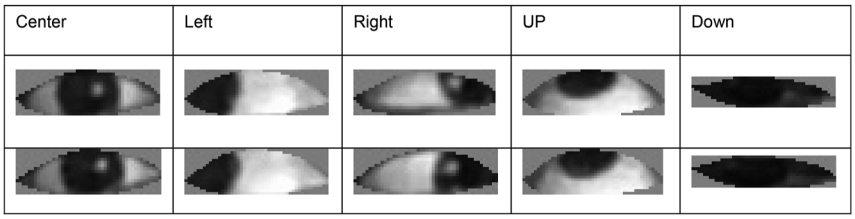


Figure 8: MediaPipe – local dataset.

There are two methods for detecting the eye region:

(1) Utilizing landmarks, either by using a library (dlib) or a more developed library (MediaPipe) to more correctly describe the landmarks of the face, this paper used the MediaPipe library.
(2) Determining the eye region using the Haar Cascade Classifier technique.

There are two basic approaches to identifying the direction of eye gaze using simply the webcam, as follows:

3.3.2.1 ***The direct method is the convolution neural networks (CNN) method:*** CNN is a deep neural network that uses a grid-like arrangement to analyze input. It is made up of three layers: convolutional, pooling, and fully connected (FC). The convolutional layer filters data, whereas the pooling layer samples feature maps. The final layer, the FC, generates the final output using an activation function such as sigmoid or SoftMax (*93*).

The architecture of the proposed model (Di-eyeNET) is separated into two blocks, as illustrated in Figure 10, Table 3:

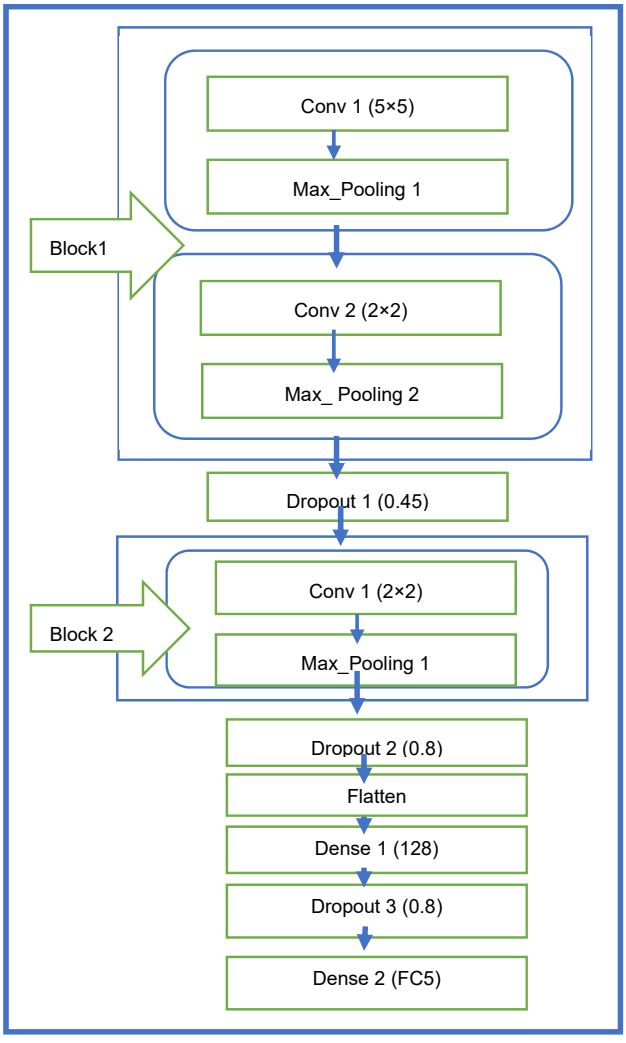


Figure 10: Architecture of Di-eyeNET.

Table 3: Summary of the parameters for training of Di-eyeNET.

Parameters	Local
Total images	5000
No. images-train	4000
No. images-test	1000
No. class	5
Total parameters	46,822
Trainable parameters	46,822
Non-trainable parameters	0
No. epoch	100
Batch-size	32
val_loss	0.0183
val_accuracy	0.9959
Early stopping	After 98 epochs val-loss not improved
Optimizer	Adam
Loss function	Categorical cross-entropy

The initial block is made up of two convolutional layers that use 128 large 5 × 5 filters that have a stride of 2 to collect adequate spatial information while reducing the dimension of the output feature maps. The layer is followed by a max-pooling layer that reduces the feature map size by 2 × 2 along each dimension using ReLU activations.

The second block is then used, which consists of one convolutional layer plus a max-pooling layer with two filters. The spatial dimensions associated with the feature maps reduce by half after block 1 and after each max-pooling function, we increase the total number of filters in the convolutional layers.

To ensure that the feature maps have enough representational capacity, the last two blocks' output is routed to a flattened layer, which is subsequently routed to a 128-D Fully Connected (FC) layer. To limit the number of trainable parameters while retaining great performance, the proposed design avoids a significant number of FC layers. Finally, the FC layer is connected to a single SoftMax layer that defines the identification of four eye directions.

To extract just the necessary characteristics while discarding the others, a two-step technique was used: Block 1 of the first stage comprises two convolutional layers. Since, after the first stage, we made the dropout of nearly half of the variables (characteristics), dropout was used after block 2 to reduce by about a quarter.

The model was run for more than 100 epochs with an Adam optimization strategy, a learning rate of 0.001, a batch size of 16, and an MAE loss function, with an input shape of 64 × 64. Each model's input image comprised three channels.

