Biometric System Laboratory
University of Bologna - ITALY

Davide Maltoni
davide.maltoni@unibo.it

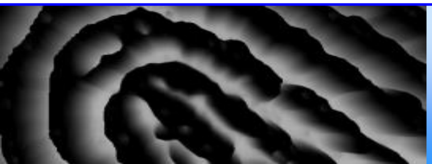
Hands on Fingerprint Recognition with OpenCV and Python

Why an Hands on Lecture?

- In past editions of biometric schools we gave a more classical lecture on fingerprint recognition
  - you can download a PDF from: http://bias.csr.unibo.it/maltoni/FingLecture2019.pdf

- The new “hands on” format is aimed at:
  - Demonstrating (in practice) the basic building-blocks of fingerprint recognition
  - Showing that classical algorithms remain an important background in biometrics

NOTE: even if modern machine learning techniques can improve a number of fingerprint processing and recognition tasks (especially on latent fingerprints), fingerprint recognition was not drastically reshaped by deep learning as other biometric modalities, and classical minutiae-based approaches are still the state-of-the-art.

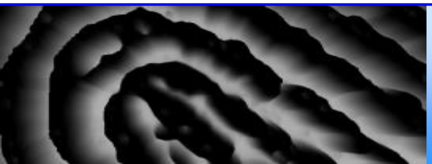


Links to the code

- These slides accompany a practical example on a Jupyter notebook (credits Raffaele Cappelli), which can be run locally or on Google Colab.

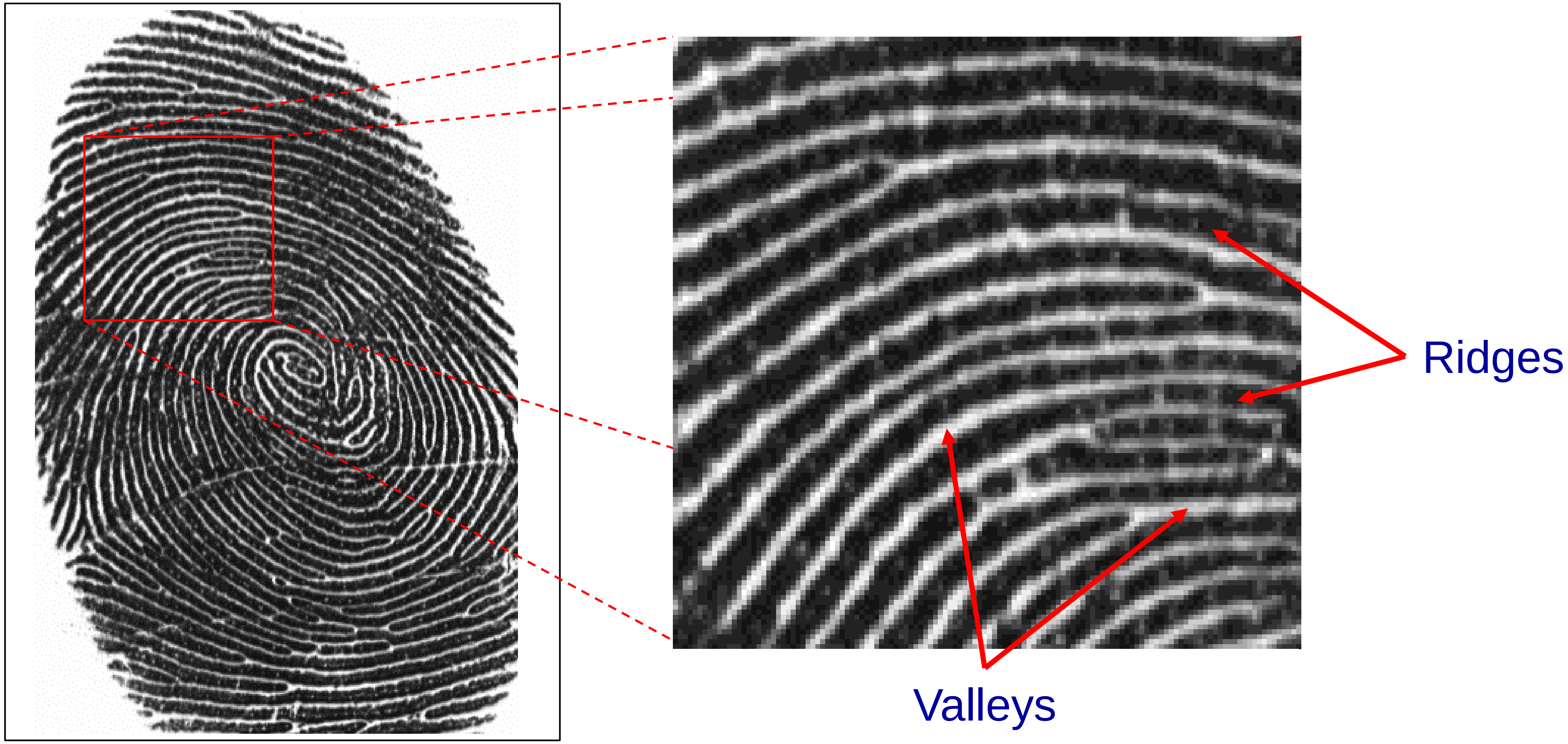
- How to run the notebook locally (recommended):
  - Download https://tinyurl.com/hands-on-fr
  - Required: a Jupyter installation with OpenCV, ipywidgets, matplotlib
    - In Anaconda: "conda install -c conda-forge -y opencv notebook ipywidgets matplotlib"

- How to run the notebook on Colab:
  - Open https://colab.research.google.com/drive/1u5X8Vg9nXWPEDFFtUwbkdbQxBh4hba_M



