Iris Recognition System

Gherasim Dragoș-George
Universitatea Tehnica Gheorghe Asachi
Facultatea de Automatica si Calculatoare
Iasi, Romania
dragos-george.gherasim@student.tuiasi.ro

Varvara Ioan
Universitatea Tehnica Gheorghe Asachi
Facultatea de Automatica si Calculatoare
Iasi, Romania
ioan.varvara@student.tuiasi.ro

I. Introduction

Iris recognition is the process of recognizing a person by analyzing the random pattern of the iris (Figure 1). The automated method of iris recognition is relatively young, existing in patent only since 1994.The iris is a muscle within the eye that regulates the size of the pupil, controlling the amount of light that enters the eye. It is the coloured portion of the eye with colouring based on the amount of melatonin pigment within the muscle (Figure 2).

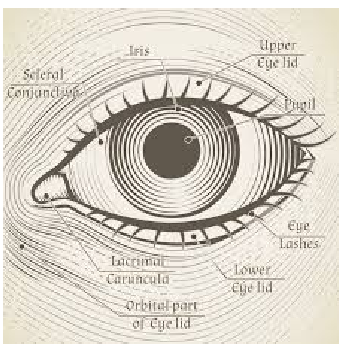


Fig. 1. Human Eye

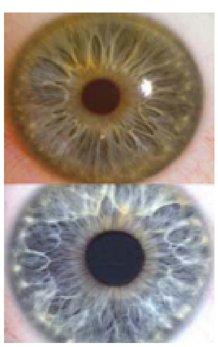


Fig. 2. Colored portion of eye

II. State of the Art and Related Work

There are different biometric authentication systems that are based on the characteristics of an individual. Recognition of face, fingers, voice and iris are among the most used characteristics for recognition. Currently, iris recognition is also used in different security systems, and recently, in clinical application systems.

In [3] it is mentioned that one of the main complications that have arisen to carry out the iris recognition has been the distance in which the image is acquired. When an image is obtained at a distance greater than 3 m, the iris image regularly becomes blurred, and, therefore, deficient in details such as to identify the texture of the iris due to the loss of information compared to images that are obtained at a smaller distance. In [4] other problems are identified, such as movement, lighting and noise, as well as the present refraction in the images. In addition to the obstruction of the eyelids, the use of glasses and hair prevent obtaining a complete image of the iris. It is considered that the fundamental objective of the segmentation process is to extract the iris texture from the structures that surround it, for example, the pupil, the eyelids, the sclera and to eliminate or reduce reflections of light in the iris. In recent years, segmentation methods have been presented with the aim of increasing the percentages of success in the identification. To facilitate segmentation processes, different iris databases have been used, in different sizes, distances and positions.

Study [8] has proposed an automatic segmentation algorithm using the circular Hough transform to identify the Iris pupil boundary and linear Hough transform for detecting the occluding eyelids. To remove the eyelashes and reflections thresholding is employed. The segmented region is normalized using the Daughman's rubber sheet model. The normalized Iris image is convolved with the 1D Log-Gabor wavelets and the resulting phase data is used to extract the feature from the Iris image. And finally for the template matching Hamming distance is employed. This recognition rate achieved a FAR of 0.005 percent and FRR of 0.238 percent for CSISA images.

A. Canny's Edge Detector Method

The identification of shapes by means of their edges in the images facilitates the classification of objects. In order to carry out the identification, some figures can be formed by the edges that compose them. [5] use the Canny's method with the first derivative for edge detection, based on the variation of intensity between pixels.

B. Hough Transform

The Hough transform consists of constructing a parametric space of regular geometric structures. The maximum zones of this space denote the regions with a high probability of finding these structures. Various investigations have shown that it is possible to detect different figures. Investigations have been conducted in which the Hough transform has been used to locate and segment the iris using different methodological approaches that mostly aimed at the elimination of noise, the location of the eye, and location of the center of the pupil, using different techniques to achieve iris segmentation. the Hough transform was used to detect the center of the pupil and from it to project the iris. Using different techniques for the elimination of noise on the images, in [6] the elimination of noise was made by applying a Gaussian filter.