The models of CNN employed in the present research were built with Keras, an open-source neural network library written in Python. Additionally, we applied the most effective ResNet50 and VGG16 pre-trained model architectures.

VGG16 is a convolutional neural network model for image recognition. It is unique because it uses only 16 layers with weights rather than a large number of hyper-parameters (*94*, *95*).

We used this pseudo code to import and load the VGG16 pre-trained model:

```
from keras. applications. vgg16 import VGG16
model= VGG16 ( )
```

where the size image must be (224 × 224).

As part of the skip connection concept, the Resnet50 classic neural network served as the foundation for several computer activities. Using 150 layers, this model enables us to train CNN (*94*).

We loaded the ResNet50 pre-trained model with this pseudo code:

keras.models.load_model('resnet_50.h5')

where the size image must be (64 × 64).

Several training experiences for Di-eyeNET, ResNet50, and the VGG16 were undertaken. Analyses of the results demonstrate that the suggested model (Di-eyeNET) beats other techniques on certain parameters. The accuracy of Di-eyeNET, ResNet50, and VGG16 were 0.9959, 0.7933, and 1.7965, respectively; the MAEs of Di-eyeNET, ResNet50, and VGG16 were 0.0183, 0.5532, and 0.1416, respectively; during the testing phase, all models have almost comparable loss values; however, VGG16 has greater loss values during the training phase.

The size of the suggested model is decreased to 75 %, and the response time to 80 % when contrasted with the other models.

3.3.2.2 ***Engineering method:*** The engineering method is based on calculating distances, where after finding the eye region using Python libraries to determine the landmarks of the face and the eye.

There are many ways to know the position of the iris of the eye, which reflects the direction of the gaze, as the following:

Draw two perpendicular lines (horizontal and vertical) using specific equations (horizontal line, vertical line) on the iris region (blackest region), which is detected using the Hough transform and contour detection. After that, we identify the point of junction of the two straight lines, which represents the location of the iris and the direction of the gaze. The pseudo-code Python to draw two perpendicular lines are shown in Figure 11:

Used the Euclidean Distance function to find the point of intersection (ratio) of two perpendicular lines in Python as shown in Figure 12:

Divide the eye region into five regions: three horizontally, in which the blackest region represents the direction of gaze (left, center, right), and two vertically, in which the blackest region represents the direction of gaze (up, down), as shown in Figure 13:

Used the EAR feature to calculate the proportion of the distance between both landmarks of the eyes to know the closed-eye state, which is shown in Figure 14.

```
# horizontal (hori), vertical (ver)
   hori_right = landmarks[dir_indices[0]]
   hori_left = landmarks[dir _indices [8]]
     ver_top = landmarks[dir _indices [12]]
   ver_bottom = landmarks[dir_indices [4]]
     cv.line(img, hori_right, hori_left, utils.RED, 3)
     cv.line(img, ver_top, ver_bottom, utils. RED, 3)
```

Figure 11: Draw two perpendicular lines procedure.

```
   hori_Distance = euclaidean_Distance (hori_right, hori_left)
   ver_Distance = euclaidean_Distance (ver_top, ver_bottom)
   left_Ratio = left_hori_Distance/left_ver_Distance
   right_Ratio = right_hori_Distance/ right_ver_Distance
    ratio = (right _Ratio+left_Ratio)/2
```

Figure 12: Point of intersection of two perpendicular lines procedure.

```
piece_w = int(w/3)
piece_h = int(h/2)
  right_piece = threshed_eye[0:h, 0:piece_h]
  center_piece = threshed_eye[0:h, piece_h: piece_h+piece_h]
  left_piece = threshed_eye[0:h, piece_h +piece_h:w]
  top_piece = threshed_eye[0:piece_w,0:w]
  down_piece = threshed_eye[piece_w+piece_w:h,0:w]
```

Figure 13: Divide the eye region procedure.

```
lefte = calc_EAR(left_Eye)
righte = calc_EAR(right_Eye)
EAR_F = (lefte+righte)/2
EAR_F = round(EAR_F,2)
if EAR_F <0.16:
      closed-eye
else:
   opened-eye
```

Figure 14: EAR procedure.

4 Results and discussion

This section contains results. The experimental strategies are described, followed by a discussion of the particular results.

4.1 Experimental setup

The suggested system was created using the Num Py, Open CV, Tensor Flow, Keras, dlib, and MediaPipe libraries in Python 3.9 for training the models.

To create this system, we utilized a laptop with an i7 processor, 16 GB of RAM, and a GeForce GTX graphics card. The development environment utilized was Anaconda. Training and testing were conducted as two separate steps in the system's adoption.

4.2 System evaluation

Evaluation is an important phase in which the experimental data are utilized to forecast the results and assess the algorithms. Accuracy, loss (MAE), response time, and model size

are some of the metrics used to assess the performance of the various models (*1*).

1. Accuracy: is the key statistical effectiveness parameter used to evaluate a model. Accuracy is explained in Equation (3).

$$\mathrm{ACC} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{FP} + \mathrm{TN} + \mathrm{FN}} \times 100$$

   Here, true positives are denoted by TP, true negatives by TN, false positives by FP, and false negatives by FN.
2. Mean Absolute Error (MAE): is a statistical indicator of the differences (loss) between measured (y') and real (y). Equation (4) shows the formula for calculating MAE.

$$\mathrm{MSE} = \frac{1}{N}\sum_{i=1}^{N}\left|Y_i - Y_i'\right|$$

   Here, n is the total number of classes, y' denotes output measured, and y denotes output actually.
3. Response time: For real-time applications, response time is critical.
4. Models Size (Load Time): When implemented in real-time, size models must load rapidly and easily. It is difficult to load the model instantly when it is too huge. Except for the proposed model, the majority of other models are too huge to be employed in real-time.