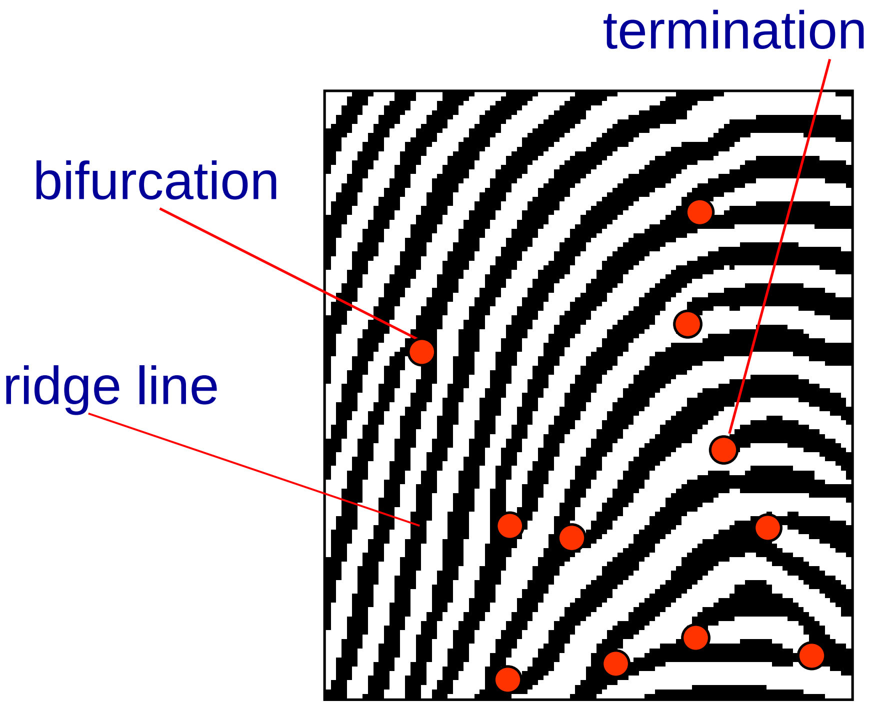
Fingerprint anatomy

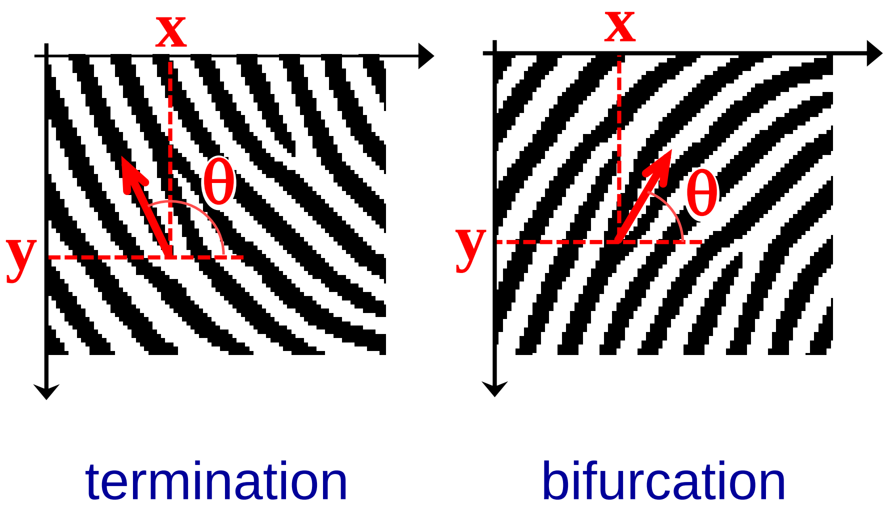
A fingerprint is composed of a set of lines (ridge lines), which mainly flow parallel, making a pattern (ridge pattern).



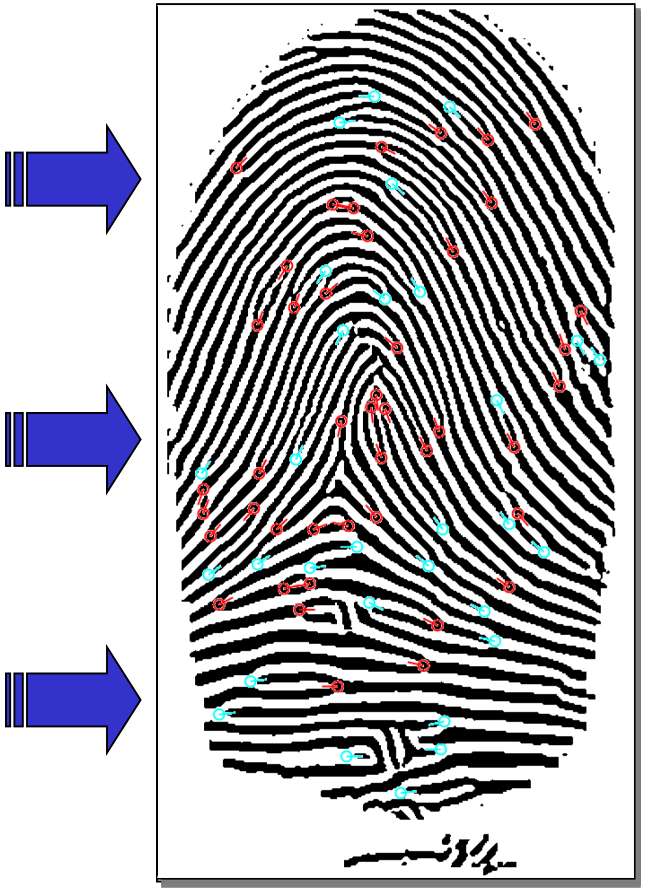
Minutiae

Minutiae are determined by the termination or the bifurcation of the ridge lines; they are usually represented by the coordinates (x, y), the angle θ between the minutia tangent and the horizontal axis, and the type (termination/bifurcation).