C. Iris Segmentation

In the realm of iris segmentation research, various techniques and algorithms have been developed to enhance the accuracy and efficiency of this crucial process in the field of biometric recognition. A common approach involves using the Hough transform for detecting circles associated with the iris in an image. Algorithms such as Daugman's algorithm have made significant contributions, providing a robust method for iris segmentation based on the analysis of the Hough transform.

Furthermore, preprocessing techniques, such as applying the GaussianBlur filter to enhance image quality and reduce noise, have often been integrated into algorithms to ensure reliable results. Advanced methods have also explored machine learning techniques, such as convolutional neural networks, to achieve more precise iris segmentation.

Studies in iris segmentation have taken into consideration critical variables, such as adaptability to varying lighting conditions and robustness against interference factors. This research has emphasized the development of techniques that enable precise and rapid segmentation, contributing to continuous progress in the field of iris-based biometric recognition. By evaluating and comparing these methods, researchers have sought to identify optimal solutions for efficient iris segmentation in different contexts and usage scenarios[9].

D. Scale-invariant feature transform(SIFT)

The scale-invariant feature transform (SIFT) is a computer vision algorithm to detect, describe, and match local features in images, invented by David Lowe in 1999. Applications include object recognition, robotic mapping and navigation, image stitching, 3D modeling, gesture recognition, video tracking, individual identification of wildlife and match moving.[10]

SIFT keypoints of objects are first extracted from a set of reference images[1] and stored in a database. An object is recognized in a new image by individually comparing each feature from the new image to this database and finding candidate matching features based on Euclidean distance of their feature vectors. From the full set of matches, subsets of keypoints that agree on the object and its location, scale, and orientation in the new image are identified to filter out good matches. The determination of consistent clusters is performed rapidly by using an efficient hash table implementation of the generalised Hough transform. Each cluster of 3 or more features that agree on an object and its pose is then subject to further detailed model verification and subsequently outliers are discarded. Finally the probability that a particular set of features indicates the presence of an object is computed, given the accuracy of fit and number of probable false matches. Object matches that pass all these tests can be identified as correct with high confidence.

III. Method Description

For this project we decided to develop a functional code for detecting the iris and the pupil. This involves applying image pre-processing operations using the Python programming language, along with some libraries (OpenCV, NumPy, Pillow, Tkinter), within the PyCharm working environment.

A. Acquisition Phase

The image acquisition was performed from the CASIA-IrisV4 dataset (Chinese Academy of Sciences Institute of Automation - Iris Version 4)[7]. This dataset is renowned for its extensive collection of iris images, widely used for research and the development of recognition algorithms. CASIA-IrisV4 contains a total of 54,607 iris images from more than 1,800 genuine subjects and 1,000 virtual subjects. All iris images are 8 bit gray-level JPEG files, collected under near infrared illumination or synthesized. We also succeded to capture some personal iris images using phone's camera.

B. Pre-Processing Phase

Acquired images went through a previous process also called pre-processing, mainly due to the fact that photographs were obtained by different devices and conditions. Images that were obtained in color were transformed to gray scale in order to work better with them. It is important to note that the original images in the dataset are by default in grayscale. This conversion facilitates subsequent processing, considering that most edge detection and contour analysis algorithms operate efficiently on grayscale images. This image preprocessing step prepares the data for the subsequent stages of our analysis and feature extraction process.

C. Applying Gaussian Filter

The Gaussian filter blurs the desired area and cuts the noise with higher frequencies. It works the same as mean filters while representing average weight uniformly. These are linear filters that reduce the noise and blur the edges effectively. They are created as matrices in digital image processing, passing through each pixel of the selected portion.