4.3 Results and discussion

In this paper, four methods were proposed to track the direction of eye gaze, which are: CNN of Haar, CNN of Mesh, Mesh of Inter. Point, and Mesh of Split.

There are two methods of the first direct type: CNN of Haar and CNN of Mesh which depend on training convolutional neural networks, while Mesh of Inter. Point and Mesh of Split are two types of geometric methods, which depend on calculating the distances between facial landmarks using the Python library (MediaPipe).

Mesh of Inter. Point method uses three functions (a function to determine the iris of the eye, a function to draw the two perpendicular lines, and a function to know and examine the intersection point of the straight lines), whereas the Mesh of Split method uses two functions (a function to divide the eye into five sections and a function to determine the iris of the eye in any part after the division; the darker region represents the direction of gaze).

The results in the following tables were acquired utilizing the proposed approach to enhance the lighting since the results of the Haar cascade method without an enhancement approach were extremely few and did not exceed 50 % in the event of a poorly lit examination room.

The difficulties of doing studies in real-time appear apparent since the results obtained are high, but their accuracy may be reduced by half if standard criteria are not followed.

It is important to note that while the results of the CNN methods were nearly 99 % accurate in the training phase, they were significantly lower in the real-time examination phase due to the challenge of controlling the proper lighting and the subject's sitting position, as the angles of taking pictures affect the quality of the image entering the network and consequently affect the anticipated result. Therefore, all of these factors were taken into consideration when the results for the following tables were obtained, allowing for a fair comparison of the proposed techniques.

In the CNN of Haar method, the dataset images for training were constructed using the Haar Cascade Classifier approach by drawing a rectangle around the eye and truncating it, with the cut-out eye area defined as integrating the brow and the area around the eye with the eye. Thus, the large number of features trained by the Di-eyeNET method influenced the decision-making stage of the real-time tests, where only three classes were identified (it did not identify the top direction in real-time); although training accuracy was high (99 %) for all classes and test accuracy was 93 % (see Table 4).

The experiments were repeated by increasing the number of images and persons and modifying the training settings, but no satisfactory results were produced owing to the existence of like features impacting the resolution, such as the brow (which constitutes a major portion of the image).

Table 4: The results of the accuracy experiments of the proposed methods in a limited mobility range.

Proposed methods	No. of individual	Accuracy %					Total accuracy %
		Right	Center	Right	Center	Right	
CNN of Haar	15	95.12	98.594	95.257	86.891	89.44	93.0604
CNN of mesh	15	97.88	99.8	99.8	95.57	96	97.47
Mesh of inter. P.	15	93.85	95.97	92.087	90.54	91.99	92.8874
Mesh of split	15	85	96	86	88	79	89.2

As a result, we used the mesh function to create images for our dataset, which was distinguished by the creation of images that took only the eye because it draws a frame around the eye and then crops the eye only, and there are no additional features, as it only focuses on the features of the iris in the CNN method's training phase. As a result, the findings of the training and testing phases were highly accurate (99.7 %) for all classes, as shown in Table 5.

The results of the proposed methods vary according to the various assessment measures including accuracy, MAE, model size, and response time. According to the results of the experiments in Table 6, the CNN method outperforms the engineering method. The CNN method is the direct method because it uses a single function, whereas the engineering method uses multiple functions, as well as its accuracy is higher because neural networks play a significant role in determining whether the human eye looks up, forward, or down by measuring the parameters of the top eyelid and the amount of iris movement.

Moreover, the CNN of Mesh method is the most rapid and accurate. As for the time to load the software when running, which depends on the size of the software, note that the time to load optimal method (CNN of Mesh) is relatively longer, but the response time when working with it in real-time is the shortest, which is important because the faster the response, the more practical and acceptable the software at the user.

As a result, the techniques based on CNN produce superior results in the limited mobility range, whereas the geometry methods produce the best results in the wide mobility range.

An analysis of all experiments led to a conclusion, the software performed better when utilizing ready-made videos rather than trying the system in real-time since real-time results are affected by head movement and light, and the person's distance from the screen, which has a direct effect on the testing results. The individual must be no more than 50 cm from the screen and must not turn their head since the camera is not sufficiently accurate to capture images clearly if they are farther away. Furthermore, to improve the system's real-time performance, we proposed a strategy to improve the lighting of the webcam. The results in the table show that the accuracy was greatly improved while not affecting the response time, which is one of the most important measures of the system's success when implemented in real-time.

Despite the use of eye trackers, appearance-based approaches have low accuracy in gaze evaluation. Variable lighting, changes in ocular images, and head posture fluctuations all contribute to this variability, making high accuracy in camera-based gaze detection algorithms difficult (*70*).

Table 7 shows that all researchers employed strong constant light and maintained a consistent head position.

Table 5: The results of the accuracy experiments of the proposed methods in a wide mobility range.

Proposed methods	No. of individual	Accuracy %					Total accuracy %
		Right (0)	Center (1)	Left (2)	Top (3)	Down (4)	
CNN of Haar	15	95.3	98.9	95.4	87.09	89.9	93.318
CNN of mesh	15	98.58	99.887	99.809	97.597	98	98.7746
Mesh of inter. point	15	97.599	99.9432	95.099	97.11	97.544	97.459
Mesh of split	15	96.6	99.8	98.8	90.2	91.13	95.2

Table 6: The results of the experiments of the proposed methods using the measures of the evaluation.