Feature extraction: main steps



Fingerprint image

↓

Orientation image | Fingerprint foreground | Frequency image

↓

Ridge pattern

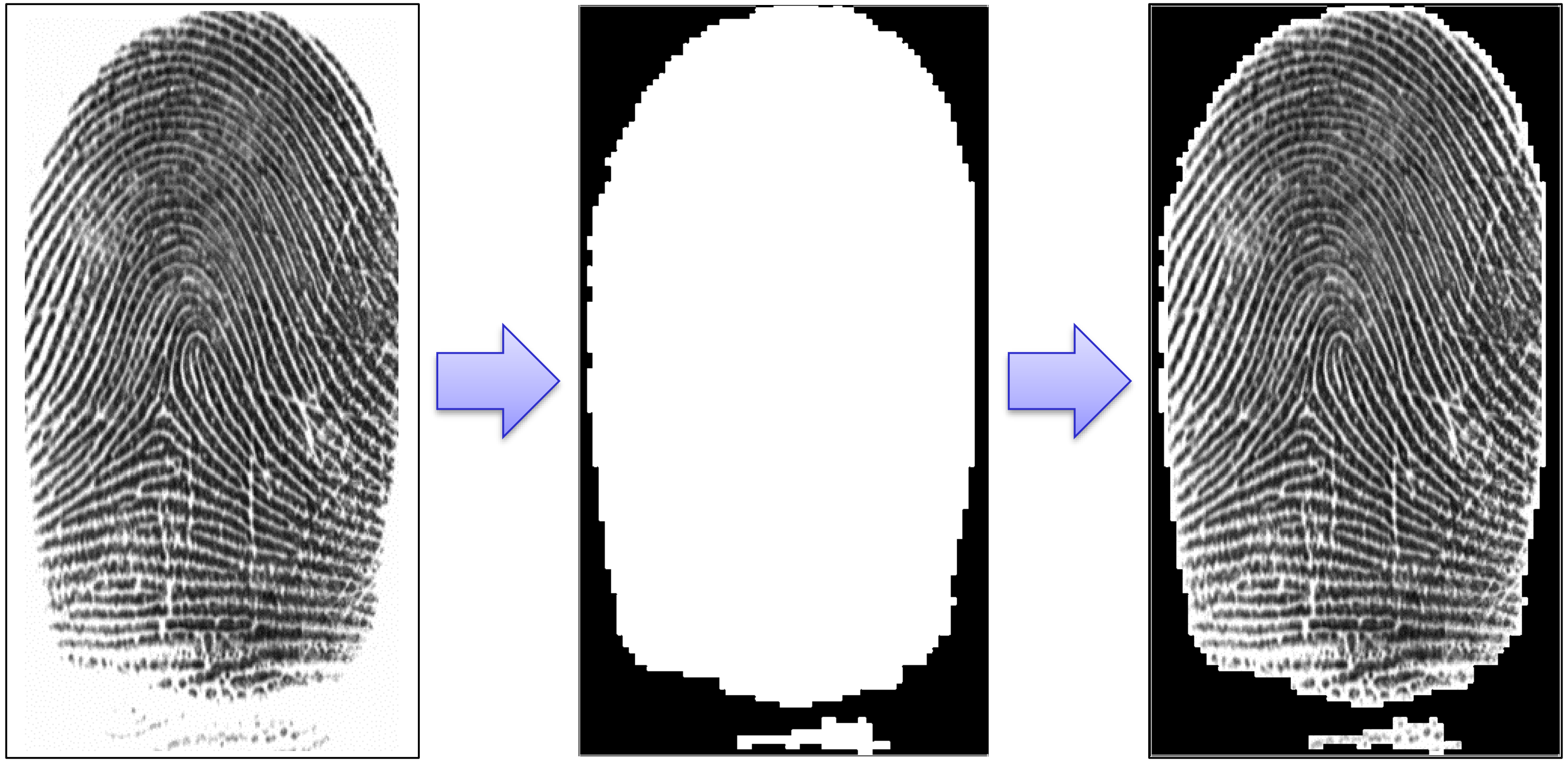
↓

Minutiae

Segmentation

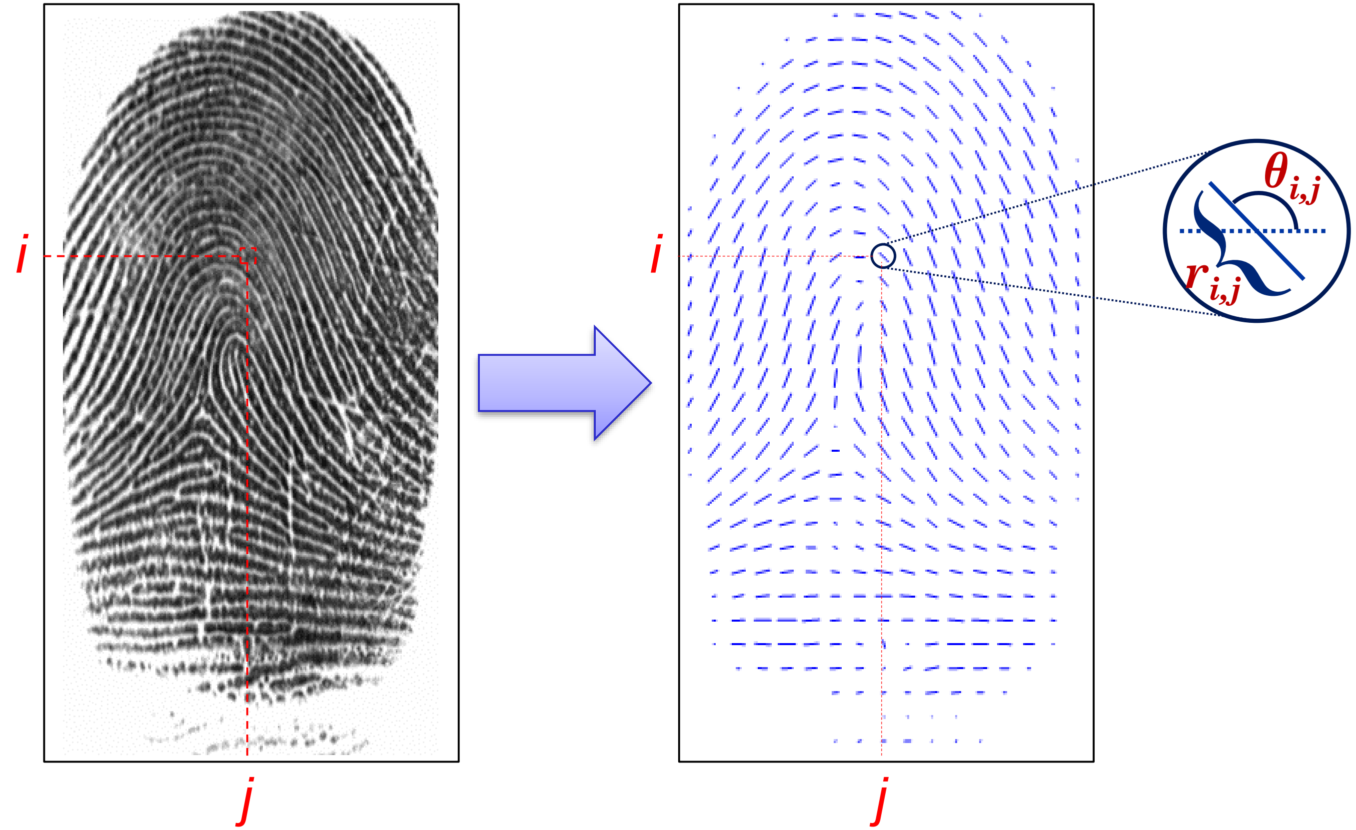
The segmentation stage is aimed at separating the fingerprint area (foreground) from the background.

The foreground is characterized by the presence of a striped and oriented pattern; background presents a uniform pattern.



Local ridge orientation

The local ridge orientation at $[i, j]$ is the angle $\theta_{ij} \in [0,180°[$ that the fingerprint ridges form with the horizontal axis in an arbitrary small neighborhood centered at $[i, j]$.

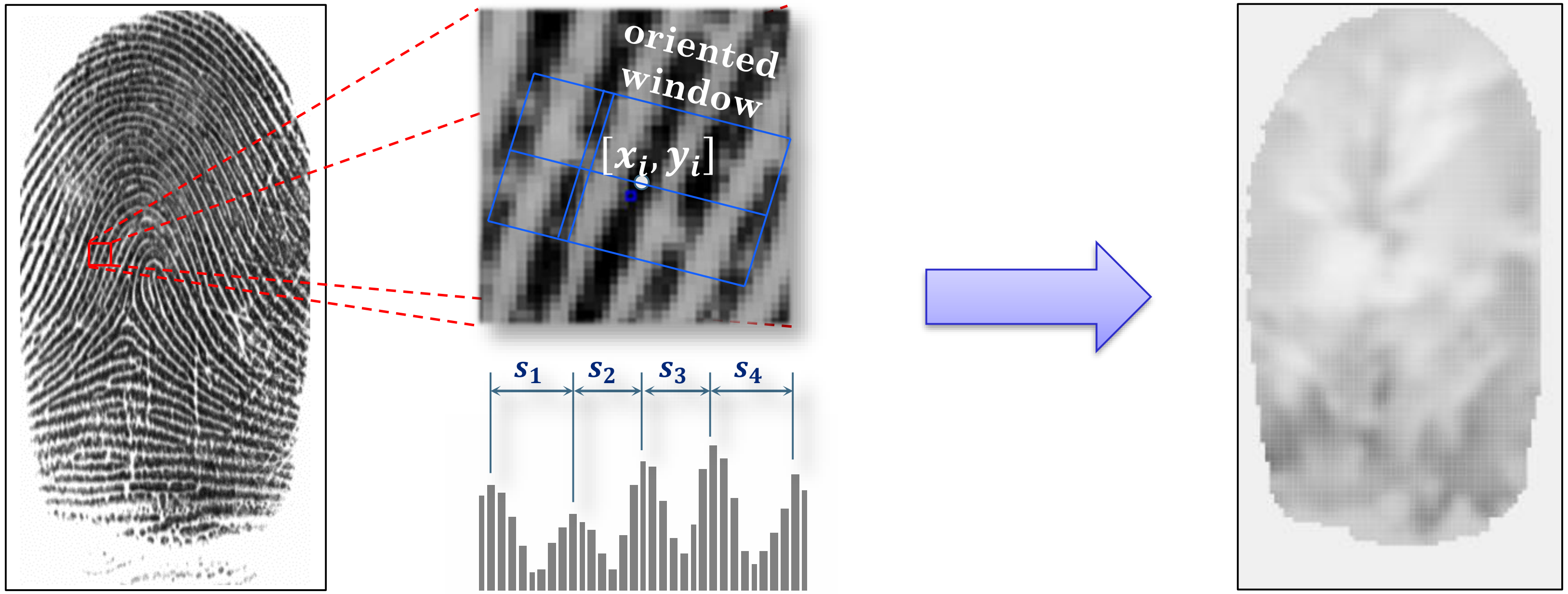


The simplest approach to extract local ridge orientations is based on computation of gradient phase angles.

