Ukrainian Catholic University

Faculty of Applied Sciences

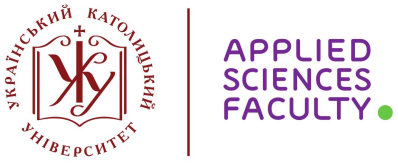
Business Analytics & Computer Science Programmes

Fingerprint recognition with Principal Component Analysis

Linear Algebra final project report

Authors:
Dmytro Strus
Taras Martsin

April 2024



1 Familirization

1.1 Introduction

Fingerprint identification stands as a cornerstone in biometric technology, offering a highly accurate and widely adopted means of personal authentication. Despite being one of the oldest biometric technologies, fingerprint recognition continues to be prevalent due to its reliability and simplicity. In the contemporary era, where digital security is paramount, the development of effective fingerprint recognition systems holds critical significance. In our project, we aim to implement a fingerprint recognition system using Principal Component Analysis (PCA). This report outlines our goals, approaches, and progress in achieving accurate fingerprint recognition. aiming to contribute to advancements in security protocols, medical forensics, and access control mechanisms.

This technology is used in various areas such as:

1. Digital security (in smartphones, laptops, access control systems)
2. Law enforcement and forensics (for suspect identification, fingerprints are valuable forensic evidence in criminal investigations)
3. Border control and immigration (at border crossings and immigration checkpoints to verify the identity of travelers)
4. Financial transactions. (as an additional layer of security for online banking, electronic payments and ATM transactions) Choosing fingerprint recognition over other biometric technologies such as facial or iris recognition has several advantages: this method is highly accurate and reliable in identifying individualis, it's non-invasive, individuals typically are comfortable with providing their fingerprints for authentication purposes and it's also cost-effective which can be a major factor for large deployments such as border control or access control applications. In our project, we aim to implement a fingerprint recognition system, with the primary metric for success being the percentage of accurately identified fingerprint images. This report outlines our goals, approaches, and progress in achieving accurate fingerprint recognition. aiming to contribute to advancements in security protocols, medical forensics, and access control mechanisms.

1.2 Problem Statement

Prior research has explored various techniques for fingerprint recognition, including minutiae-based approaches, orientation field analysis, and texture-based methods. Our focus is on PCA coupled with orientation field analysis for core points detection, offering robustness to noise and low-quality images. To achieve this goal, we plan to study and implement various mathematical algorithms for all necessary stages of fingerprint processing.

1.3 Related Works

In our quest to develop an effective fingerprint recognition system, we have delved into existing literature to understand the landscape of fingerprint processing and recognition techniques. Our review has uncovered a variety of approaches and methodologies employed by researchers in this field. "A Fingerprint Recognition Algorithm Based on

Principal Component Analysis" focuses on PCA algorithm development. The proposed algorithm aims to address the authentication aspect of fingerprint recognition, specifically concerned with verifying whether a given fingerprint image belongs to a stored database. The paper outlines the algorithm's steps, including database storage, PCA-based image space construction, classification criteria definition, and decision-making based on thresholds derived from a new parameter "H", whose main purpose is controlling false positives and false negatives.

Here's how their algorithm works briefly:

1. **Region of Interest (ROI) Extraction**:
   (a) In a fingerprint image, singular points are crucial, often used as proof of classification and located at the central position in the image. Identical areas around singular points cannot be found in different fingerprints. The algorithm locates the central position and establishes the direction of ROI based on singular points.
2. **Feature Extraction with PCA**:
   (a) After extracting the ROI from fingerprint images, PCA is applied to extract statistical features from these regions. Each fingerprint's ROI is treated as a vector, which is then projected into a matrix using PCA. This process transforms the high-dimensional feature space into a lower-dimensional subspace while retaining as much information as possible.
   (b) PCA identifies the most significant patterns or directions of variation within the ROI dataset and represents these patterns as principal components.
3. **Dimensionality Reduction**:
   (a) The principal components are ranked based on the amount of variance they explain in the original dataset. The algorithm selects a subset of the most significant principal components to represent the fingerprint features.
   (b) By retaining only the most relevant principal components, the dimensionality of the feature space is reduced, making the data more manageable while preserving the essential characteristics of the fingerprints.
4. **Feature Matching and Verification**:
   (a) Once the feature vectors are obtained through PCA, the algorithm computes the Euclidean distance between these vectors to measure the similarity between different fingerprints.
   (b) A predefined threshold is used to determine whether two fingerprints are considered a match based on their feature vectors. If the distance falls below the threshold, the fingerprints are identified as belonging to the same individual; otherwise, they are classified as different.

This method is overall good. However there are some cons: this method is dependent on central position in the fingerprint, it will work poorly if the central of the image is of poor quality and it won't work for matrices more than 1000×1000 This method would work well if the center of the image isn't distorted and the image itself transforms to

a matrix less than 1000×1000, but since our dataset may contain truncated images and the size of them is well beyond the limit for this approach (about $200K - 250K \times 200K - 250K$), the authors decided to use core point detection instead and then apply PCA for fingerprint recognition. The similarities of the approaches are regarding the dimensionality reduction, while retaining only the most relevant principal components; feature matching and verification, relying on euclidean distance and predefined threshold sigma the final goal, as both approaches represent each fingerprint's unique characteristics in a lower-dimensional feature space. There are differences too, for example, there is no data standardization before applying PCA, which allows us to work with larger datasets.