Proposed methods	Mobility range	No. of individual	ACC	MAE	Response real-time	Load time
CNN of Haar	Wide	15	93.318	0.0763	1.5 s	4 s
	Limit	15	93.06	0.0799	1.5 s	4 s
CNN of mesh	Wide	15	98.774	0.0183	1 s	12 s
	Limit	15	97.47	0.0312	1 s	12 s
Mesh of inter. P	Wide	15	97.459	0.0362	3 s	5 s
	Limit	15	92.8874	0.0577	3 s	5 s
Mesh of split	Wide	15	95.2	0.0453	2 s	10 s
	Limit	15	89.2	0.1996	2 s	10 s

Table 7: Comparison of the recent works of researchers with the suggested study.

References	Equipment	Accuracy (˚ or %)	Method	Dataset	Function
(*10*) (2020)	Tobii eye tracker	80 %	Model and appearance-based	289,222 images	CNN
(*11*) (2020)	One camera	5.65°	Appearance-based	90 subjects	CNN
(*15*) 2021	Webcam	94.39 %	Appearance-based	NITSGoP database	CNN & ESR
(*68*) (2021)	Infrared sensors & wearable eye tracker	80 %	Appearance-based	7094 eye images	CNN
(*70*) (2022)	Webcam	84 %	Appearance-based	6800 images	CNN
This study (2023)	Webcam	98.77 %	Model and appearance-based	MediaPipe – local dataset	CNN of mesh

Table 7 compares the recent works of researchers who use CNN based on specific tunable factors (equipment, accuracy, method, dataset, and function).

It demonstrates that there is now a trend toward employing a standard camera and away from eye-tracking equipment since the emphasis has been on methods based on appearance. The accuracy results in Table 7 also indicate that the method presented for this study is the best.

5 Conclusions

This study provided affordable real-time gaze-tracking software that may be utilized with standard laptop or desktop computers regardless of the need for any extra devices. The cutting-edge CNN network (Di-eyeNET) was employed together with the MediaPipe library of Face Mesh, using unmodified webcams typically used with PCs.

Unaltered webcams provide images of poor quality that are especially susceptible to changes in illumination. As a result, getting excellent results was extremely hard in real-time. This research, on the other hand, shows that the use of well-organized parameters with a CNN network can yield valuable findings for real-run implementation. The proposed model (Di-eyeNET) was used to determine the direction of gaze in our research, and it has proven its effectiveness, as the implementation of the method of neural networks when the direction of gaze is required within a limited mobility range (i.e., a computer screen), while engineering methods increase their effectiveness when the direction of gaze is required within a wide mobility range.

However, the proposed method proved effective when compared to existing methods. In addition to its high accuracy, it is a comfortable method for the user because it is non-intrusive and relies on appearance.

In the future, we look forward to using the proposed method on mobile devices and working on improving it for use in mobile environments.

Research ethics: The local Institutional Review Board deemed the study exempt from review.

Author contributions: The authors have accepted responsibility for the entire content of this manuscript and approved its submission.

Competing interests: The authors state no conflict of interest.

Research funding: None declared.

Data availability: The raw data can be obtained on request from the corresponding author.

References

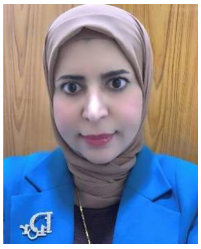
1. Adnan M., Sardaraz M., Tahir M., Dar M. N., Alduailij M., Alduailij M. A Robust Framework for Real-Time Iris Landmarks Detection Using Deep Learning. *Appl. Sci.* **2022**, *12*(11), 2022.
2. Ahmed M., Laskar R. H. Eye Center Localization Using Gradient and Intensity Information under Uncontrolled Environment. *Multimed. Tools Appl.* **2022**, *81*(5), 7145—7168.
3. Ahmad N., Laskar R. H., Hossain A., Ahmed M. Precise Eye Center Localization in a Practical Environment. In *IEEE Reg. 10 Annu. Int. Conf. Proceedings/TENCON* 2021; pp. 533—538.
4. Mou J., Shin D. Effects of Social Popularity and Time Scarcity on Online Consumer Behaviour Regarding Smart Healthcare Products: An Eye-Tracking Approach. *Comput Hum Behav* **2018**, *78*, 74—89.
5. Kumar D., Sharma A. Electrooculogram-Based Virtual Reality Game Control Using Blink Detection and Gaze Calibration. In *2016 International conference on advances in computing, communications and informatics, ICACCI 2016*, 2016; pp. 2358—2362.
6. Pastel S., Chen C. H., Martin L., Naujoks M., Petri K., Witte K. Comparison of Gaze Accuracy and Precision in Real-World and Virtual Reality. *Virtual Real.* **2021**, *25*, 175—189.
7. Ansari M. F., Kasprowski P., Obetkal M. Gaze Tracking Using an Unmodified Web Camera and Convolutional Neural Network. *Appl. Sci.* **2021**, *11*(19), 2021.
8. Farnsworth B. Eye Tracker Prices. Available at: https://imotions.com/blog/eyetracker-prices/.2019.
9. Yiu Y. H., Aboulatta M., Raiser T., Ophey L., Flanagin V. L., Eulenburg P., Ahmadi S. A. DeepVOG: Open-Source Pupil Segmentation and Gaze Estimation in Neuroscience Using Deep

