International Journal of Scientific Research in Computer Science, Engineering and Information Technology

ISSN : 2456-3307

OPEN ACCESS

Available Online at : www.ijsrcseit.com

doi : https://doi.org/10.32628/IJSRCSEIT

Fingerprint Recognition and Verification using Fourier Domain Filtering and Histogram Equalization Techniques

[1]G. Nancharaiah, [2]G. Sai Teja Kumari, [3]K.Lakshmi, [4]J. Harika, [5]B.Srinu

[1]Assistant Professor, [2,3,4,5] UG Student

Department of CSE, Sri Vasavi Institute of Engineering and Technology, Nandamuru, Andhra Pradesh, India

ARTICLEINFO

Article History:

Accepted: 10 April 2024

Published: 22 April 2024

Publication Issue

Volume 10, Issue 2

March-April-2024

Page Number

698-704

ABSTRACT

Fingerprint Recognition is a vital method in biometric identification and verification of human beings in various domains like Security, Digital Forensics, Internet of Things (IoT), and many more. Each individual human is having distinct fingerprint pattern than others, hence it is one of the most prominent and widely used method to distinguish individuals. Many research studies and solutions have been developed in biometric domain since a decade, which influences now in making the process of fingerprint recognition more optimized, faster and efficient. However, present fingerprint acquisition/recognition systems have some limitations, mainly longer computation time for fingerprint matching and evaluating the results. This paper presents a procedure for fingerprint matching that takes into account minutiae features in finger print images and the process of creating an OpenCV structure for minutiae extraction and matching of fingerprints.

Keywords: Fingerprint, OpenCV, Biometric, Security and Authentication, Minutiae

I. INTRODUCTION

In contemporary world, biometric authentication is considered to be the most reliable authentication or access techniques followed by most corporates and offices. Biometric authentication technique uses human biological features such as face, fingerprint, retinal image, voice and speech, walking gait, veins and so on. In this, the most commonly and widely used biometric is fingerprint. Fingerprint is considered widely for authentication, because it is unique and permanent throughout a person's lifetime. Therefore, fingerprint of humans are used for pinpointing of culprits, personal identification object in official documents, authorizing access to a secured environment, etc. Most of the fingerprint recognition methods are based on the characteristics of local ridges, which are also called as minutiae, and the relationship among the ridges. In fingerprint scanning, a minutiae refers to specific plot points of a fingerprint. In general,

any fingerprint identification system has to capture the image of fingerprint and perform a similarity analysis with the stored images in the dataset that contains fingerprint images of a group of individuals.

Fingerprint Recognition is a vital method in biometric identification and verification of human beings in various domains like Security, Digital Forensics, Internet of Things (IoT), and many more. Each individual human is having distinct fingerprint pattern than others, hence it is one of the most prominent and widely used method to distinguish individuals. So it's very important for us to distinguish individuals for security point of view.

Many research studies and solutions have been developed in biometric domain since a decade, which influences now in making the process of fingerprint recognition more optimized, faster and efficient. However, present fingerprint acquisition/recognition systems have some limitations, mainly longer computation time for fingerprint matching and evaluating the results. This paper presents a procedure for fingerprint matching that takes into account minutiae features in finger print images and the process of creating an OpenCV structure for minutiae extraction and matching of fingerprints.