1.4 Possible approaches

Possible approaches for core points detection:

Minutiae-Based Approach:

Overview: The minutiae-based approach focuses on extracting and analyzing minutiae points, which are local ridge characteristics such as ridge endings and bifurcations. These minutiae points serve as distinctive features for fingerprint matching.

Algorithm: The Crossing Numbers (CN) algorithm is commonly employed for minutiae extraction. This algorithm scans the fingerprint image and identifies minutiae points based on the local topology of ridges (raised lines or patterns present on the surface of the skin).

Orientation Field-Based Approach:

Overview: The orientation field-based approach involves analyzing the orientation field of fingerprints to identify singularities, which often correspond to core points, is another promising avenue.

Algorithm: Techniques such as phase-based analysis or directional filtering are commonly employed for singularity detection in the orientation field. Phase-based analysis involves computing the phase of the Fourier transform of the fingerprint image to identify regions of rapid phase change, which indicate singularities.

The decision in favor of Orientation Field-Based approach was made because:

- It performs better in cases of poor image quality or obscured fingerprints
- Provides invariance to translation and rotation of the fingerprint
- Is more sufficient for large datasets
- It is more robust to certain types of distortions, such as smudging or partial fingerprint impressions

Possible approaches for fingerprint recognition:

PCA (Principal Component Analysis):

- PCA is a dimensionality reduction technique used to reduce the complexity of high-dimensional data while preserving most of its important features. In the context of fingerprint recognition, PCA can be applied to reduce the dimensionality of fingerprint images or feature vectors extracted from fingerprints, making subsequent processing and matching more efficient.

SIFT (Scale-Invariant Feature Transform):

- SIFT is a feature detection and description algorithm used in computer vision tasks, such as object recognition and image stitching. While not specifically designed for fingerprint recognition, SIFT can be employed to detect and describe distinctive local features in fingerprint images, which can then be used for matching.

ICA (Independent Component Analysis):

- ICA is a statistical technique used to separate a multivariate signal into additive, independent components. In the context of fingerprint recognition, ICA can potentially be used for feature extraction or dimensionality reduction, similar to PCA. It aims to find underlying independent sources of variability in the data.

The decision in favor of PCA approach was made because:

- It is computationally efficient compared to techniques like SIFT and ICA, especially when dealing with large datasets. It involves straightforward linear transformations, making it faster to compute compared to the more complex algorithms like SIFT or ICA.
- It captures the global structure of the data by identifying the principal components that explain the maximum variance. In fingerprint recognition, this can be advantageous if the overall shape or structure of the fingerprint is important for matching, rather than local features, also reaching noise reduction, for instance.

2 Methodology

2.1 Chosen Algorithm

We've chosen an algorithm from papers "An algorithm for fingerprint core point detection" by Atipat Julasayvake* and Somsak Choomchuay* and "A Novel Method for Fingerprint Core Point Detection" by Navrit Kaur Johal, Prof. Amit Kamra, which relies on the orientation-field based approach for core detection. Overall, the authors have the combined algorithm of finger proccesing in order to crop based on optimal core points detection and orientation field estimation to get the image with only useful and important information and then apply PCA on this cropped image for further matching and evaluation of recognized fingerprints.

We selected PCA (Principal Component Analysis) for its ability to reduce dimensionality while preserving essential information. It is a dimensionality reduction technique commonly used in data analysis and machine learning. PCA aims to transform the data into a new coordinate system where the axes (principal components) are orthogonal to each other and are ordered by the amount of variance they explain in the original data.

Pros:

1. Dimensionality Reduction (effectively reduces the number of features (dimensions) in the dataset while retaining most of the important information)
2. Decorrelation of Features (transforms the original features into a set of orthogonal components, eliminating multicollinearity between them)
3. Noise Reduction (filter out noise present in the data, by focusing on the principal components that capture the most variance)

Cons:

1. Possibility of important information loss
2. Assumption of Linearity (assumes that the data can be effectively represented as a linear combination of the principal components)
3. Sensitivity to Scaling

The pros and cons choosing finding optimal core point based on orientation field estimation and cropping was decsribed above .

2.2 Data

Our dataset consists of 50 folders - people, inside of each one there are 5 pictures of different fingers of size 200×200. First procedure the authors applied on it - core point detection on every single finger image to get a dataset with cropped images. Both of these datasets will serve us to test the efficiency of our algorithms: PCA of default/cropped finger on default/cropped dataset will provide all the necessary information about accuracy and time consumption, which is also preferred. When assessing the similarity between an uploaded fingerprint and those in a standard dataset, the goal is to determine if their distance is below a predefined threshold value, σ. Each image in the dataset, along with the user's uploaded desired picture, is transformed into a real-valued vector in R^m, where m equals the square of the image's size. These vectors are then organized into a data matrix of size $l \times m$, where l is the number of samples in the dataset.

