Fingerprint Descriptor Model Utilizing Euclidean Minutiae Features

Yurii Pohuliaiev[1], Kirill Smelyakov[1]

[1]Kharkiv National University of Radio Electronics, 14 Nauky Ave, Kharkiv, 61166, Ukraine

Abstract

This paper introduces a new way to describe fingerprint details using basic geometric properties to improve how well we can identify people biometrically. The method concentrates on where fingerprint lines end or split, fixing issues with older methods like SIFT and HOG, and avoiding the heavy load of deep learning. The descriptor uses distances and angles around the center of fingerprint details, making it resistant to changes in position, rotation, and size. The goals include checking how well the details stay the same, creating a model based on geometry, and testing it on regular data. By mixing math proofs with real-world adjustments, the model can spot shifts 92.85% of the time, which is helpful for aligning fingerprints, but needs extra work for negative matches. This simple, non-machine learning method can be easily expanded for different biometric uses.

Keywords

fingerprint, minutiae, descriptor, biometrics, Euclidean features, affine transformations

1. Introduction

1.1. Motivation

Fingerprint features, like ridge endings and bifurcations, are identity markers because they are unique and stable [1]. Current systems have such disadvantage as dependency from image quality and alignment, especially touch-free systems popular after COVID [2]. Older methods convert fingerprint details into vector forms, losing detail arrangement. Machine learning can reach high accuracy (e.g., 97% on datasets) but demands computation and can be biased [3, 4].

This paper introduces a fingerprint descriptor using Euclidean geometry to locate alignment shifts quickly. By using distances and angles between point pairs, focused on the center of the minutiae, the method remains constant despite shifts, rotations, and scaling, avoiding reliance on machine learning[5]. This solution supports alignment in places with limited resources, like phones, offering a flexible option to neural networks [2]. It finds shifts, a step to prepare fingerprints for matching, rather than full identification. It facilitates matching fingerprints from different scanners and conditions.

1.2. State of the Art

Fingerprint recognition involves looking at minutiae, using local descriptors, and using deep learning. Local descriptor methods, like Orientation Local Binary Patterns (OLBP), extract ridge and valley details successfully, but often miss topological minutiae details, resulting in extra features in unclear images [6].

Deep learning has accuracies of 97% on datasets like FVC2000 using convolutional neural networks (CNNs) [3, 7]. These methods demand training data and computation, challenging real-time use, and bring up bias questions [4, 8].

Contactless fingerprint recognition is more useful because of hygiene concerns [2, 9] but struggles with image quality and spatial orientation in uncontrolled conditions. Incomplete fingerprints require alignment methods to handle distortions [9]. The descriptor described uses invariant Euclidean metrics

ProfIT AI'25: 5th International Workshop of IT-professionals on Artificial Intelligence, October 15–17, 2025, Liverpool, UK
yurii.pohuliaiev@nure.ua (Y. Pohuliaiev); kyrylo.smelyakov@nure.ua (K. Smelyakov)
https://nure.ua/staff/kirilo-sergijovich-smeljakov (K. Smelyakov)
0009-0005-5883-1661 (Y. Pohuliaiev); 0000-0001-9938-5489 (K. Smelyakov)