Local ridge frequency

The local ridge frequency f_{xy} at $[x, y]$ is the number of ridges per unit length along a hypothetical segment centered at $[x, y]$ and orthogonal to the local ridge orientation θ_{xy}.

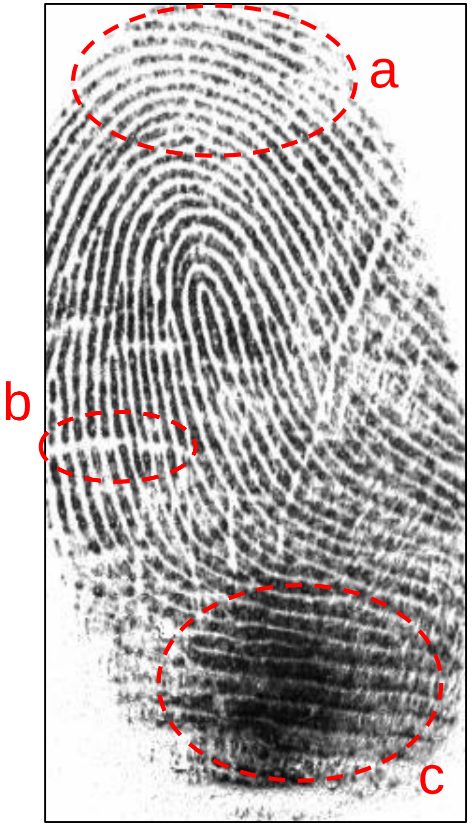
A possible approach is to count the average number of pixels between two consecutive peaks of gray-levels along the direction normal to the local ridge orientation.



Enhancement (1)

The performance of feature extraction and comparison algorithms are strictly related to the image quality.

The objective of enhancement techniques is to improve the fingerprint image quality.



Typical degradations:

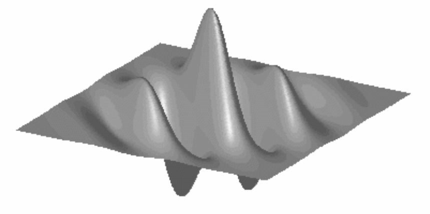
a. ridge lines are not continuous;
b. cuts, creases and bruises on the finger;
c. parallel ridges are not well separated.

The most widely used technique for fingerprint enhancement is based on contextual filters.

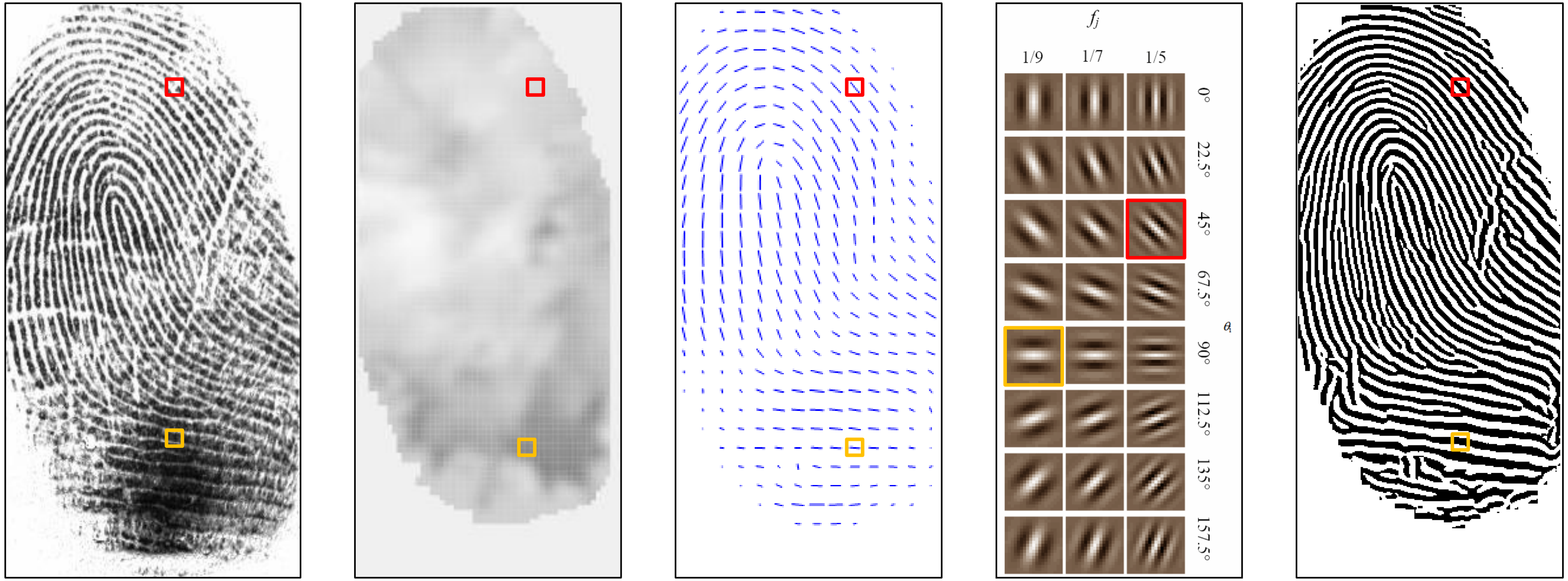
In contextual filtering, the characteristics of the filter used change according to the local context.

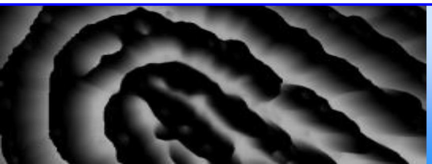
Enhancement (2)

The local context of a fingerprint is represented by the ridge orientation and frequency.



Gabor filter: sinusoidal plane wave tapered by a Gaussian.