First of all the image is processed to obtain optimal core point. To this end, the image passes few steps in order to estimate this core point. Then after having the core point of fingerprint image, the region of interest near the optimal core points is cropped and the image size reduces having only region of interest of fingerprint. This process is done in order to have only the useful and unique information about the exact fingerprint for further transformation in PCA

3 Theoretical Background

3.1 Core Points detection

Our algorithm based on papers "An algorithm for fingerprint core point detection" by Atipat Julasayvake* and Somsak Choomchuay* and "A Novel Method for Fingerprint Core Point Detection" by Navrit Kaur Johal, Prof. Amit Kamra. All references for formulas and values will be based on these two papers

The algorithm consists of few steps:

Firstly, the image is transformed into a matrix of size m*n, where the value in $I(i,j)$ represents the grayscale(intensity) value, (with range from 0-255 where 0 is black).

The next step is image normalization where normalization is a process that changes the range of pixel intensity value to deal with very dark and very light pictures with desired mean $M_0 = 50$ and variance $V_0 = 50$ according to optimal technique described in papers for 320×320 size images. Here M_i and V_i are the estimated mean and variance of matrix $I(i,j)$, and the resulting normalized matrix is denoted as $N(i,j)$ Note that I and N is image and normalized image and $I(i,j)$ and $N(i,j)$ is pixel of i row and j column in image and normalized image.

$$N(i,j) = \begin{cases} M_0 + \sqrt{\frac{V_0(I(i,j)-M_i)^2}{V_i}} & \text{if } I(i,j) > M \\ M_0 - \sqrt{\frac{V_0(I(i,j)-M_i)^2}{V_i}} & \text{otherwise} \end{cases}$$

$$M_i = \frac{1}{MN} \sum_{i-0}^{M-1} \sum_{j=0}^{N-1} I(i,j)$$

$$V_i = \frac{1}{MN} \sum_{i-0}^{M-1} \sum_{j=0}^{N-1} (I(i,j) - M(I))^2$$

After this we need to estimate the orientation field, which represents the direction in which ridges (lines, curves) are oriented in an image. denoting $\theta(i,j)$ as the local ridge at pixel (i,j), however as it's usually specialized for a block rather than each pixel, we'd first need to divide our image into these blocks. Thus, an image is divided into a set of non-overlapping blocks, size of $w \times w$ ($w = 20$ was estimated as optimal block for further computing). We compute the gradient at each pixel (i,j) which is the center of the block, using the Sobel operator for ∂_x and ∂_y . Now we estimate the local orientation using the following formulas.

$$V_x(i,j) = \sum_{u=i-\frac{w}{2}}^{i+\frac{w}{2}} \sum_{v=j-\frac{w}{2}}^{j+\frac{w}{2}} 2\partial_x(u,v)\partial_y(u,v)$$

$$V_y(i,j) = \sum_{u=i-\frac{w}{2}}^{i+\frac{w}{2}} \sum_{v=j-\frac{w}{2}}^{j+\frac{w}{2}} \partial_x^2(u,v)\partial_y^2(u,v)$$

Where $V_x(i,j)$ and $V_y(i,j)$ represent represent gradient magnitudes in X and Y directions respectively with $\partial_x(u,v)$ and $\partial_y(u,v)$ which gradients at each pixel (i,j) which is the center of the block.

Then we estimate the orientation for each batch using the arctangent function applied to the ratio of gradients along the y and x axes of the block centered at pixel (i ,j).

$$\theta(i,j) = \frac{1}{2}\tan^{-}1\left(\frac{V_y(i,j)}{V_x(i,j)}\right)$$

Here, the $\theta(i,j)$ is the least square estimate of the local ridge orientation of the block centered at pixel (i,j) (θ is a matrix of radians).

Example of obtained images from literature for original image's size 320×320

Figure 1: Intermediate obtained images: (a) original image (b) Local field estimation (image after normalization with $M_0 = 50$ and $V_0 = 50$) (c) after orientation field fine-tune estimation

Next step is to crop image with some specified size near optimal core point. While estimating orientation field in every orientation block $\theta(i,j)$ of size $w \times w$ the difference of direction components is computed by formulas

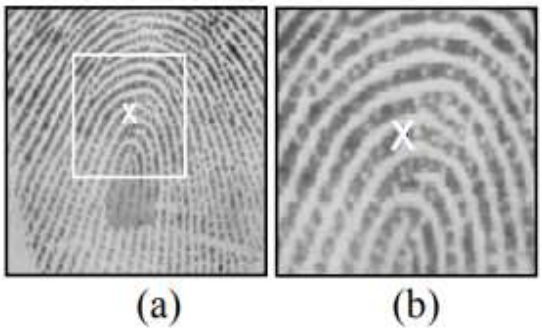
$$Diff\ Y = \sum_{k=1}^{w} \sin 2\theta(k,w) - \sum_{k=1}^{w} \sin 2\theta(k,1) \tag{1}$$

$$Diff\ X = \sum_{l=1}^{w} \cos 2\theta(w,l) - \sum_{l=1}^{w} \cos 2\theta(1,l) \tag{2}$$