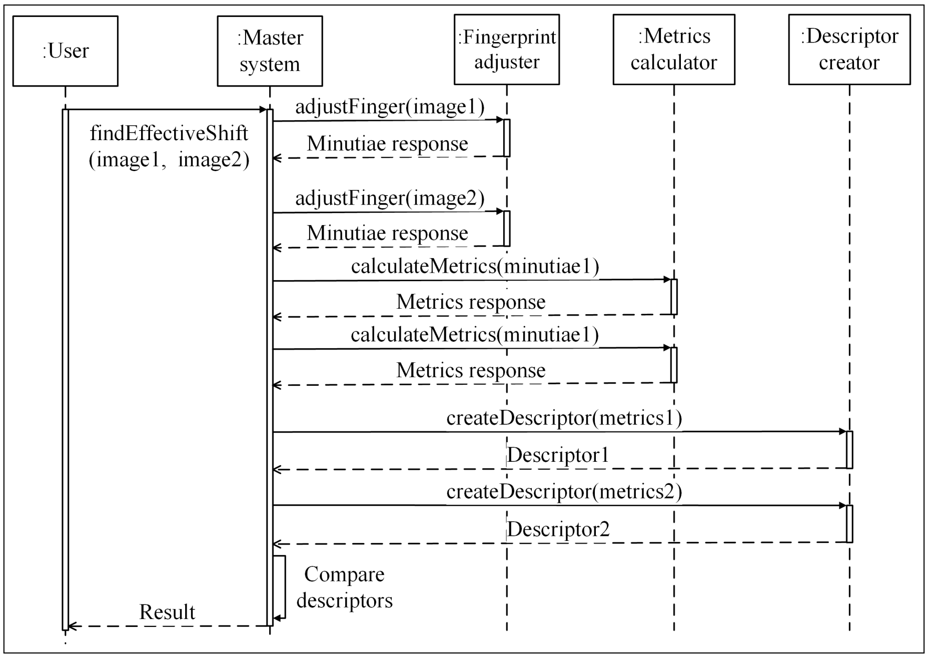

Figure 1: Sequence diagram of proposed flow

to address problems, aligning images based on shifts without neural networks. By keeping the topological features of minutiae unlike deep learning, this method helps with alignment in biometric conditions.

1.3. Objectives and Tasks

The goal is to create a fingerprint descriptor based on Euclidean geometry for shift-based matching, without machine learning, tackling alignment issues while remaining efficient, which involves these steps:

1. Study minutiae traits (location, direction, type) for stability in images that are changed or distorted.
2. Develop a descriptor model that uses relative coordinates focused on the center of mass of the minutiae, along with tables of Euclidean measurements (distances and angles).
3. Create a method for picking out a key group of minutiae near the center of mass, to reduce edge problems and boost dependability.
4. Test the model using the FVC2000 (DB1_B) dataset [10], adjusting settings (like square side length and distance/angle limits) to measure how well shifts are detected.

2. Fingerprint Euclidean Descriptor Model

2.1. Summary

The descriptor is a tensor of Euclidean minutiae attributes, processed via coordinate alignment, metric calculation, feature extraction, and matching, per Figure 1. Minutiae are extracted using open-source algorithms [11] for compatibility with fingerprint processing pipelines. This favors computation, making it suited to mobile biometric systems.

2.2. Mathematical Model

A fingerprint image is represented as a pixel matrix in 24-bit RGB format:

$$P = \mathbb{Z}^3 \cap [0, 255]^3, \quad I \in P^{H \times W} \tag{1}$$

where P is the set of pixels, I is the image, and H and W are height and width.

Minutiae extraction can be represented as a four-stage process:

- **Binarization**: converting the image to a binary mask, $I_b \in \{0,1\}^{H\times W}$, using a functional transformation $M_b = f(I)$.
- **Skeletonization**: constructing a discrete skeleton, $S_b = f(M_b)$, to highlight ridge structures.
- **Detection**: identifying terminations ($M_1 = \{p \in S_b | N(p) = 1\}$) and bifurcations ($M_2 = \{p \in S_b | N(p) \geq 3\}$).
- **Filtering**: removing artifacts, $M_1' = \{p \in M_1 | d(p) \leq \sigma_p\}, M_2' = \{p \in M_2 | d(p) \leq \sigma_p\}$, where $d(p)$ is the distance to the nearest ridge and σ_p is the error threshold.

As a result we get a minutiae set:

$$M = \{(x, y, \alpha, \chi) | x \in [0, W], y \in [0, H], \alpha \in [0, 2\pi), \chi \in \{0, 1\}\} \tag{2}$$

where x, y are coordinates, α is the orientation angle, and χ denotes minutiae type (0 for terminations, 1 for bifurcations).

To ensure invariance to affine transformations, minutiae coordinates should be transformed relatively to the center of mass:

$$M' = T_{rel}(M) = \{(x - \bar{x}, y - \bar{y}, \alpha, \chi) \in M\}, \quad \bar{x} = \frac{1}{|M|}\sum x_i, \quad \bar{y} = \frac{1}{|M|}\sum y_i \tag{3}$$

The center of mass is stable under affine transformations, with mathematical expectations $E[\bar{x}] = \mu_x$, $E[\bar{y}] = \mu_y$, and variances $D[\bar{x}] = \frac{\sigma_x^2}{n}$. This stability is derived from the consistent distribution of minutiae across papillary patterns, which remain unique to each finger [1]. Thus a subset of minutiae is selected within a square region centered on the center of mass to reduce edge artifacts:

$$M' = \Phi_B(M') = \{m \in M' : |x| \leq \frac{B}{2} \wedge |y| \leq \frac{B}{2}\} \tag{4}$$

where B is the square side length, optimized empirically, $|x|$ and $|y|$ represents absolute difference to square center respectively.