Fig. 3. Images before applying Gaussian Filter

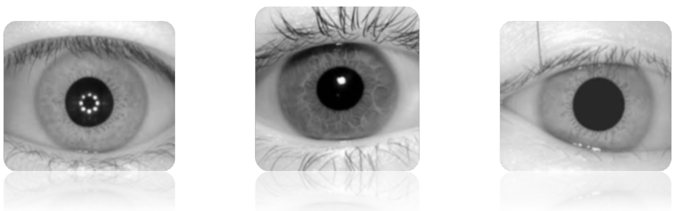


Fig. 4. Images after applying Gaussian Filter

D. Using Canny's Edge Detector algorithm

Canny's algorithm was used to detect the edges present in the image and to facilitate the object identification, mainly circles, by means of the Hough transform. The edge detection was made taking into account the intensity variation existing between one or more regions present in an image.

Points out that Canny's method uses the first derivate for the edge detection, taking into account the intensity: in those regions where the intensity does not change, it is established a value of 0, while in the case of a sudden intensity change, a value of 1 is established. These characteristics are used for edge detection.

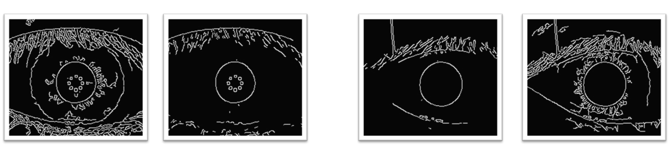


Fig. 5. Images after applying Canny's Edge Detection

E. Applying Hough transform

The Hough transform is a standard computer vision algorithm that can be used to determine the parameters of simple geometric objects, such as lines and circles, present in an image.

The circular Hough transform can be employed to deduce the radius and centre coordinates of the pupil and iris regions. Firstly, an edge map is generated by calculating the first derivatives of intensity values in an eye image and then threshold the result. From the edge map, votes are cast in Hough space for the parameters of circles passing through each edge point.

Here's a simplified explanation of the Hough Transform in the context of iris recognition:

1) Edge Detection: Before applying the Hough Transform to detect the iris, it's necessary to detect the edges of the iris in the image. This can be done using edge detection algorithms such as Canny Edge Detection.

2) Parametric Representation of Circles/Ellipses: The iris can be approximately described by a parametric equation of a circle or ellipse. A parametric equation for a circle, for example, can be written in general form as:

$$(x-a)^2 + (y-b)^2 = r^2$$

where (a,b) is the center of the circle, and r is the radius.

3) Hough Transformation: The goal of the Hough Transform is to identify lines or curves in an image. To detect circles or ellipses, a variant of the Hough Transform known as the Hough Circle Transform is used.

4) Accumulation in Hough Space: Each point on the iris edge is transformed into a curve in Hough space. If multiple curves intersect at the same point in Hough space, it indicates that these points are part of a circle or ellipse.

5) Choosing the Relevant Circles or Ellipses: After accumulation, points of maximum in Hough space indicate suitable circles or ellipses to represent the iris. These points can be identified and used to extract the necessary information for iris recognition.

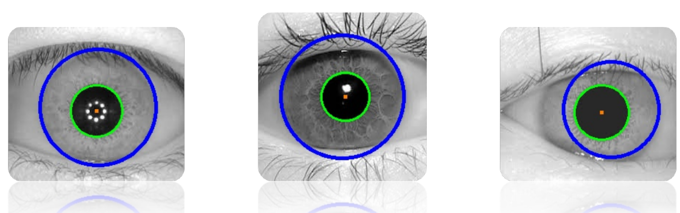


Fig. 6. Images after applying Hough Transform

F. Iris Segmentation

The segmentation of the iris represents a crucial step in iris recognition technologies, aiming to isolate and precisely highlight this distinctive region of the human eye. This technology is based on the idea that the iris, with its unique and complex patterns, can serve as a unique biometric feature for individual identification and authentication. The process of iris segmentation involves the use of sophisticated algorithms capable of detecting and extracting the iris from an eye image.