Learning. *J. Neurosci. Methods* **2019**, *324*, 108307. https://doi.org/10.1016/j.jneumeth.
10. Meng C., Zhao X. Webcam-Based Eye Movement Analysis Using CNN. *IEEE Access* **2017**, *5*, 19581–19587.
11. Sattar H., Fritz M., Bulling A. Deep Gaze Pooling: Inferring and Visually Decoding Search Intents from Human Gaze Fixations. *Neurocomputing* **2020**, *387*, 369–382.
12. Cheng Y., Zhang X., Lu F., Sato Y. Gaze Estimation by Exploring Two-Eye Asymmetry. *IEEE Trans. Image Process.* **2020**, *29*, 5259–5272.
13. Ahmed M., Laskar R. H. Evaluation of Accurate Iris Center and Eye Corner Localization Method in a Facial Image for Gaze Estimation. *Multim. Syst.* **2021**, *27*, 1–20.
14. Valtakari N. V., Hooge I. T. C., Viktorsson C., Nyström P., Falck-Ytter T., Hessels R. S. Eye Tracking in Human Interaction: Possibilities and Limitations. *Behav. Res. Methods* **2021**, *53*, 1–17.
15. Zhuang Y., Zhang Y., Zhao H. Appearance-Based Gaze Estimation Using Separable Convolution Neural Networks. In *2021 IEEE 5th Advanced Information Technology, Electronic And Automation Control Conference (IAEAC)*, Vol. 5, 2021; pp. 609–612.
16. Ahmed M., Laskar R. H. Evaluation of Accurate Iris Center and Eye Corner Localization Method in a Facial Image for Gaze Estimation. *Multimed. Syst.* **2021**, *27*(3), 429–448.
17. Cheng Y., Wang H., Bao Y., Lu F. Appearance-Based Gaze Estimation with Deep Learning: A Review and Benchmark, 2021. Available at: http://arxiv.org/abs/2104.12668.
18. Deng J., Guo J., Zhou Y., Yu J., Kotsia I., Zafeiriou S. RetinaFace: Single-Stage Dense Face Localisation in the Wild, 2019. Available at: http://arxiv.org/abs/1905.00641.
19. Chen Y., Song L., Hu Y., He R. Adversarial Occlusion-Aware Face Detection. In *2018 IEEE 9th International Conference on Biometrics Theory, Applications and Systems, BTAS*, 2018.
20. Dwisnanto Putro M., Nguyen D. L., Jo K. H. Fast Eye Detector Using CPU Based Lightweight Convolutional Neural Network. In *International Conference on Control, Automation and Systems, 2020-Octob (October), 12–16*, 2020.
21. Ahmed N. Y. Real-time Accurate Eye Center Localization for Low-Resolution Grayscale Images. *J. Real-Time Image Process.* **2021**, *18*(1), 193–220.
22. Leo M., Cazzato D., De Marco T., Distante C. Unsupervised Approach for the Accurate Localization of the Pupils in Near-Frontal Facial Images. *J. Electron. Imag.* **2013**, *22*(3), 033033.
23. Wang N., Gao X., Tao D., Yang H., Li X. Facial Feature Point Detection: a Comprehensive Survey. *Neurocomputing* **2018**, *275*, 50–65.
24. Ahmed M., Laskar R. H. Eye Center Localization in a Facial Image Based on Geometric Shapes of Iris and Eyelid under Natural Variability. *Image Vis. Comput.* **2019**, *88*, 52–66.
25. Ahmed M., Laskar R. H. Eye Detection and Localization in a Facial Image Based on Partial Geometric Shape of Iris and Eyelid under Practical Scenarios. *J. Electron. Imaging* **2019**, *28*(03), 1.
26. Xia Y., Lou J., Dong J., Qi L., Li G., Yu H. Hybrid Regression and Isophote Curvature for Accurate Eye Centre Localization. *Multimed. Tools Appl.* **2020**, *79*(1), 805–824.
27. Abbasi M., Khosravi M. R. A Robust and Accurate Particle Filter-Based Pupil Detection Method for Big Datasets of Eye Video. *J. Grid Comput.* **2020**, *18*(2), 305–325.
28. Choi J. H., Lee K. I., Song B. C. Eye Pupil Localization Algorithm Using Convolutional Neural Networks. *Multimed. Tools Appl.* **2020**, *79*(43–44), 32563–32574.
29. Liu Z.-T., Jiang C.-S., Li S.-H., Wu M., Cao W.-H., Hao M. Eye State Detection Based on Weight Binarization Convolution Neural Network and Transfer Learning. *Appl. Soft Comput.* **2021**, *109*, 107565. https://doi.org/10.1016/j.asoc.2021.107565.
30. Ahmad N., Yadav K. S., Ahmed M., Hussain Laskar R., Hossain A. An Integrated Approach for Eye Centre Localization Using Deep Networks and Rectangular-Intensity-Gradient Technique. *J. King Saud Univ. – Comput. Inf. Sci.* **2022**, *34*(9), 7153–7167.
31. Sun Y., Wang X., Tang X. Deep Convolutional Network Cascade for Facial Point Detection. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2013; pp. 3476–3483.
32. Zhou E., Fan H., Cao Z., Jiang Y., Yin Q. Extensive Facial Landmark Localization with Coarse-To-Fine Convolutional Network Cascade. In *Proceedings of the IEEE International Conference on Computer Vision*, 2013; pp. 386–391.
33. Chandran P., Bradley D., Gross M., Beeler T. Attention-Driven Cropping for Very High-Resolution Facial Landmark Detection. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2020; pp. 5860–5869.
34. Zhang K., Zhang Z., Li Z., Qiao Y. Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks. *IEEE Signal Process. Lett.* **2016**, *23*(10), 1499–1503.
35. Feng Z. H., Kittler J., Awais M., Huber P., Wu X. J. Wing Loss for Robust Facial Landmark Localisation with Convolutional Neural Networks. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2018; pp. 2235–2245.
36. Choi J. H., Lee K.Il, Kim Y. C., Song B. C. Accurate Eye Pupil Localization Using Heterogeneous CNN Models Department of Electronic Engineering, Inha University, Republic of Korea Display Research Center, Samsung Display Co., LTD, Republic of Korea. In *2019 IEEE International Conference on Image Processing (ICIP)*, 2019; pp. 2179–2183.
37. Lee K. I., Jeon J. H., Song B. C. Deep Learning-Based Pupil Center Detection for Fast and Accurate Eye Tracking System. In *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 12364 LNCS*, 2020; pp. 36–52.
38. Ablavatski A., Vakunov A., Grishchenko I., Raveendran K., Zhdanovich M. Real-Time Pupil Tracking from Monocular Video for Digital Puppetry, 2020, 4–7. Available at: http://arxiv.org/abs/2006.11341.
39. Ogino Y., Toizumi T., Tsukada M. Fast Eye Detector Using Siamese Network for NIR Partial Face Images. ArXiv:2202.10671v2 [Cs.CV] 4 Jan 2023. Available at: http://arxiv.org/abs/2202.10671.
40. Bazarevsky V., Kartynnik Y., Vakunov A., Raveendran K., Grundmann M. BlazeFace: Sub-Millisecond Neural Face Detection on Mobile GPUs. *CVPR Workshop on Computer Vision for Augment. Virtual Real.* **2019**, 3–6.
41. Larrazabal A. J., García Cena C. E., Martínez C. E. Video-Oculography Eye Tracking Towards Clinical Applications: A Review. *Comput. Biol. Med.* **2019**, *108*, 57–66.
42. Maurage P., Masson N., Bollen Z., D'Hondt F. Eye Tracking Correlates of Acute Alcohol Consumption: A Systematic and Critical Review. *Neurosci. Biobehav. Rev.* **2020**, *108*, 400–422.