The descriptor comprises two matrices:

$$d_{ij} = \sqrt{\Delta x^2 + \Delta y^2}, \quad a_{ij} = \mathrm{atan2}(\Delta y, \Delta x) \tag{5}$$

$$D = [d_{ij}]_{1..|M'|,1..|M'|}, \quad A = [a_{ij}]_{1..|M'|,1..|M'|} \tag{6}$$

where D contains pairwise Euclidean distances and A contains angles relative to the x-axis.

Matching is performed using thresholds σ_1 (distance) and σ_2 (angle):

$$\Delta D = |D_1(m_1, m_1') - D_2(m_2, m_2')|, \quad d' = \begin{cases} \Delta D/\sigma_1 & \Delta D \leq \sigma_1 \\ \infty & \Delta D > \sigma_1 \end{cases} \tag{7}$$

$$S_m = d' + a', \quad S = [S_m]_{m_1 \in M_1', m_2 \in M_2'} \tag{8}$$

The matching score is:

$$R(m_1, m_2) = \frac{n}{\min(|M_1'|, |M_2'|)} \tag{9}$$

where n is the order of the maximum minor of the score matrix S. This score quantifies the proportion of matched minutiae, enabling shift detection.

In a practical scenario with two fingerprints made from the same finger, the descriptor should be able to identify a translation offset by matching minutiae pairs with high R values validated by their orientation and type which ensures robustness against scanner variations.

Figure 2: First original fingerprint

Figure 3: Second original fingerprint

2.3. Software Implementation

The descriptor is implemented in C#, Python, and OpenCV, processing the FVC2000 (DB1_B) dataset. The workflow, visualized in Figures 2 to 7, includes image preprocessing, minutiae extraction, descriptor computation, and matching. The system is optimized for efficiency, suitable for real-time applications on mobile devices.

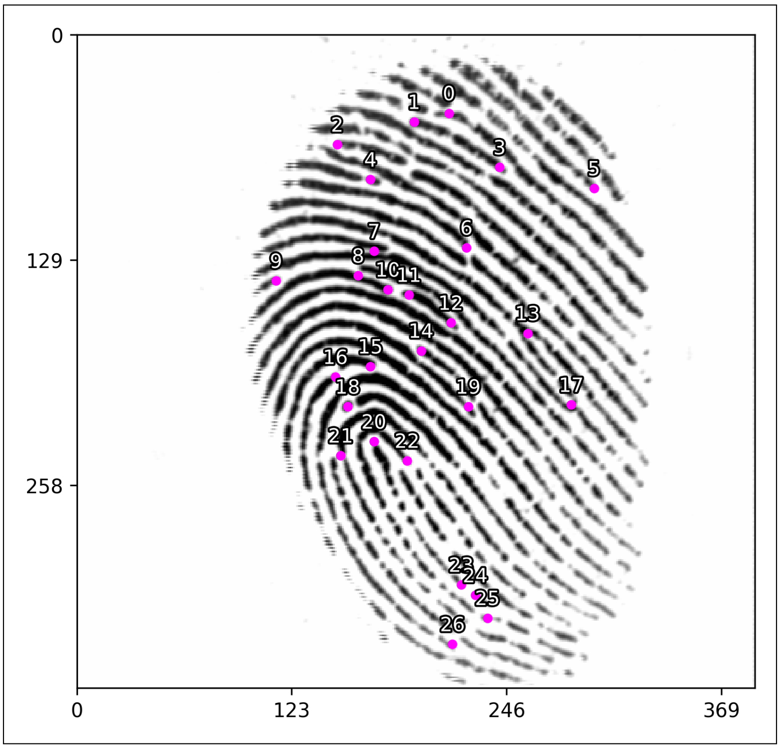

Figure 4: Detected minutiae on first fingerprint

3. Experiments Methodology

3.1. Hardware

Experiments used an Intel i5-12400F, 16GB RAM, GTX 1650, and Windows 11 for processing and replication.

3.2. Data

The FVC2000 DB1_B dataset (100 fingers, 8 impressions per finger) [10] went through preprocessing to improve contrast and reduce noise, addressing common image issues [2].

3.3. Plan and Metric

The experiments tuned four parameters: square side length (B), distance threshold (σ_1), angle threshold (σ_2), and matching score threshold ($R_{threshold}$). Shift detection success is:

$$Q(I_1, I_2) = 1 \text{ if } \exists [i, j] : R_{result}[i, j] = 1 \text{ else } 0 \tag{10}$$

where R_{result} combines experimental and verified matches. A score balances performance:

$$Q' = 0.7 \cdot Q + 0.3 \cdot \Delta Q \tag{11}$$

This emphasizes detection rates and efficiency, tested for statistical validity.