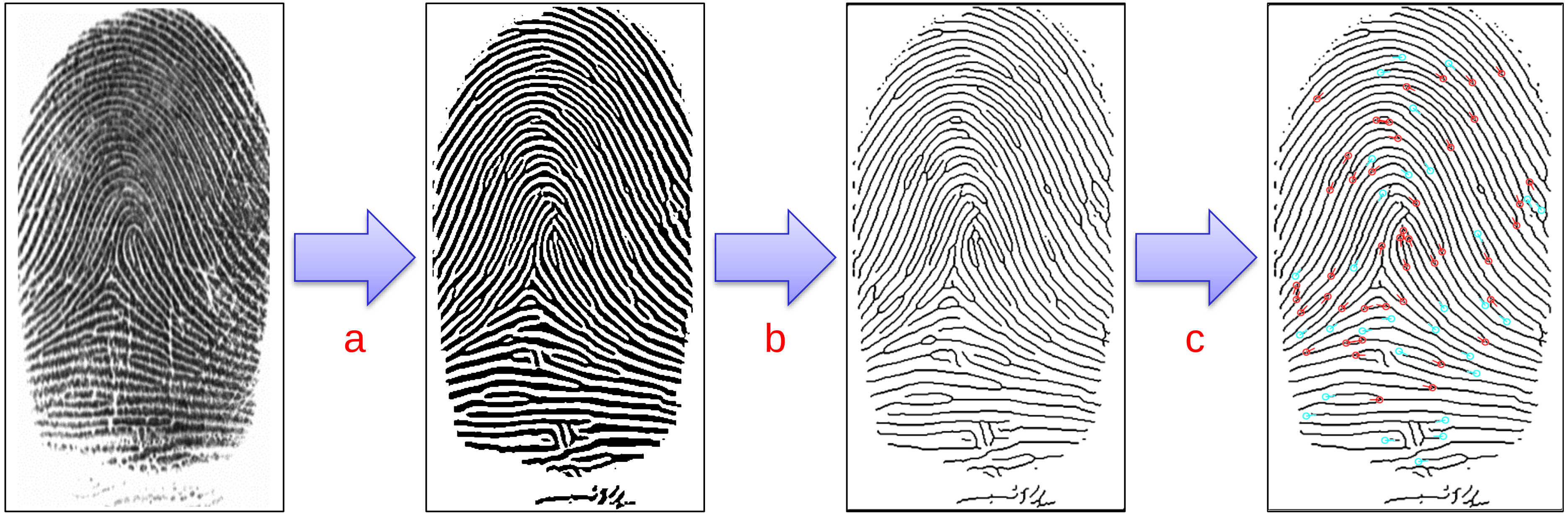
Minutiae detection (1)

Traditional approach:

a. Enhancement/Binarization: conversion into a binary image;

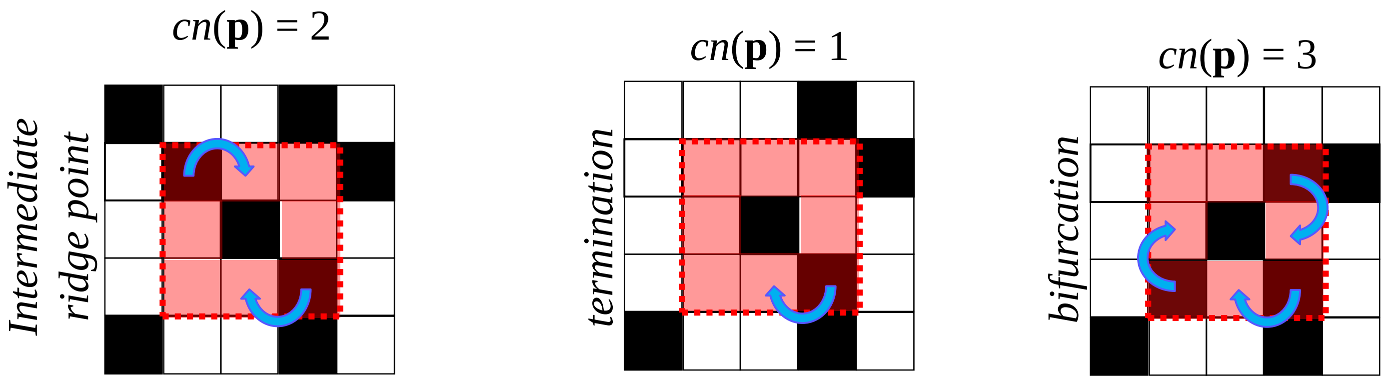
b. Thinning: the binary image is thinned to reduce the ridge thickness to one pixel;

c. Detection: an image scan then allows to detect minutiae.



Minutiae detection (2)

Minutiae detection is based on the computation of the crossing number (*cn*):



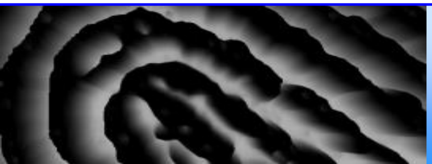
It is simple to note that a pixel **p** is:

– an intermediate ridge point if $cn(\mathbf{p})=2$;

– a termination if $cn(\mathbf{p})=1$;

– a bifurcation if $cn(\mathbf{p})=3$;

– part of a more complex minutia if $cn(\mathbf{p})> 3$.

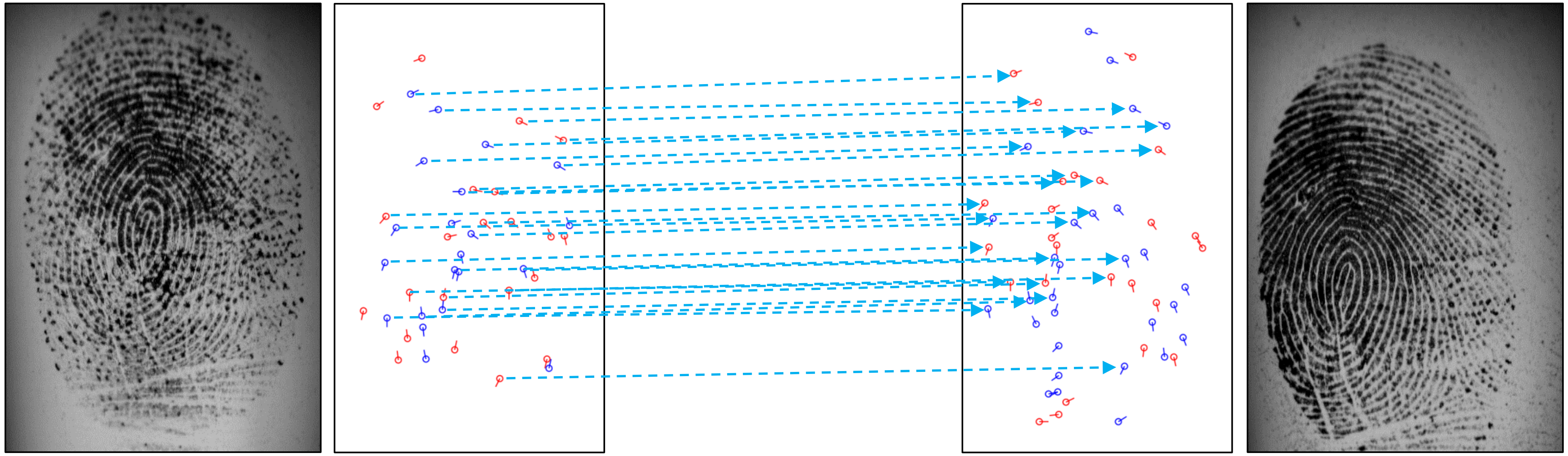


Minutiae-based fingerprint comparison

In minutiae-based comparison, the fingerprint is represented by a feature vector of variable length whose elements are the fingerprint minutiae.