43. Bueno A. P. A., Sato J. R., Hornberger M. Eye Tracking — The Overlooked Method to Measure Cognition in Neurodegeneration? *Neuropsychologia* **2019**, *133*, 107191. https://doi.org/10.1016/j.neuropsychologia.2019.107191.
44. Robertson E. K., Gallant J. E. Eye Tracking Reveals Subtle Spoken Sentence Comprehension Problems in Children with Dyslexia. *Lingua* **2019**, *228*, 102708. https://doi.org/10.1016/j.lingua.2019.06.009.
45. Sun J. C. Y., Hsu K. Y. C. A Smart Eye-Tracking Feedback Scaffolding Approach to Improving Students' Learning Self-Efficacy and Performance in a C Programming Course. *Comput. Hum. Behav.* **2010**, *95*, 66—72.
46. Molina A. I., Redondo M. A., Lacave C., Ortega M. Assessing the Effectiveness of New Devices for Accessing Learning Materials: An Empirical Analysis Based on Eye Tracking and Learner Subjective Perception. *Comput. Hum. Behav.* **2014**, *31*(1), 475—490.
47. Kerr R., Fuad M. M. M. A Real-Time Lazy Eye Correction Method for Low-Cost Webcams. *Proc. Comput. Sci.* **2019**, *159*, 281—290.
48. Pavlas D., Lum H., Salas E. How to Build a Low-Cost Eye-Tracking System. *Ergonom. Des.* **2012**, *20*(1), 18—23.
49. Lee J. W., Cho C. W., Shin K. Y., Lee E. C., Park K. R. 3D Gaze Tracking Method Using Purkinje Images on Eye Optical Model and Pupil. *Opt. Lasers Eng.* **2012**, *50*(5), 736—751.
50. Borsato F. H., Morimoto C. H. Towards a Low Cost and High-Speed Mobile Eye Tracker. In *Eye Tracking Research and Applications Symposium (ETRA)*, 2019.
51. Krafka K., Khosla A., Kellnhofer P., Kannan H., Bhandarkar S., Matusik W., Torralba A. Eye tracking for Everyone. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2016-Decem*, 2016; pp. 2176—2184.
52. Soliman N. F., Mohamed E., Magdi F., El-Samie F. E. A., AbdElnaby M. Efficient Iris Localization and Recognition. *Optik* **2017**, *140*, 469—475.
53. Sardar M., Mitra S., Uma Shankar B. Iris Localization Using Rough Entropy and CSA: A Soft Computing Approach. *Applied Soft Computing Journal* **2018**, *67*, 61—69.
54. Park S., Zhang X., Bulling A., Hilliges O. Learning to Find Eye Region Landmarks for Remote Gaze Estimation in Unconstrained Settings. *ETRA '18. Eye Track. Res. Appl. Symp.* **2018**, https://doi.org/10.1145/3204493.3204545.
55. Sewell W., Komogortsev O. Real-time Eye Gaze Tracking with an Unmodified Commodity Webcam Employing a Neural Network. In *Conference on Human Factors in Computing Systems — Proceedings*, 2010; pp. 3739—3744.
56. Zhang X., Sugano Y., Fritz M., Bulling A. Appearance-based Gaze Estimation in the Wild. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 07—12-June*, 2015; pp. 4511—4520.
57. Li J., Zhang Y. Learning SURF Cascade for Fast and Accurate Object Detection. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2013; pp. 3468—3475.
58. Baltrušaitis T., Robinson P., Morency L. P. Continuous Conditional Neural Fields for Structured Regression. In *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 8692 LNCS (PART 4)*, 2014; pp. 593—608.
59. Zhang Y., Tian X., Jia N., Wang F., Jiao L. Deep Tracking Using Double-Correlation Filters by Membership Weighted Decision. *Pattern Recogn. Lett.* **2020**, *136*, 161—167.
60. Zhang X., Sugano Y., Fritz M., Bulling A. It's Written All Over Your Face: Full-Face appearance-Based Gaze Estimation. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, 2017-July*, 2017; pp. 2299—2308.
61. Palmero C., Selva J., Bagheri M. A., Escalera S. Recurrent CNN for 3D Gaze Estimation Using Appearance and Shape Cues. In *British Machine Vision Conference 2018, BMVC 2018*, 2019.
62. Park S., Spurr A., Hilliges O. Deep Pictorial Gaze Estimation. In *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 11217 LNCS*, 2018; pp. 741—757.
63. Mahanama B., Jayawardana Y., Gaze-Net S. J. Appearance-Based Gaze Estimation Using Capsule Networks. In *Proceedings of the 11th Augmented Human International Conference, Winnipeg, MB, Canada, 2020-March, 1—4*, 2020.
64. Kaur A. Wheelchair Control for Disabled Patients Using EMG/EOG Based Human Machine Interface: A Review. *J. Med. Eng. Technol.* **2021**, *45*(1), 61—74.
65. Drakopoulos P., Koulieris G., Mania K. Eye Tracking Interaction on Unmodified Mobile VR Headsets Using the Selfie Camera. *ACM Trans. Appl. Percep.* **2021**, *18*(3), 1—20.
66. Pai Y. S., Bait M. L., Lee J., Xu J., Peiris R. L., Woo W., Billinghurst M., Kunze K. NapWell: An EOG-Based Sleep Assistant Exploring the Effects of Virtual Reality on Sleep Onset. *Virtual Real.* **2021**, *26*, 1—15.
67. Teiwes W., Merfeld D. M., Young L. R., Clarke A. H. Comparison of the Scleral Search Coil and Video-Oculography Techniques for Threedimensional Eye Movement Measurement. In *Three-Dimensional Kinematics of Eye, Head and Limb Movements*; Routledge: London, 2020; pp. 429—443.
68. Ou W.-L., Kuo T.-L., Chang C.-C., Fan C.-P. Deep-Learning-Based Pupil Center Detection and Tracking Technology for Visible-Light Wearable Gaze Tracking Devices. *Appl. Sci.* **2021**, *11*(2), 851.
69. Xia Y., Liang B., Li Z., Gao S. Gaze Estimation Using Neural Network and Logistic Regression. *Comput. J.* **2021**, *65*(8), 2034—2043.
70. Modi N., Singh J. Real-Time Camera-Based Eye Gaze Tracking Using Convolutional Neural Network: A Case Study on Social Media Website. *Virtual Real.* **2022**, *26*(4), 1489—1506.
71. Zhu Y., Zabaras N. Bayesian Deep Convolutional Encoder—Decoder Networks for Surrogate Modeling and Uncertainty Quantification. *J. Comput. Phys.* **2018**, *366*, 415—447.
72. Zhao Z., Zheng P., Xu S., Wu X. Object Detection With Deep Learning: A Review. *IEEE Trans. Neural Netw. Learn. Syst.* **2019**, *PP*, 1—21.
73. Al-Sabban W. H. Real-Time Driver Drowsiness Detection System Using Dlib Based on Driver Eye/Mouth Monitoring Technology. *Commun. Math. Appl.* **2022**, *13*(2), 807—822.
74. Viola P., Jones M. Rapid Object Detection Using a Boosted Cascade of Simple Features. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2001; p. 1.
75. Kamarudin N., Jumadi N. A., Mun N. L., Keat N. C., Ching A. H. K., Mahmud W. M. H. W., Morsin M., Mahmud F. Implementation of Haar Cascade Classifier and Eye Aspect Ratio for Driver Drowsiness