4. Results and Discussion

4.1. Parameter Tuning

- **Experiment 1 (B)**: Testing square side length B from 20 to 200 pixels, $B = 160$ was best, with a 90.00% detection rate and 66.43% score (Table 1). Larger values improved accuracy but cut efficiency.

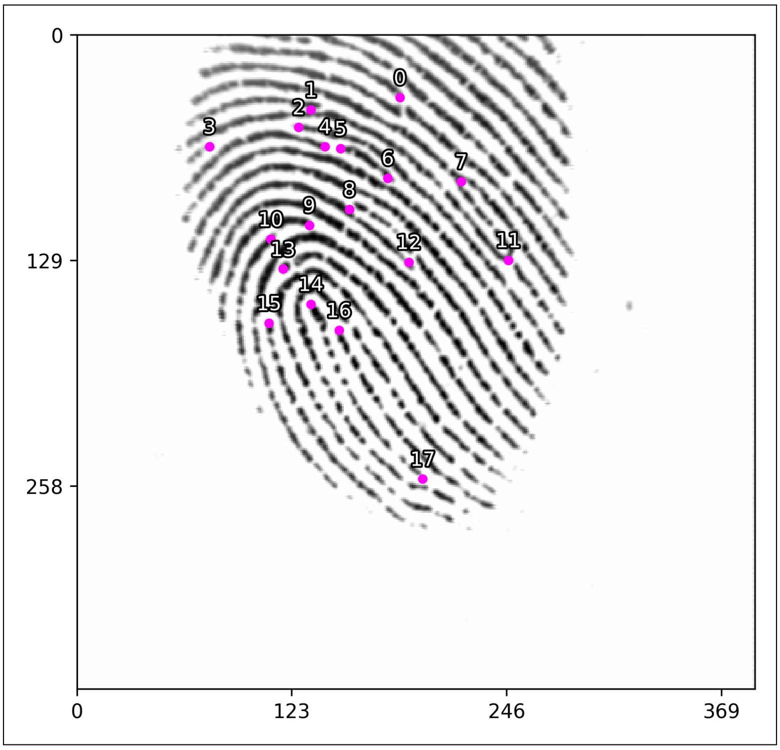

Figure 5: Detected minutiae on second fingerprint

Table 1
Effective square side B

No.	Value	Result	Increment	Score
1	20	0.00%	0.00%	0.00%
8	160	90.00%	11.43%	66.43%
10	200	92.85%	0.71%	65.21%

- **Experiment 2 (σ_1)**: A distance threshold of $\sigma_1 = 9$ gave a 65.42% score, balancing accuracy and cost.
- **Experiment 3 (σ_2)**: An angle threshold of $\sigma_2 = 48°$ scored 65.64
- **Experiment 4 ($R_{threshold}$)**: A matching score threshold of $R = 30\%$ had a 67.35% score.

The system was also tested for noise and partial fingerprints, using forensic scenarios [9]. Early data shows accuracy with noise, but more work is needed for distortions, setting up improvements.

4.2. Discussion

The model had a 92.85% shift detection rate, like deep learning [3], but used less computation. It managed only 1.6% of the data of a brute-force approach (Table 2), suiting low-resource uses. Despite brute-force approach is more time-efficient, it proposes much more combinations, not really relevant as a shift candidates. Model also needs post-processing to remove false negatives, like other methods [2]. The descriptor resists affine changes, aiding alignment across scanners, but struggles with rotations. The lightweight design is good for real-time uses like mobile authentication or fingerprint analysis [9]. Later studies could mix metrics with learning for filtering or extend to palm prints [2]. These ideas will be implemented in further researches, including datasets and analysis.