Within the iris recognition application, iris segmentation is a crucial process for the identification and precise isolation of this specific region of the eye. The segmentation method implemented in the "segmentirisalgorithm" goes through several critical steps. Firstly, the image undergoes a GaussianBlur filter to enhance clarity and reduce potential noise artifacts. Subsequently, the Canny method is applied to highlight the edges of the iris, and the Hough transform identifies the circles present in the image. We select the circle with the maximum radius, assuming it represents the iris, and reduce the radius to avoid the image edges. By drawing this circle on an empty image, a precise representation of the isolated iris is obtained.

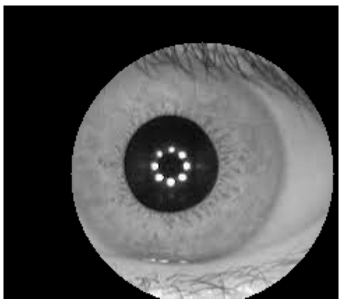


Fig. 7. Segmented Iris

G. Iris Recognition

This process relies on detecting and comparing the distinctive features of the iris using the Scale-Invariant Feature Transform (SIFT) algorithm and evaluating matches with the Fast Library for Approximate Nearest Neighbors (FLANN) algorithm. The final result is compared to a threshold, representing the minimum percentage of matches required to declare that the images are sufficiently similar.

The SIFT algorithm is a popular method for detecting and describing key points (distinctive features) in an image. Proposed by David G. Lowe in 1999, it has become one of the most widely used methods in the fields of visualization and object recognition. In the context of this algorithm, key points and descriptors are essential concepts for detecting and describing the distinctive features of an image.

Key points are distinctive locations in an image selected based on local properties (such as contrast variations, corners, edges, etc.). These key points are used to locate significant regions of the image that can be easily recognized and tracked. Algorithms like SIFT identify these key points by analyzing intensity variations in the vicinity of each pixel.

Descriptors are mathematical representations of the regions around the key points. They store information about texture, shape, and intensity distribution around the key point and are used to determine the degree of similarity between two regions of images. In the case of SIFT, descriptors are calculated by analyzing intensity gradients in the surrounding region of the key point.

The FLANN algorithm (Fast Library for Approximate Nearest Neighbors) is an efficient and flexible library for finding nearest neighbors in large datasets. Initially developed to provide a fast solution for finding nearest neighbors in high-dimensional spaces, it can be used to speed up searches in multidimensional datasets. Finding nearest neighbors involves identifying points in a dataset that are most similar to a given reference point. FLANN includes multiple search algorithms, with KD-Tree being the one used, which organizes data in space to facilitate quick search.

The combined result of these algorithms is a list representing matches considered sufficiently good based on a specific criterion (0.6 times the distance to the nearest neighbor). The percentage of matches is calculated relative to the minimum number of key points between the two images. If this percentage exceeds the threshold, the images are considered sufficiently similar.

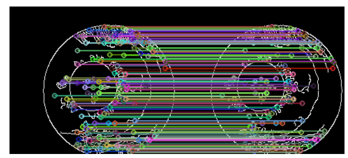
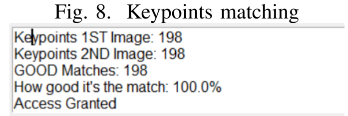


Fig. 8. Keypoints matching



Keypoints 1ST Image: 198
Keypoints 2ND Image: 198
GOOD Matches: 198
How good it's the match: 100.0%
Access Granted

Fig. 9. Results

H. Code Examples

```
def detect_outer_iris(self, image):
    outer_detection = cv2.GaussianBlur(image, ksize: (7, 7), sigmaX: 1)
    outer_detection = cv2.Canny(outer_detection, 20, 70, apertureSize=3)

    hough_circle = cv2.HoughCircles(outer_detection, cv2.HOUGH_GRADIENT, dp: 1.3, minDist: 800)
    if hough_circle is not None:
        hough_circle = np.round(hough_circle[0, :]).astype("int")
        for (x, y, radius) in hough_circle:
            cv2.circle(image, center: (x, y), radius, color: (0, 128, 255), thickness: 4)

1 usage
def detect_inner_iris(self, image):
    inner_detection = cv2.GaussianBlur(image, ksize: (7, 7), sigmaX: 1)
    inner_detection = cv2.Canny(inner_detection, 100, 120, apertureSize=3)

    hough_circles = cv2.HoughCircles(inner_detection, cv2.HOUGH_GRADIENT, dp: 1, minDist: 800,
                                     param1=50, param2=20, minRadius=0, maxRadius=60)
    circles = np.round(hough_circles[0, :]).astype("int")

    for (x, y, r) in circles:
        cv2.circle(image, center: (x, y), r, color: (0, 128, 255), thickness: 2)
        cv2.rectangle(image, (x - 2, y - 2), (x + 2, y + 2), (0, 128, 255), -1)
```