Detection Using Raspberry Pi. *Univ. J. Electric. Electron. Eng.* **2019**, *6*(5), 67—75.
76. Abdullah R. M., Alazawi S. A. H., Ehkan P. SAS-HRM: Secure Authentication System for Human Resource Management Reem. *Al-Mustansiriyah J. Sci.* **2023**, *34*(3), 64—71.
77. Viola P., Jones M. Robust Real-Time Face Detection. *Int. J. Comput. Vis.* **2004**, *57*(2), 137—154.
78. Rakhmatulin I., Duchowski A. T. Deep Neural Networks for Low-Cost Eye Tracking. *Proc. Comput. Sci.* **2020**, *176*, 685—694.
79. Roesler O., Kothare H., Burke W., Neumann M., Liscombe J., Cornish A., Habberstad D., Pautler D., Suendermann-Oeft D., Ramanarayanan V. Exploring Facial Metric Normalization For Within- and Between-Subject Comparisons in a Multimodal Health Monitoring gent. In *ACM International Conference Proceeding Series*, 2022; pp. 160—165.
80. Aman, Sangal A. Drowsy Alarm System Based on Face Landmarks Detection Using MediaPipe FaceMesh. In *Proceedings of First International Conference on Computational Electronics for Wireless Communications., Haryana, India, 11—12 June 2021*; Springer: Berlin/Heidelberg, Germany, 2022; pp. 363—375.
81. Albadawi Y., AlRedhaei A., Takruri M. Real-Time Machine Learning-Based Driver Drowsiness Detection Using Visual Features. *J. Imaging* **2023**, *9*(5), 1—18.
82. Tonsen M., Zhang X., Sugano Y., Bulling A. Labelled Pupils in the Wild: a Dataset for Studying Pupil Detection in Unconstrained Environments. In *Eye Tracking Research and Applications Symposium (ETRA)*, Vol. 14, 2016; pp. 139—142.
83. Kartynnik Y., Ablavatski A., Grishchenko I., Grundmann M. Real-time facial surface geometry from Monocular video on mobile GPUs. 2—5, 2019. Available at: http://arxiv.org/abs/1907.06724.
84. Datahacker. How to Detect Eye Blinking in Videos Using dlib and opencv in python. https://datahacker.rs/011-how-to-detect-eye-blinking-in-videos-using-dlib-and-opencv-in-python/ (accessed May 20, 2022).
85. Rosebrock A. Eye Blink Detection with opencv, python, and dlib. https://pyimagesearch.com/2017/04/24/eyeblink-detection-opencv-python-dlib/ (accessed May 7, 2022).
86. Soukupova T., Cech J. *Real-Time Eye Blink Detection using Facial Landmarks*; Research Reports of CMP, Czech Technical University: Prague, *5*, 2016; pp. 1—8.
87. Vitek M., Rot P., Štruc V., Peer P. A Comprehensive Investigation into Sclera Biometrics: A Novel Dataset and Performance Study. *Neural Comput. Appl.* **2020**, *32*(24), 17941—17955.
88. Rot P., Vitek M., Grm K., Emeršič Ž., Peer P., Štruc V. Deep Sclera Segmentation and Recognition. In *Advances in Computer Vision and Pattern Recognition*, 2020.
89. Rot P., Emersic Z., Struc V., Peer P. Deep Multi-Class Eye Segmentation for Ocular Biometrics. In *2018 IEEE International Work Conference on Bioinspired Intelligence, IWOBI 2018 — Proceedings*, 2018; pp. 1—8.
90. Ali Z., Park U., Nang J., Park J. S., Hong T., Park S. Periocular Recognition Using uMLBP and Attribute Features. *KSII Trans. Internet Inf. Syst.* **2017**, *11*(12), 6133—6151.
91. Jiang Z., Li H., Liu L., Men A., Wang H. A Switched View of Retinex: Deep Self-Regularized Low-Light Image Enhancement. *Neurocomputing* **2021**, *454*, 361—372.
92. Khassaf N. M., Shaker S. H. Image Retrieval Based Convolutional Neural Network. *Al-Mustansiriyah J. Sci.* **2020**, *31*(4), 43—54.
93. Al-Tai M. H., Nema B. M., Al-Sherbaz A. Deep Learning for Fake News Detection: Literature Review Mohammed. *Al-Mustansiriyah J. Sci.* **2023**, *34*(2), 70—81.
94. Kanade P., David F., Kanade S. Convolutional Neural Networks (CNN) based Eye-Gaze Tracking System using Machine Learning Algorithm. *Eur. J. Electric. Eng. Comput. Sci.* **2021**, *5*(2), 36—40.
95. Akinyelu A. A., Blignaut P. Convolutional Neural Network-Based Technique for Gaze Estimation on Mobile Devices. *Front. Artif. Intell.* **2022**, *4*, 1—11.