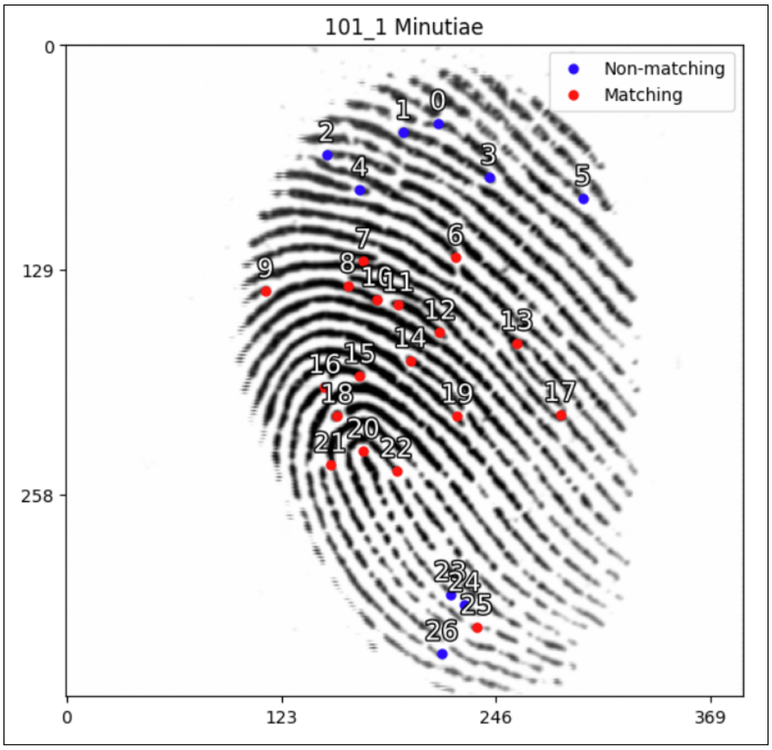

Figure 6: Comparison results for first fingerprint

Table 2
Model vs. brute-force

Metric	Proposed	Brute-force
Pairs	61436	3839640
Time (ms)	1682	282

5. Conclusions

The descriptor uses Euclidean characteristics for a 92.85% shift detection rate, resisting affine changes. It aligns fingerprints well, working as a light choice, but requires post-processing, restricting use for identification. Next steps:

- Improve rotation invariance by gauging fingerprint angles.
- Add hybrid ML for filtering.
- Add other modalities, like vein patterns.

These changes will be studied, building on this data to address more biometric issues.

Contributions of Authors

Problem formulation – Kirill Smelyakov; methods, theory, implementation, analysis, article – Yurii Pohuliaiev.

Conflict of Interest

No conflict of interest.

Figure 7: Comparison results for second fingerprint

Financing

Self-funded.

Data Availability

Data and source code available upon request [11].

Use of Artificial Intelligence

No AI methods used.

Acknowledgments

All authors approved the manuscript.

Declaration on Generative AI

The author(s) have not employed any Generative AI tools.

References

[1] S. Hemalatha, Relevance of contemporary fingerprint-based biometric methods in authentication, in: International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE), 2020. doi:10.1109/ic-ETITE47903.2020.342.

[2] K. Sreehari, M. Anzar, Recent advancements and inventions in contactless fingerprint recognition, Computers and Electrical Engineering 122 (2025). doi:10.1016/j.compeleceng.2024.109894.

[3] S. Garg, et al., CNN with inversion and several augmentation approaches on the FVC2000_DB4 dataset, Systems and Soft Computing 6 (2024). doi:10.1016/j.sasc.2024.200106.
[4] X. Guo, et al., Deep contrastive learning demonstrating commonalities among the fingerprints, Science Advances 10 (2024). doi:10.1126/sciadv.adi0329.
[5] M. R. Chen, H., M. Zhang, Recent progress in visualization and analysis of finger-print level 3 features, ChemistryOpen 11 (2022). doi:10.1002/open.202200091.
[6] A. Kumar, Orientation Local Binary Patterns (OLBP) attaining competitive outcomes on the FVC2002, FVC2004, and FVC2006 datasets, SN Computer Science 1 (2020) 67. doi:10.1007/s42979-020-0068-y.
[7] M.-S. Pan, et al., Optimization fingerprint reconstruction using deep learning algorithm, 2022 17th International Microsystems, Packaging, As-sembly and Circuits Technology Conference (IMPACT) (2002) 1–3. doi:10.1109/impact56280.2022.9966693.
[8] Y. Guo, Creation of counterfeit fingerprints with DCGAN and verification through a Siamese Network, in: International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering (ICBAIE), 2023. doi:10.1109/ICBAIE59714.2023.10281210.
[9] J. Martins, et al., Use of incomplete fingerprints in forensic applications, Sensors 24 (2024). doi:10.3390/s24020664.
[10] D. Maio, et al., Fvc2000: Fingerprint verification competition 24 (2002) 402–412. doi:10.1109/34.990140.
[11] Y. Pohuliaiev, Euclidean descriptor software implementation, https://drive.google.com/file/d/1bRD2yqc7Zg6NSBpGYvtAVv-hiJrekJJh/view?usp=sharing, 2025. Google Drive repository.