II.RELATED WORK

Mohamed Hedi et al. (2017) presented a new minutiae based matcher that employs Expanded Delaunay Triangulation (EDT) method to yield local structures related to minutiae. The method also applies innovative steps to extract new feature called minutiae triplets. This technique is also more robust against displacement of minutiae while processing the images compared to other methods that relies on sides' length. Ultimately, to fasten the matching activity, the EDC technique uses an improvised merge-join process to fasten the identification of local structures. Jacob, I. Jeena (2019) discussed about capsule network based biometric. Deep learning is one of an eye storming techniques in the present generation. This deep learning technique is being used in biometric recognitions as it enhances accuracy and for automatically learning. The disadvantage in conventional neural network is got replaced by this capsule neural network as this method overcomes ability of removing noise which degrades accuracy that are caused due to small disturbances. This capsule network method is utilized with the fuzzified image for retina based biometric recognition. This is found 99% accurate. Raj,jennifer S (2019) discussed about the benefits of computational intelligence. Artificial intelligence is the recent trending way where a machine is trained to have an intelligence artificially like searching, planning etc. It faces some failures in predicting in certain areas .These failures of AI makes a path for computational intelligence. So for overcoming failures and draw backs in AI, this paper has given some computational intelligence techniques. Gerald P et a. (2014) proposed a method of checkerboard sampling algorithm using ANN for partial fingerprint identification. Authors claim that this work can save memory space in the database. It also introduces an encryption technique for the whole fingerprint. Sun Bei et al. (2016) presented a method that accomplishes a decent classification performance of imperfect or partially corrupted fingerprint images. The method involves two steps. Initially a skeleton model of corrupted fingerprint in created using preprocessing techniques. Secondly, a finger print verification system is created that identifies the core point of the corrupted fingerprint and recreates or models the trivial features called minutiae. Thirdly, a neural network architecture is developed for classification of fingerprints.

III.PROPOSED SYSTEM

In making the process of fingerprint recognition more optimized, faster and efficient. However, present fingerprint acquisition/recognition systems have some limitations, mainly longer computation time for fingerprint matching and evaluating the results. This

paper presents a procedure for fingerprint matching that takes into account minutiae features in finger print images and the process of creating an OpenCV structure for minutiae extraction and matching of fingerprints.

Advantages:

- Proposed system is focused on extracting minutiae features from fingerprint.
- It used for both i.e. for minutiae extraction and matching of fingerprints.

In this section, the developed fingerprint verification system and algorithms used to implement the preprocessing, segmentation, feature extraction and selection have been discussed. It also presents detailed description of the verification technique. Figure 2 shows the block diagram of the proposed fingerprint verification system. It consists mainly of five phases: Preprocessing, Segmentation, Feature Extraction, Feature Matching and Analysis as it will be explained in detail shortly.

Fig 1: Fingerprint verification system block diagram

IV.RESULTS AND DISCUSSION

```
import cv2
import numpy as np
import skimage.morphology
from skimage.morphology import convex_hull_image, erosion
from skimage.morphology import square
import math
import glob
import os

class MinutiaeFeature(object):
    def __init__(self, locX, locY, Orientation, Type):
        self.locX = locX;
        self.locY = locY;
        self.Orientation = Orientation;
        self.Type = Type;

class FingerprintFeatureExtractor(object):
    def __init__(self):
        self._mask = []
        self._skel = []
        self.minutiaeTerm = []
        self.minutiaeBif = []
```

```
def __skeletonize(self, img):
    img = np.uint8(img > 128)
    self._skel = skimage.morphology.skeletonize(img)
    self._skel = np.uint8(self._skel) * 255
    self._mask = img * 255

def __computeAngle(self, block, minutiaeType):
    angle = []
    (blkRows, blkCols) = np.shape(block);
    CenterX, CenterY = (blkRows - 1) / 2, (blkCols - 1) / 2
    if (minutiaeType.lower() == 'termination'):
        sumVal = 0;
        for i in range(blkRows):
            for j in range(blkCols):
```