A minutia is represented by the tuple $m = \{x, y, \theta, t\}$ containing the minutia coordinates, its orientation and type.

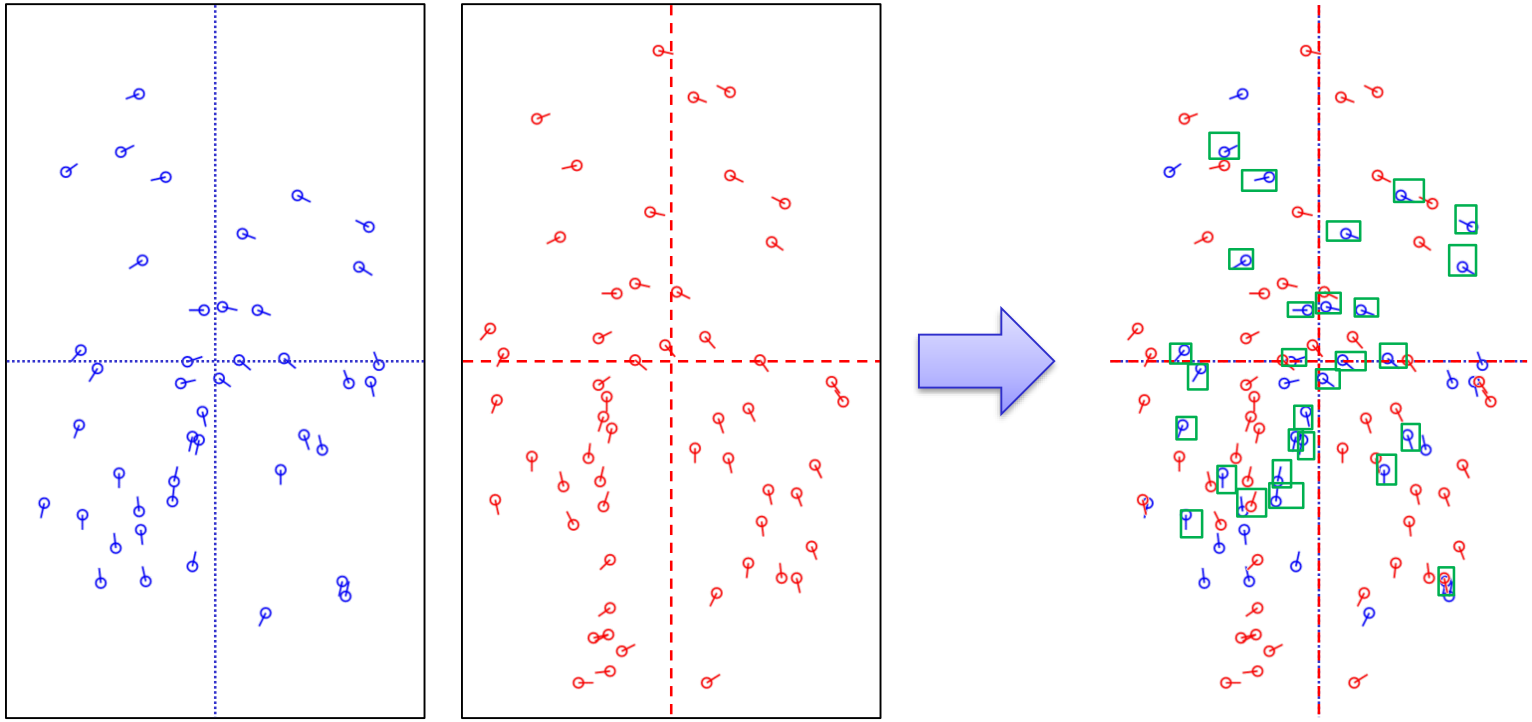
$T_1 = \{m_1, m_2, \dots, m_u\}$ $T_2 = \{m'_1, m'_2, \dots, m'_v\}$



$$score = \frac{\#pairs}{(u+v)/2}$$

Global minutiae-based approaches

The objective of global minutiae-based approaches is to apply a global transformation that allows to maximize the number of resulting paired minutiae.

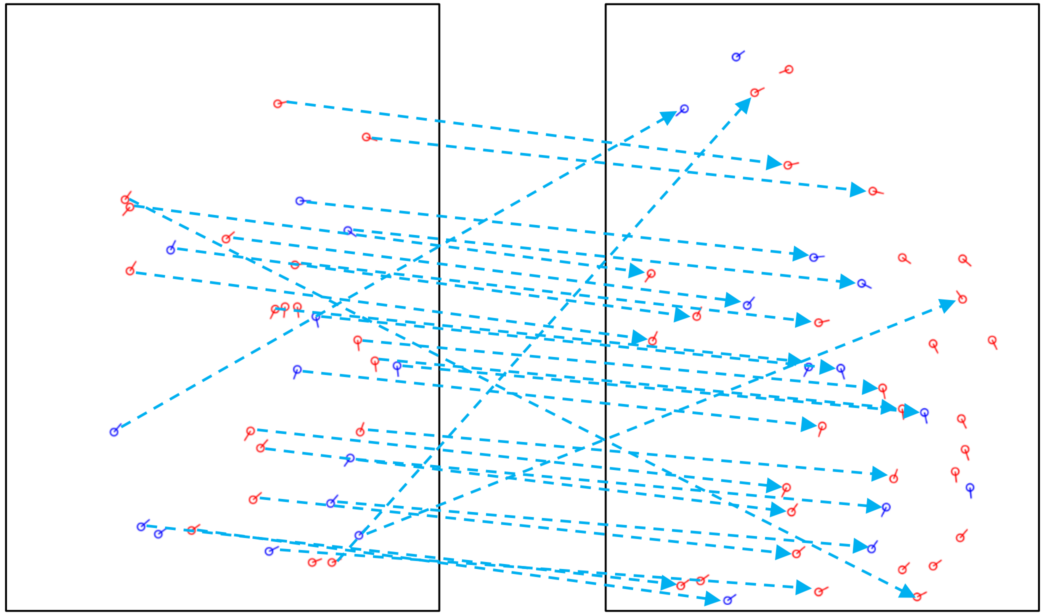


Typically use Hough transform or Ransac implementations to find the best rigid transformation to align two minutiae sets.

Local minutiae-based approaches

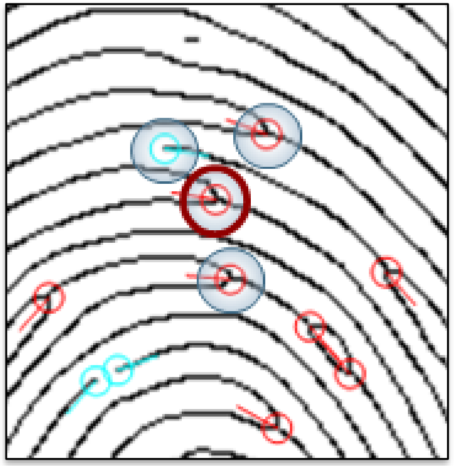
The objective of local minutiae-based approaches is to pair minutiae using local minutiae features invariant to global transformations without a pre-alignment step. Usually they are based on the following steps:

1. for each minutia local features are computed from local minutiae neighborhoods.
2. the minutiae are paired according to local features (fast, robust to distortion but less distinctive).
3. a consolidation step is performed to verify if local matches hold at global level.