Fig. 10. Detecting iris and pupil

The detectouteriris function takes an image and applies a Gaussian filter to reduce noise, followed by the application of the Canny algorithm for edge detection. Then, the Hough transform is used to identify the outer circle of the iris, and if a circle is found, it is highlighted in the image by drawing a circle with a specific outline.

On the other hand, the detectinneriris function performs a similar process, applying the Gaussian filter and then the Canny algorithm. However, in this case, a different configuration of the Hough transform is used to detect the inner circle of the iris, which is also highlighted in the image by drawing a circle and a small rectangle to mark the center of the detected circle.

```python
def iris_recognition(self):
    file_path = self.file_path.get()

    if file_path:
        image = cv2.imread(file_path, cv2.IMREAD_GRAYSCALE)
        image2 = self.segment_iris_algorithm(image)
        inserted_image = cv2.GaussianBlur(image2, (7, 7), 1)
        inserted_image = cv2.Canny(inserted_image, 20, 70, apertureSize=3)

        threshold = 90
        access_granted = False

        for index in range(1, 8):
            image_to_compare_read = cv2.imread(f'data_set/Iris{index}.jpg', cv2.IMREAD_GRAYSCALE)
            image_to_compare_segmented = self.segment_iris_algorithm(image_to_compare_read)
            image_to_compare = cv2.GaussianBlur(image_to_compare_segmented, (7, 7), 1)
            image_to_compare = cv2.Canny(image_to_compare, 20, 70, apertureSize=3)

            sift = cv2.SIFT_create()
            kp_1, desc_1 = sift.detectAndCompute(inserted_image, None)
            kp_2, desc_2 = sift.detectAndCompute(image_to_compare, None)

            index_params = dict(algorithm=0, trees=5)
            search_params = dict()
            flann = cv2.FlannBasedMatcher(index_params, search_params)

            matches = flann.knnMatch(desc_1, desc_2, k=2)

            good_points = []
            for m, n in matches:
```

```python
                if m.distance < 0.6 * n.distance:
                    good_points.append(m)

            number_keypoints = min(len(kp_1), len(kp_2))

            if len(good_points) / number_keypoints * 100 >= threshold:
                threshold = len(good_points) / number_keypoints * 100
                best_good_points = good_points
                best_kp_1 = kp_1
                best_kp_2 = kp_2
                best_image = image_to_compare
                access_granted = True

        if access_granted:
            result_text = f"Keypoints 1ST Image: {len(best_kp_1)}\n" \
                          f"Keypoints 2ND Image: {len(best_kp_2)}\n" \
                          f"GOOD Matches: {len(best_good_points)}\n" \
                          f"How good it's the match: {threshold}%\n" \
                          "Access Granted"

            result = cv2.drawMatches(inserted_image, best_kp_1, best_image, best_kp_2, best_good_points, None)
            self.show_image(result)
        else:
            result_text = "Access Denied"

        self.result_text.delete(1.0, tk.END)
        self.result_text.insert(tk.END, result_text)
```

Fig. 11. Iris Recognition

This function takes an input iris image, applies segmentation, Gaussian blur, and edge detection to enhance its features. It then iterates through a dataset of pre-processed iris images, employing the Scale-Invariant Feature Transform (SIFT) for keypoint detection and the Fast Library for Approximate Nearest Neighbors (FLANN) for feature matching. The matching process involves assessing the quality of keypoint matches and determining the percentage of keypoints that contribute to a successful match. Access is granted if the best match surpasses a specified threshold, and denial occurs if no suitable match is found. The function provides detailed information on the quality of the match, including the number of keypoints, good matches, and the overall match percentage.