Fig 2. Results screenshot

```
    def __getTerminationBifurcation(self):
        self._skel = self._skel == 255;
        (rows, cols) = self._skel.shape;
        self.minutiaeTerm = np.zeros(self._skel.shape);
        self.minutiaeBif = np.zeros(self._skel.shape);

        for i in range(1, rows - 1):
            for j in range(1, cols - 1):
                if (self._skel[i][j] == 1):
                    block = self._skel[i - 1:i + 2, j - 1:j + 2];
                    block_val = np.sum(block);
                    if (block_val == 2):
                        self.minutiaeTerm[i, j] = 1;
                    elif (block_val == 4):
                        self.minutiaeBif[i, j] = 1;

        self._mask = convex_hull_image(self._mask > 0)
        self._mask = erosion(self._mask, square(5))  # Structuing element for mask erosion = square(5)
        self.minutiaeTerm = np.uint8(self._mask) * self.minutiaeTerm

    def __removeSpuriousMinutiae(self, minutiaeList, img, thresh):
        img = img * 0;
        SpuriousMin = [];
        numPoints = len(minutiaeList);
        D = np.zeros((numPoints, numPoints))
        for i in range(1,numPoints):
            for j in range(0, i):
                (X1,Y1) = minutiaeList[i]['centroid']
                (X2,Y2) = minutiaeList[j]['centroid']

                dist = np.sqrt((X2-X1)**2 + (Y2-Y1)**2);
                D[i][j] = dist
                if(dist < thresh):
                    SpuriousMin.append(i)
                    SpuriousMin.append(j)
```

Fig 3. Results screenshot

```
    def __cleanMinutiae(self, img):
        self.minutiaeTerm = skimage.measure.label(self.minutiaeTerm, connectivity=2);
        RP = skimage.measure.regionprops(self.minutiaeTerm)
        self.minutiaeTerm = self.__removeSpuriousMinutiae(RP, np.uint8(img), 10);

    def __performFeatureExtraction(self):
        FeaturesTerm = []
        self.minutiaeTerm = skimage.measure.label(self.minutiaeTerm, connectivity=2);
        RP = skimage.measure.regionprops(np.uint8(self.minutiaeTerm))

        WindowSize = 2  # --> For Termination, the block size must can be 3x3, or 5x5. Hence the window selected is 1 or 2
        FeaturesTerm = []
        for num, i in enumerate(RP):
            (row, col) = np.int16(np.round(i['Centroid']))
            block = self._skel[row - WindowSize:row + WindowSize + 1, col - WindowSize:col + WindowSize + 1]
            angle = self.__computeAngle(block, 'Termination')
            if(len(angle) == 1):
                FeaturesTerm.append(MinutiaeFeature(row, col, angle, 'Termination'))

        FeaturesBif = []
        self.minutiaeBif = skimage.measure.label(self.minutiaeBif, connectivity=2);
        RP = skimage.measure.regionprops(np.uint8(self.minutiaeBif))
        WindowSize = 1  # --> For Bifurcation, the block size must be 3x3. Hence the window selected is 1
        for i in RP:
            (row, col) = np.int16(np.round(i['Centroid']))
            block = self._skel[row - WindowSize:row + WindowSize + 1, col - WindowSize:col + WindowSize + 1]
            angle = self.__computeAngle(block, 'Bifurcation')
            if(len(angle) == 3):
                FeaturesBif.append(MinutiaeFeature(row, col, angle, 'Bifurcation'))
        return (FeaturesTerm, FeaturesBif)
```

Fig 4. Results screenshot

```
        img_dir = 'C:\\Users\\User\\Downloads\\Fingerprint-Feature-Extraction-master\\enhanced\\'
        os.chdir(img_dir)
        name = input('Enter the name of person : ')
        img = name + '.jpg'
        cv2.imwrite(img,DispImg)

        a = cv2.imread("C:\\Users\\User\\Downloads\\Fingerprint-Feature-Extraction-master\\enhanced\\xyz.jpg")
        b = cv2.imread("C:\\Users\\User\\Downloads\\Fingerprint-Feature-Extraction-master\\Fingerprint_data\\xyz.jpg")
        difference = cv2.subtract(a, b)
        result = not np.any(difference)
        if result is True:
            print("Fingerprint Matched")
        else:
            print("Fingerprint Doesn't Match !!!!!!")

        cv2.imshow('a', DispImg);
        cv2.waitKey(0)


def extract_minutiae_features(img, showResult=False):
    feature_extractor = FingerprintFeatureExtractor()
    FeaturesTerm, FeaturesBif = feature_extractor.extractMinutiaeFeatures(img)

    if(showResult):
        feature_extractor.showResults()

    return(FeaturesTerm, FeaturesBif)
```