And if $Diff\ X$ and $Diff\ Y$ both are negative then the potential core point is located with i and j corresponding to x and y coordinates in original image (not block). Having not only one potential core point the optimal core point is chosen in way that the distance from this core point is the smallest to other core points in order to crop the image which will have in center the optimal core point and save the most useful information about fingerprint

Optimal size for cropping is to get 150×150 cropped image taking into acoount dataset's images with size 200×200. Such cropping size reducing original image size by 25% and is optimal because too small size will lead to data loss and too big size can cause inefficient algorithm work of algorithm.

Example of cropped images from literature for original image's size 320×320



After getting cropped image(s), Principal Components Analysis (PCA) is applied to compute the principal components, $pc_1, pc_2, \ldots pc_k$, forming an orthonormal set of vectors. These principal components serve to represent the data in a lower-dimensional space without significant information loss. Each image vector, imi, is then projected onto the new basis formed by $pc_1, pc_2, \ldots . pc_k$ using a transition matrix, T, obtained by transposing the matrix composed of principal components. This transformation enables working with vectors inR^k, where Euclidean distance is well-defined. To verify if the uploaded image corresponds to a known fingerprint, the average distance between the uploaded image and those in the original dataset is calculated. In order to find desire fingerprint's owner one finds minimal distance

3.2 Principal Component Analysis

Firstly, PCA starts with Data Standardization, where the features of samples are shifted to zero mean. This is crucial for ensuring that all features contribute equally to the analysis, irrespective of their original scales.

$$X_s = X - \frac{1}{n}\sum_{i=1}^{n} x_i$$

Where X_s - matrix after deducting the mean, X - sample matrix

Secondly, PCA involves the computation of the Covariance Matrix. This matrix summarizes the relationships between different features in the standardized data. It provides insights into how features vary together

$$S = \frac{(X_s^T \cdot X_s)}{(n-1)}$$

Where S is covariance matrix

Next, PCA conducts Eigenvalue Manipulations by computing the eigenvectors and eigenvalues of the covariance matrix, using Power Method.

$$Sv = \lambda v$$

$$v_k = \frac{Sv_{k-1}}{\|Sv_{k-1}\|}$$

Where S is the covariance matrix, v is associated eigenvector, λ is scalar eigenvalue

After obtaining the eigenvectors and eigenvalues, PCA performs Result Preparation. This involves rearranging the eigenvectors and eigenvalues, typically in descending order

of eigenvalues. Subsequently, a subset of eigenvectors, representing the principal components, is selected based on the desired number of dimensions for dimensionality reduction. Finally, the data is projected onto these projector, a linear subspace $R^{m \times m}$, whose basis is formed by selected principal components), effectively reducing its dimensionality while retaining as much variance as possible.

$$B = X^T P$$

Where P is matrix obtained from eigenvectors of X_s, representing its principal components and X is the initial matrix.

After that we project desired finger onto P and compute the Euclidean distance in order to find the most similar finger of all in matrix X. The minimal distance will represent the most possible owner.

$$D = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Where D is euclidean distance

4 Implementation

Our implementation process will be devided into some steps.

- Data pre-processing
- PCA computation
- Selecting on number of components and appropriate treshold
- Testing overall accuracy of the code

4.1 Implementation nuances

We selected **Python** language for our implementation. Althoug it is not the most convinient and efficient in terms of computation, it provides a lot of packages for small tasks, such as matrix transposition or multiplication. As for packages authors selected **Numpy**. For Dataset we selected **Fingerprint Color Image Database** by MathWorks. Also we will use 100×100 images for testing, because computation of larger images will take a lot of time. Source code along with all necessary files can be found in the appendix section, as well as the intermediate results of core points procedure and PCA.

4.2 Data pre-processing

To crop our dataset images based on optimal core point detection we implemented algorithm based on described literature and theorethical background. We used OpenCV library to work with photos and compute gradients on X and Y axes using the Sobel operator. Then according to each step we have processed all dataset fingerpints and attached the example of one image in the Appendix section. Overall, it takes approximately 15 seconds to process and crop each photo so Python is not the best programming language to work with image processing, especially when it comes to computational complexity. There is a pseudocode for image cropping based on optimal core point:

Data: image
Result: cropped_image
1 norm_image = `normalization(`*image*`)`
2 grad_x = `gradient_x(`*norm_image*`)`
3 grad_y = `gradient_y(`*norm_image*`)`
4 `/* Window size estimated while processing images, optimal for fingerprint size of 200x200 */`
5 orientation_map, core_points = `orientation_estimation_field(`*norm_image, grad_x, grad_y, window_size=20*`)`
6 core_point = `optimal_core_point(`*core_points*`)`
7 cropped_image = `crop_image(`*norm_image, core_point*`)`