IV. Results

We managed to create a Python program with a simple graphical user interface using Tkinter library, OpenCv for image processing and PIL(Pillow) library for image display, in which we applied the method described above, the initial image displaying the original iris image with circles detected in both passes of the Hough Circle Transform. After grayscaling and applying the gaussian filter, Canny's Edge Detection algorithm and Hough Transform were applied twice using different parameters for a better optimization. The application allows users to load an image, detect the iris and pupil, segment the iris and perform the iris recognition using Scale-Invariant Feature Transform(SIFT) algorithm.

V. Conclusions

The Hough transform can be used for iris detection due to its circular structure. The definition and the right following up of the process suitable for image processing facilitate the detection and segmentation of the iris.

The algorithm determination to be implemented to achieve segmentation must consider conditions in which the images were acquired and in accordance with the databases used. In this way, each phase of the process will fulfill its function and will contribute to improve the image and to remove elements that are not relevant for the segmentation process.

While we have achieved success in pupil and iris segmentation, challenges persist in handling eyelid occlusion. The algorithm struggles with accurate segmentation when eyelids partially or fully cover the eye. Eyelid occlusion poses difficulties in identifying the complete iris region, leading to inaccuracies in the segmentation process.

The provided code for the iris recognition application successfully incorporates essential functionalities, such as iris segmentation, contour detection, and recognition using SIFT keys. Here are some key conclusions:

1) User-Friendly Graphical Interface: The application features an intuitive and user-friendly graphical interface, facilitating image loading, result visualization, and interaction with the provided functionalities.

2) Iris Segmentation: The method implemented for iris segmentation successfully employs GaussianBlur, Canny Edge Detection, and the Hough transform, providing a clear and precise visual representation of the iris in the image.

3) Circle and Iris Contour Detection: The algorithms for circle detection and iris contouring are efficient, emphasizing the importance of techniques such as GaussianBlur and HoughCircles for accurate iris identification in the image.

4) SIFT Recognition and Image Comparison: The SIFT recognition section and image comparison using SIFT keys add an additional level of security and authentication, highlighting the complexity of the iris recognition process.

5) Ease of Image Integration: The ability to load images into the application, either through a local file, adds flexibility and ease in exploring and testing functionalities.

References

[1] Neha Kak, Rishi Gupta, Sanchit Mahajan, (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 1, 2010.

[2] Tossy Thomas, Anu George, Dr. K P Indira Devi, "Effective Iris Recognition System".

[3] Deshpande, A. and Patavardhan, P.P. (2017) Super Resolution and Recognition of Long Range Captured Multi-Frame Iris Images. IET Biometrics.

[4] Radman, A., Zainal, N. and Suandi, S.A. (2017) Automated Segmentation of Iris Images Acquired in an Unconstrained Environment Using HOG-SVM and GrowCut. Digital Signal Processing, 64, 60-70.

[5] Lee, J., Tang, H. and Park, J. (2018) Energy Efficient Canny Edge Detector for Advanced Mobile Vision Applications. IEEE Transactions on Circuits and Systems for Video Technology, 28, 1037-1046.

[6] Koh, J., Govindaraju, V. and Chaudhary, V. (2010) A Robust Iris Localization Method Using an Active Contour Model and Hough Transform. 20th International Conference on Pattern Recognition, Istanbul, 23-26 August 2010, 2852-2856.

[7] CASIA Iris Image Database (2013).

[8] P. P. Paul and M. M. Monwar, Human Iris Recognition for Biometric, 2001.

[9] Xiaoqiang Wu and Long ZhaoStudy, Iris segmentation Algorithm based on Dense U-Net

[10] Lowe, David G. (2004). "Distinctive Image Features from Scale-Invariant Keypoints". International Journal of Computer Vision. 60 (2): 91–110.