Fig 5. Results screenshot

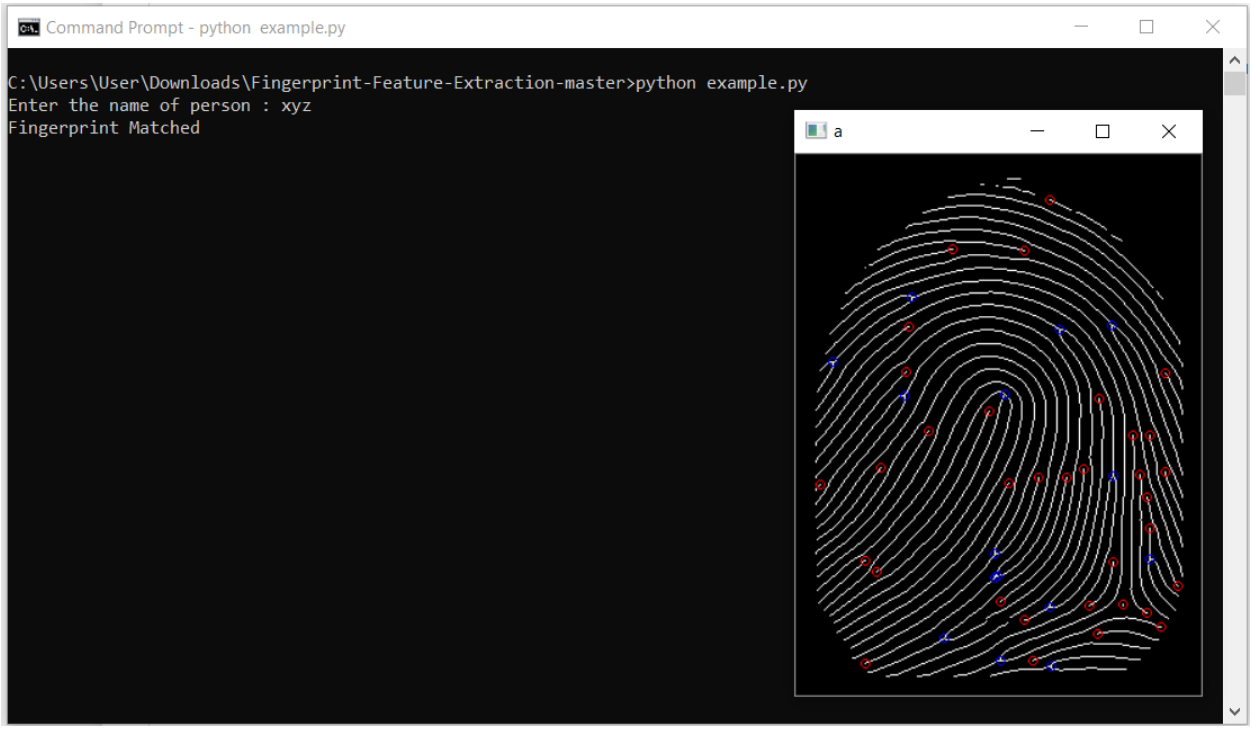

Fig 6. Results screenshot

V. CONCLUSION

The proposed system has designed architecture for near real-time fingerprint recognition with minutiae extraction. The proposed method involves feature extraction and feature matching of the biometric fingerprint images using minutiae point score. The implemented prototype, works on OpenCV Framework.

VI. FUTURE WORK

The implemented prototype, works on OpenCV Framework and can be further improved and tested with standard fingerprint datasets such as FVC2006 and Sokoto Coventry. These minutiae points of each image from local images database are compared with real-time captured biometric fingerprint's minutiae points and based on the score of matching result is produced for matching. We can implement proposed method of minutiae matching and recognition on RaspberryPi-3 hardware platform.