Algorithm 1: Pseudocode for Fingerprint Image Processing

4.3 PCA Computation

To compute Principal Component Analysis, we implemented a power iteration algorithm to find the eigenvectors for the PCA. Below is the pseudocode for the computation:

Data: selected_eigenvectors = empty array
`power_iteration(`*covariance_matrix, num_components, iterations=50*`)` n = size of covariance_matrix;
eigenvectors = empty list;
for *i = 0* **to** *num_components - 1* **do**
  v = random vector of size n;
  for *k = 0* **to** *iterations - 1* **do**
    v = normalize(dot product of covariance_matrix and v);
  end
  append v to eigenvectors;
  lambda = dot product of v and dot product of covariance_matrix and v;
  covariance_matrix = covariance_matrix - lambda * outer product of v;
end
return *transpose of eigenvectors*
`perform_pca(`*difference_matrix, num_components*`)` covariance_matrix = compute covariance of difference_matrix;
eigenvectors = power_iteration(covariance_matrix, num_components);
selected_eigenvectors = eigenvectors;
return *eigenvectors*
`project_data(`*data_matrix, eigenvectors*`)` **return** *dot product of transpose of data_matrix and eigenvectors*

Algorithm 2: Pseudocode for PCA analysis

The power iteration algorithm finds the eigenvectors of the covariance matrix, allowing for the computation of PCA with multiple iterations to ensure convergence. This provides a robust approach to identify key components and apply PCA to the fingerprint recognition process.

4.4 Parameters estimation

In this subsection, we describe the process of selecting appropriate parameters for our PCA algorithm, focusing on 'num_components' and 'iterations'. These parameters play a crucial role in determining the efficiency and accuracy of the PCA-based fingerprint recognition system.

The 'num_components' parameter represents the number of principal components retained after performing PCA. It affects both the dimensionality reduction and the information retained from the original dataset.

To select the optimal number of components, we consider the following factors:

- Variance Explained: The goal is to retain a significant portion of the variance in the data while reducing dimensionality. Typically, we aim to capture at least 95% of the variance.
- Computational Cost: Retaining more components increases computational requirements, as additional computations are required for each component.
- Dimensionality Reduction: A smaller number of components leads to greater dimensionality reduction, which can improve performance for large datasets.

The 'iterations' parameter in the power iteration process determines how many times the eigenvectors are iteratively updated to converge to the desired accuracy.

Typically, the choice of 'iterations' depends on the specific application and the size of the dataset. Empirical testing can help determine an appropriate number of iterations that balances accuracy with computational cost. For our case, we set the default to 200 iterations, which offers a good compromise between accuracy and efficiency.

As for the number of components, for image size of 150×150 we chose 100. In this case we will capture the most variance, while remaining the efficiency of the algorithm. Meanwhile, we computed treshold emperically, it equals 2000 for 150×150 image size (as average distance between fingers lays within 1000-2000 points), so that if minimal distance is higher then treshold, it will get rejected anyways.

4.5 Testing

For testing our implementation, we decided to iteratively select 1 picture ouf of each subfolder (excluding it from the initial folder). Run the PCA and print the distance between selected finger and others. If the minimal distance lays within the subfolder it was taken from, we can conclude it as verified.

- Accuracy test: As for 150×150 pictures, authors ended up with 55-60% accuracy for different numbers of components and iterations. On the other hand, original 200×200 images accuracy test resulted in $\approx 70\%$
- Efficiency test: Total pixels: The total number of pixels in each image is given by $w \times h$, where w is the width and h is the height of the single image; Change in size: Consider the ratio of pixel counts to understand how much larger one set of images is compared to another.
  - Total pixels for entire dataset (250 images):

$$22,500 \times 250 = 5,625,000 \text{ pixels.}$$

$$40,000 \times 250 = 10,000,000 \text{ pixels.}$$

– Efficiency Gain: Given that computational load is proportional to the number of pixels, you can calculate the efficiency gain as a ratio of the total pixel counts:

$$\text{Efficiency Gain} = \frac{5,625,000}{10,000,000} \approx 1.777$$

Overall, this 55-60% to 70% accuracy decrease means a lot, but as for efficiency, we achieved almost 2 times more speed just by reducing the image size using core point detection.