Bionotes



Amal Hameed Khaleel
Department of Computer Science, College of Computer Science and Information Technology, Basrah University, Barsh, 61004, Iraq
amal_albahrany@yahoo.com
https://orcid.org/0000-0003-2759-2897

Amal Hameed received a bachelor's degree and MSC degree in computer science from the Department of Computer Sciences, University of Basrah, Iraq. She currently works as an Assistant Professor at the Computer Science & Information Technology College, University of Basrah, Iraq. She has more than 20 years of experience including extensive project management in the related area. Her research interests are focused on Human-Computer Interaction, Computer Vision, and Data Mining.

Thekra HayderAli Abbas
Department of Computer Science, College of Science, Mustansiriyah University, Baghdad, Iraq
https://orcid.org/0000-0002-1202-2026

Thekra Abbas received the bachelor's degree in computer science from the Department of Computer Sciences, University of Technology, Baghdad, Iraq, in 1987, the M.Sc. degree in computer science from Mustansiriyah University, Baghdad, and the Ph.D. degree in computer science from Central South University, Hunan, China. She is an Assistant Professor with the Department of Computer Sciences, Mustansiriyah University. She leads and teaches modules in computer science. She has more than 25 years of experience including extensive project management, supervised researches in the related area. Her research interests include information technology, big data, and multimedia.

Abdul-Wahab Sami Ibrahim
Department of Computer Science, College of Education, Mustansiriya University, Baghdad, Iraq
https://orcid.org/0000-0001-5112-7640

Abdul-Wahab Sami attained his bachelor's degree in computer science from the Department of Computer Sciences at the University of Mustansiriyah, Baghdad, Iraq, in 1992. Following this, he completed his M.Sc. degree in computer science at the University of Technology, Baghdad, in 1996, and later earned his Ph.D. in computer science from the same university in 2006. Currently serving as an Assistant Professor in the Department of Computer Sciences at Mustansiriyah University, Dr. Sami not only leads and instructs various computer science modules but also boasts 23 years of comprehensive experience, encompassing teaching, extensive project management, and supervising research in related areas. His research interests span image processing, pattern recognition systems, encryption systems, and multimedia systems.