II. REFERENCES

[1]. Soonrutha Karothu, Vineeta Tiwari, Siddhant Verma, M. Sugadev. "Biometric methods to speed up fingerprint processing using OCT: A survey."International Journal of Advance Research, Ideas and Innovations in Technology 5.1 (2019)..

[2]. Gerald P. Arada1, Elmer P. Dadios2,"Partial Fingerprint Identification through Checkerboard Sampling Method Using ANN" IEEE transactions on services computing, vol. 7, no. 4, 2014.

[3]. Sun Bei1, Luo Wusheng1,, and Du Liebo1, Lu Qin1, "A fingerprint identification algorithm based on local minutiae topological property", 2016 IEEE First International Conference on Data Science in Cyberspace.

[4]. Le, Hong Hai, Ngoc Hoa Nguyen, and Tri-Thanh Nguyen. "Speeding up and enhancing a large-scale fingerprint identification system on GPU."

Journal of Information and Telecommunication,2(2), (2018), 147-162.

[5]. Sagayam, Martin & Narain Ponraj, D & Winston, J & Yaspy, J.C. & Jeba, D & Clara, A. (2019). Authentication of biometric system using fingerprint recognition with Euclidean distance and neural network classifier. International Journal of Innovative Technology and Exploring Engineering, Vol.8. ,766-771.

[6]. Ali, Mouad & Mahale, Vivek & Yannawar, Pravin & Gaikwad, Ashok. (2016). Fingerprint Recognition for Person Identification and Verification Based on Minutiae Matching. 332-339. 10.1109/IACC.2016.69.

[7]. K. Cao and A. K. Jain, Automated latent fingerprint recognition, IEEE Trans. Pattern Analysis and Machine Intelligence, vol.41, no.4, pp.788-800, 2018.

[8]. S. Guennouni, A. Mansouri and A. Ahaitouf, Biometric systems and their applications, in Eye Tracking and New Trends, IntechOpen, 2019.

[9]. M. S. Mahmood, Fingerprint identity watermark to authenticate digital camera images, Advances in Natural and Applied Sciences, vol.11, no.9, pp.117-126, 2017.

[10]. C. Champod, C. J. Lennard, P. Margot and M. Stoilovic, Fingerprints and Other Ridge Skin Impressions, CRC Press, 2017.

[11]. D. Peralta, I. Triguero, S. Garc´ıa, Y. Saeys, J. M. Benitez and F. Herrera, Robust classification of different fingerprint copies with deep neural networks for database penetration rate reduction, arXiv Preprint, arXiv:1703.07270, 2017.

[12]. M. M. Ali, V. H. Mahale, P. Yannawar and A. T. Gaikwad, Fingerprint recognition for person identification and verification based on minutiae matching, IEEE the 6th International Conferenceon Advanced Computing, pp.332-339, 2016.

[13]. M. Xu, J. Feng, J. Lu and J. Zhou, Latent fingerprint enhancement using Gabor and minutia dictionaries, IEEE International Conference on Image Processing (ICIP), pp.3540-3544, 2017.

[14]. S. Sindhu and B. Arunadevi, Fingerprint authentication based on adaptive greedy registration of minutiae pairs, The 2nd International Conference on Trends in Electronics and Informatics (ICOEI), pp.1360-1364, 2018.

[15]. Y. Xu, G. Lu, Y. Lu, F. Liu and D. Zhang, Fingerprint pore comparison using local features and spatial relations, IEEE Trans. Circuits and Systems for Video Technology, 2018.

[16]. FVC 2000, http://bias.csr.unibo.it/fvc2000/db1.asp.

[17]. FVC 2000, http://bias.csr.unibo.it/fvc2000/db2.asp.

[18]. FVC 2000, http://bias.csr.unibo.it/fvc2000/db3.asp.