5 Conclusions

In conclusion our implementation process involves steps such as data pre-processing, PCA computation, parameter selection, and testing. We were not aimed to make the best, the most accurate and efficient algorithm but we tend to implement it with PCA, analyze the results, evaluate the performance. We selected Python for its availability of packages despite its computational inefficiency. Utilizing the PCA algorithm we were eager to find for a balance between accuracy and efficiency by selecting appropriate parameters. Our testing results indicate a trade-off: while accuracy decreased from 70-75% to 55-60% for different image sizes, efficiency increased nearly twofold due to reduced computational load and images processing. Future work involves optimizing parameters for algorithms, further testing of other algorithms, seeking new and balance striking between efficiency and accuracy algorithms because we have determined, it is very time-consuming to calculate large matrices, more than 20000×20000, finding its eigenvectors and eigenvalues. Ultimately, our project seeks to contribute to advancements in security and biometric authentication systems

A Useful auxiliary facts

Author's code on GitHub - link

References

[1] An algorithm for fingerprint core point detection by Atipat Julasayvake, Somsak Choomchuay - link

[2] A Fingerprint Recognition Algorithm Based on Principal Component Analysis by Wang Yongxu, Ao Xinyu, Du Yuanfeng, Li Yongping - link

[3] A Fingerprint-based Access Control using Principal Component Analysis and Edge Detection by Ernande F. Melo, H. M. de Oliveira - link

[4] FingerPrint Recognition Using Principle Component Analysis(PCA) submitted to Prof. Neeraj Bhargava, presented by Arpit Kumar Sharma - link

[5] A Novel Method for Fingerprint Core Point Detection by Navrit Kaur Johal, Prof. Amit Kamra - link

[6] Dataset of fingerprints - link

A Appendix

Figure 1: Original image

Figure 2: Normalized image

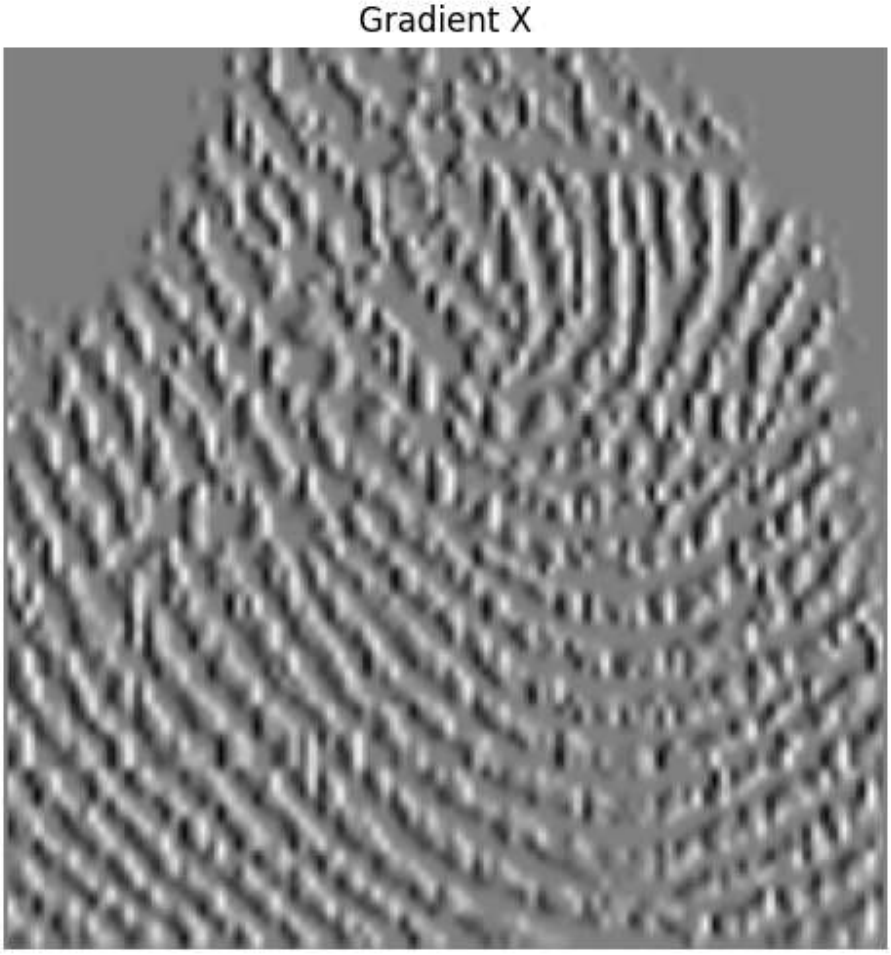


Figure 3: Gradient on X-axis

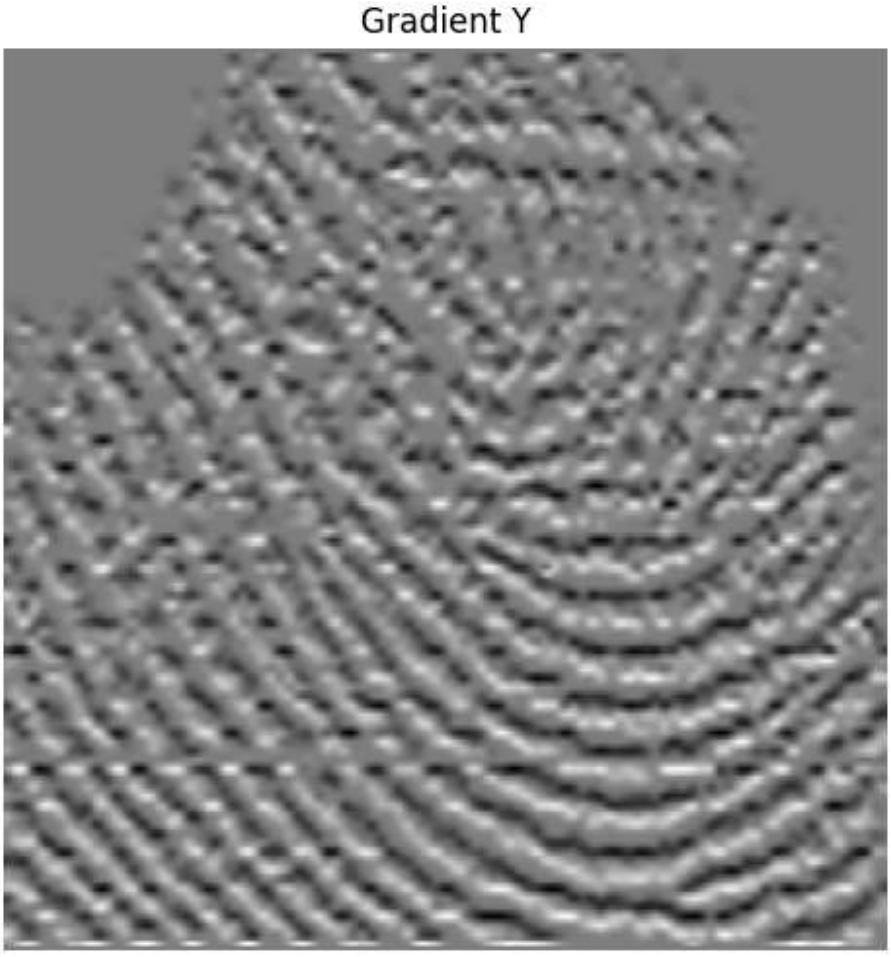


Figure 4: Gradient on Y-axis

Figure 5: Detected potential core points displayed on original image

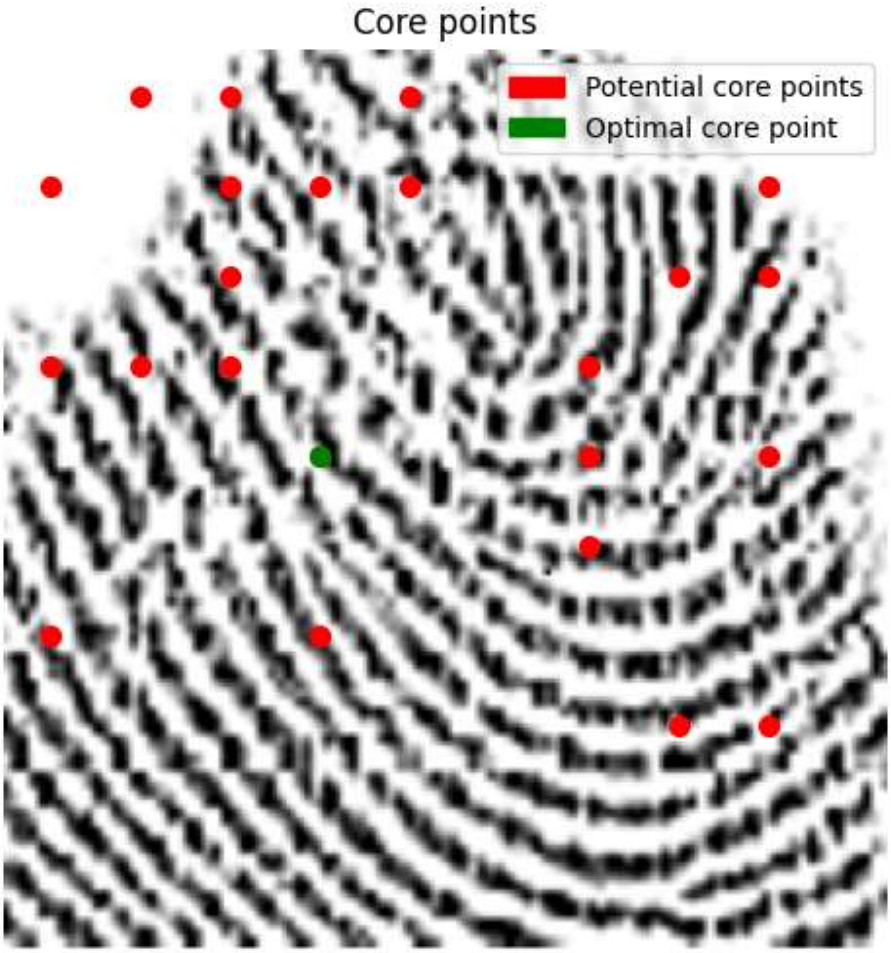


Figure 6: Detected optimal and potential core points displayed on original image

Figure 7: Optimal core point displayed on original image

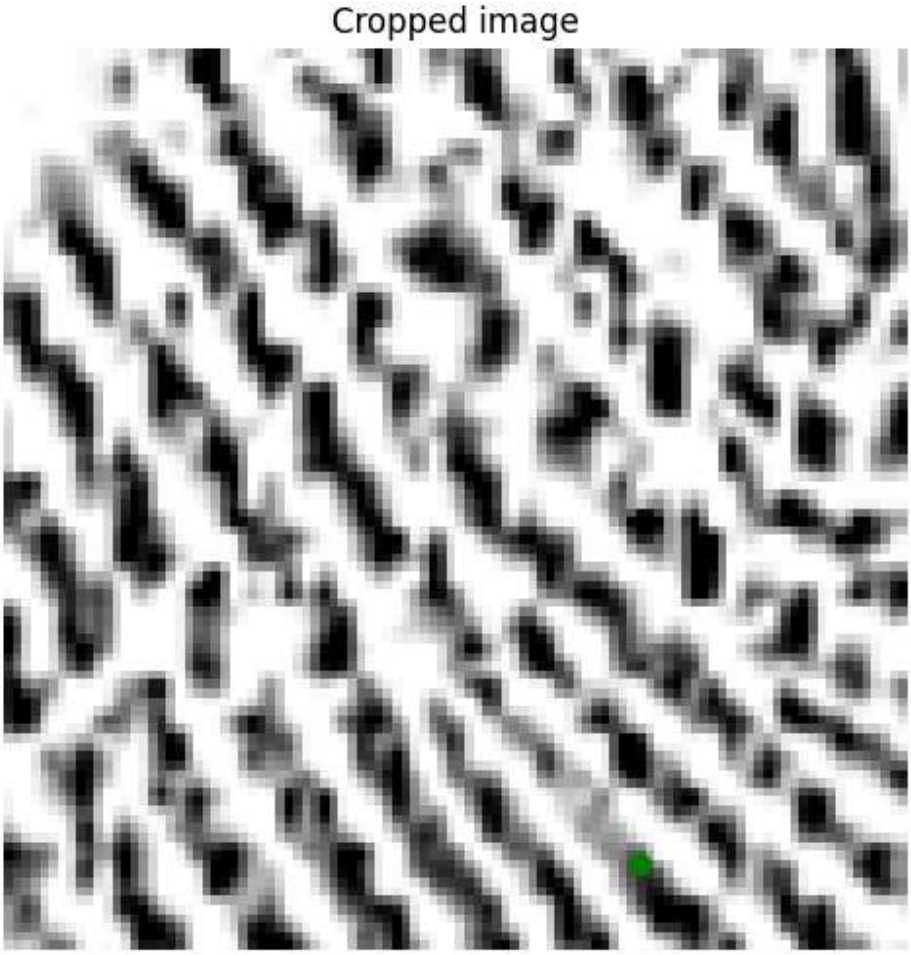


Figure 8: Cropped image based on optimal core point to size 100x100

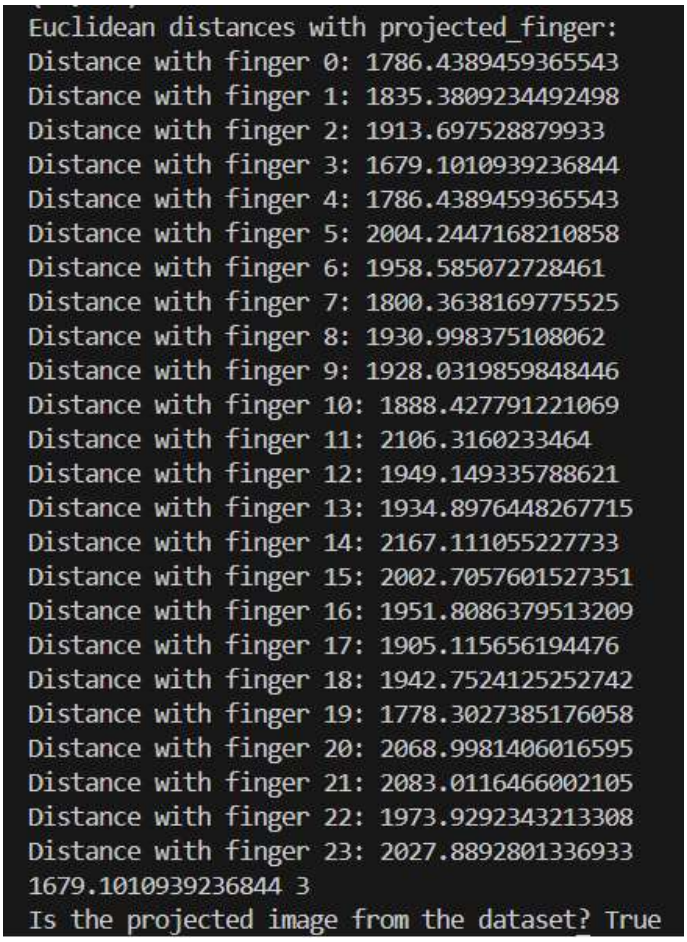


Figure 9: Example of result of Principal Component Analisys along with distances