Nearest-neighbor-based local structures

The neighbors of the central minutia are formed by its *K* closest minutiae.

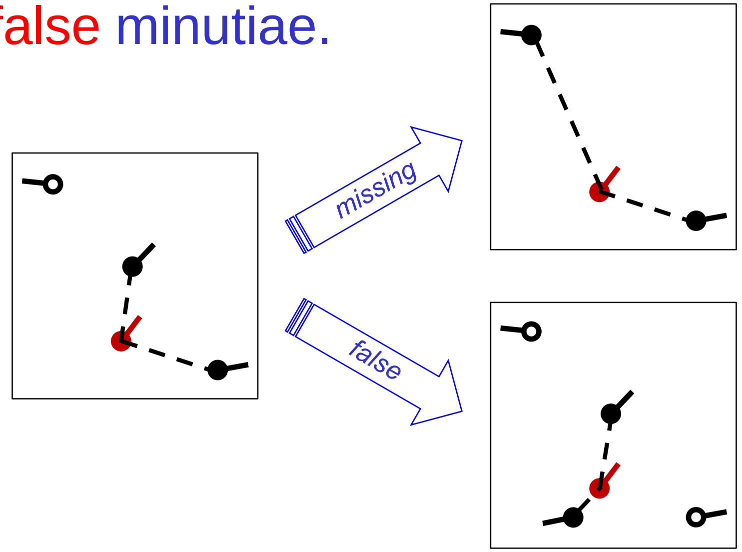


Advantages

- fixed-length descriptors that can be compared very efficiently.

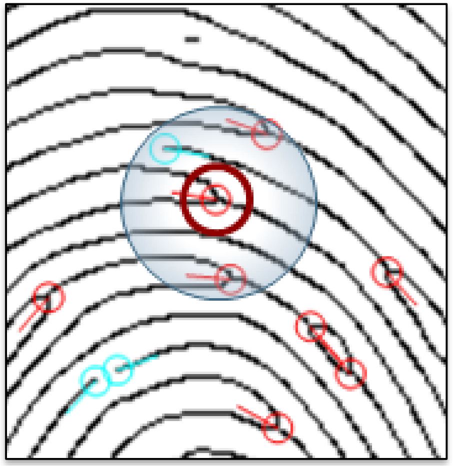
Drawbacks

- possibility of exchanging nearest neighbor minutiae due to missing or false minutiae.



Fixed-radius-based local structures

The neighbors are defined as all the minutiae that are closer than a given radius R from the central minutia.

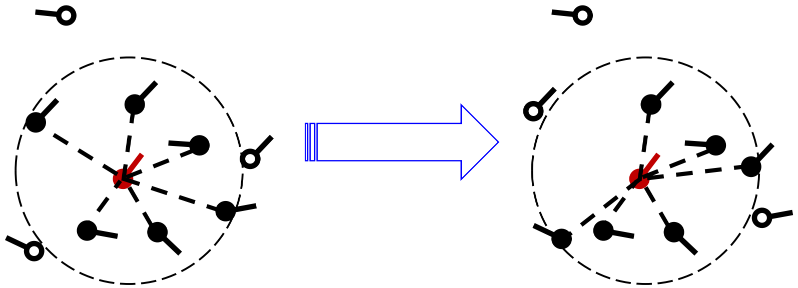


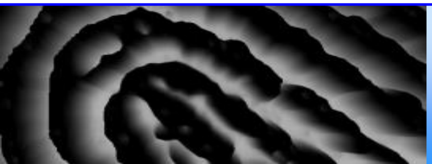
Advantages

- missing and false minutiae are better tolerated.

Drawbacks

- the descriptor length is variable and depends on the local minutiae density leading to a more complex comparison.
- minutiae close to the border can be mismatched because of different local distortion or location inaccuracy.



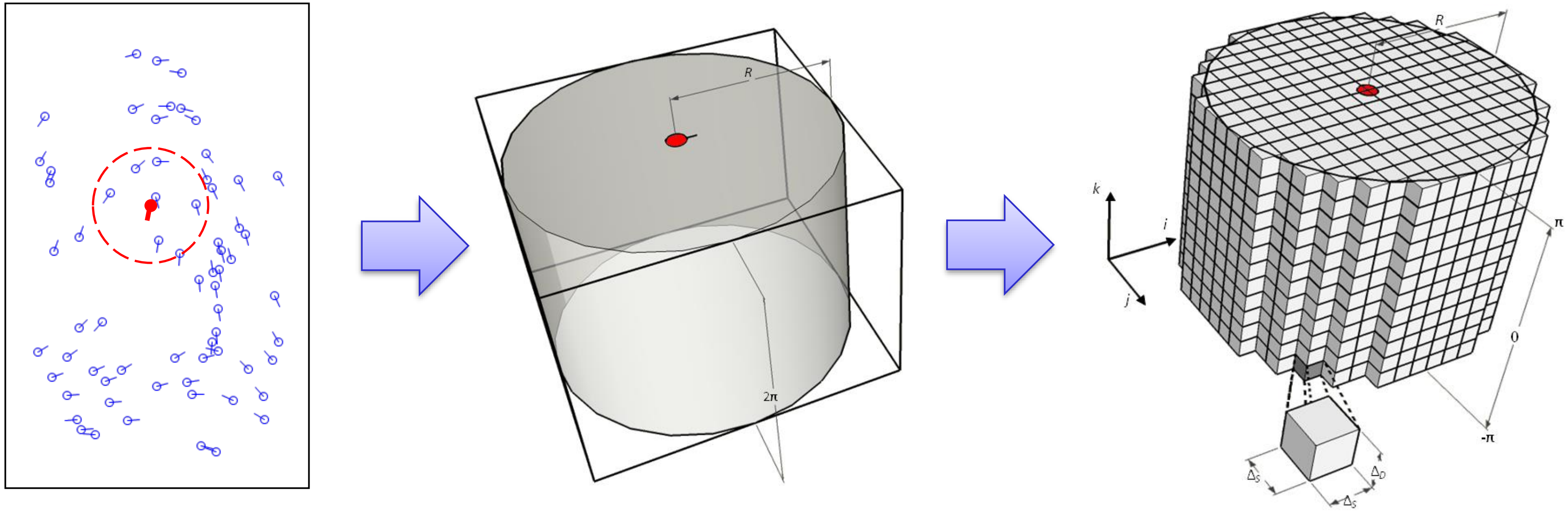


Minutia Cylinder-Code (MCC) (1)

Main advantages:

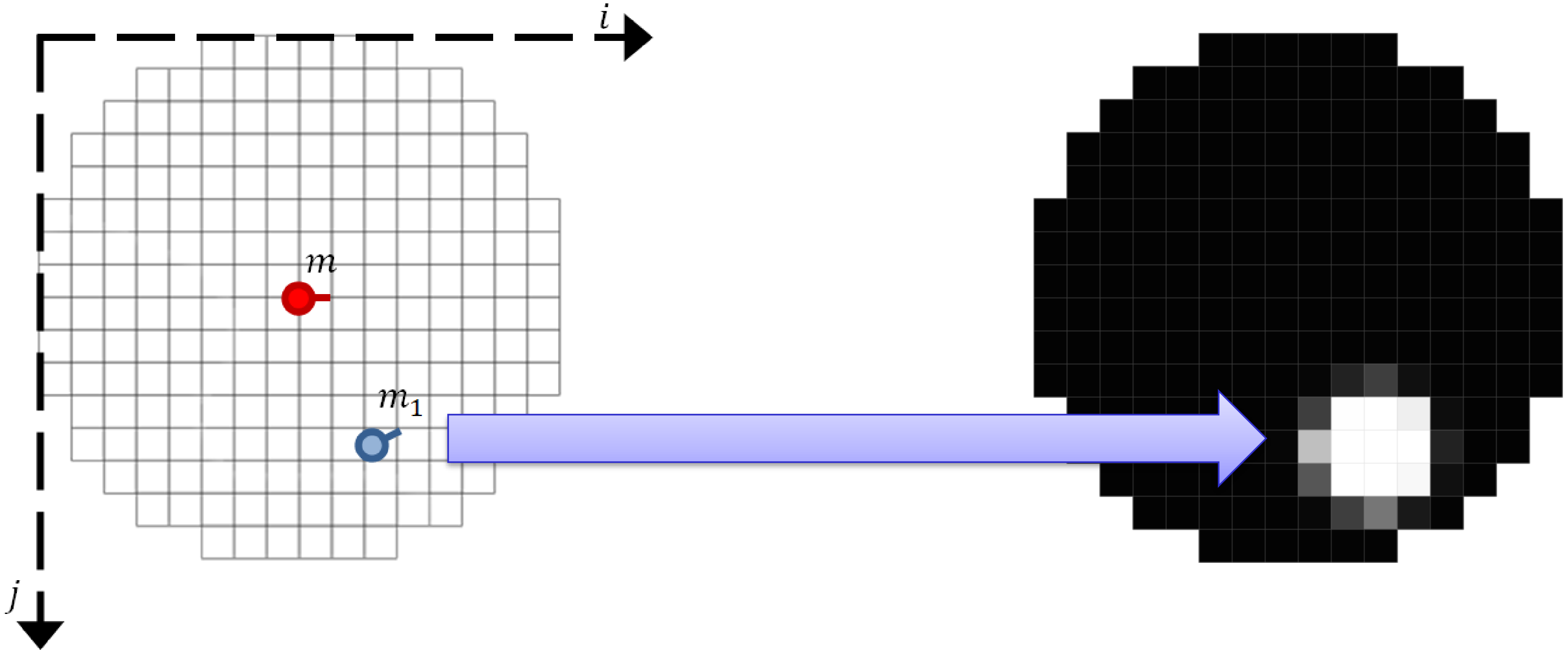
- fixed radius structure;
- fixed-length descriptors;
- tolerates local distortion and small feature extraction errors;
- bit-oriented coding;
- fast and simple local structure comparison phase;

R. Cappelli, M. Ferrara and D. Maltoni, "Minutia Cylinder-Code: a new representation and matching technique for fingerprint recognition", *IEEE tPAMI* 2010.



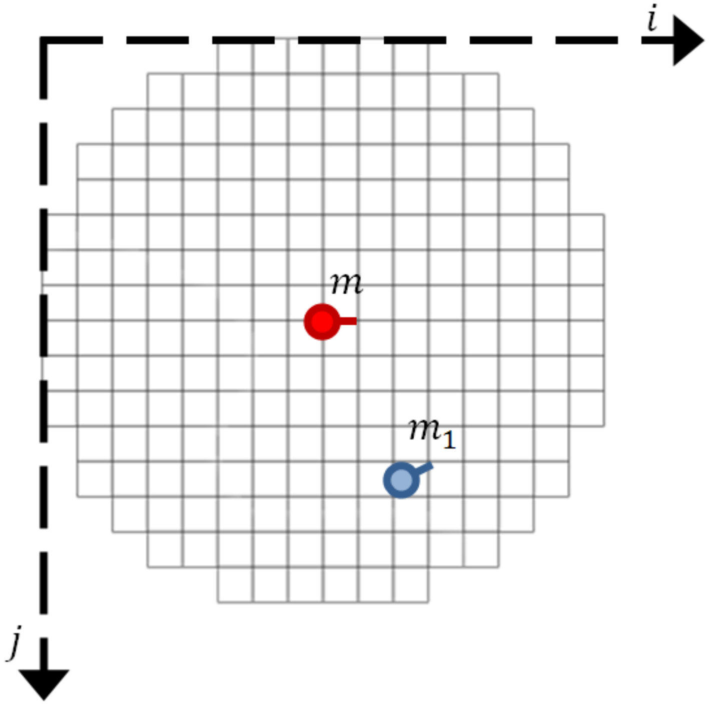
Minutia Cylinder-Code (MCC) (2)

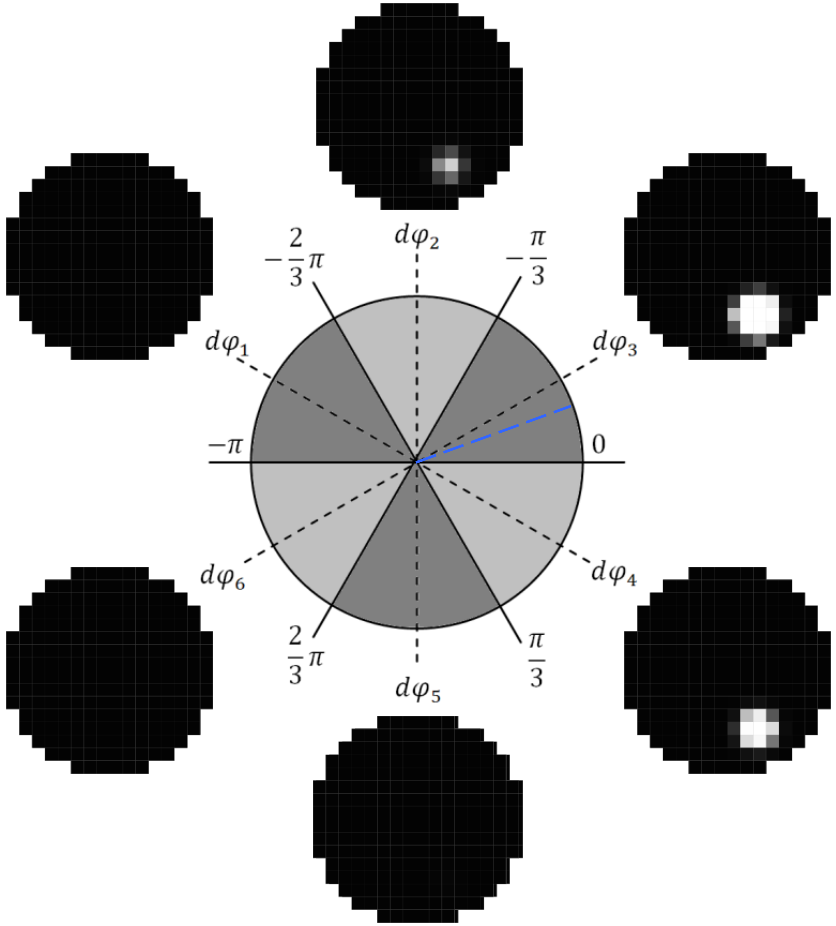
The spatial contribution of the neighbor minutia is spread over cells near its position.



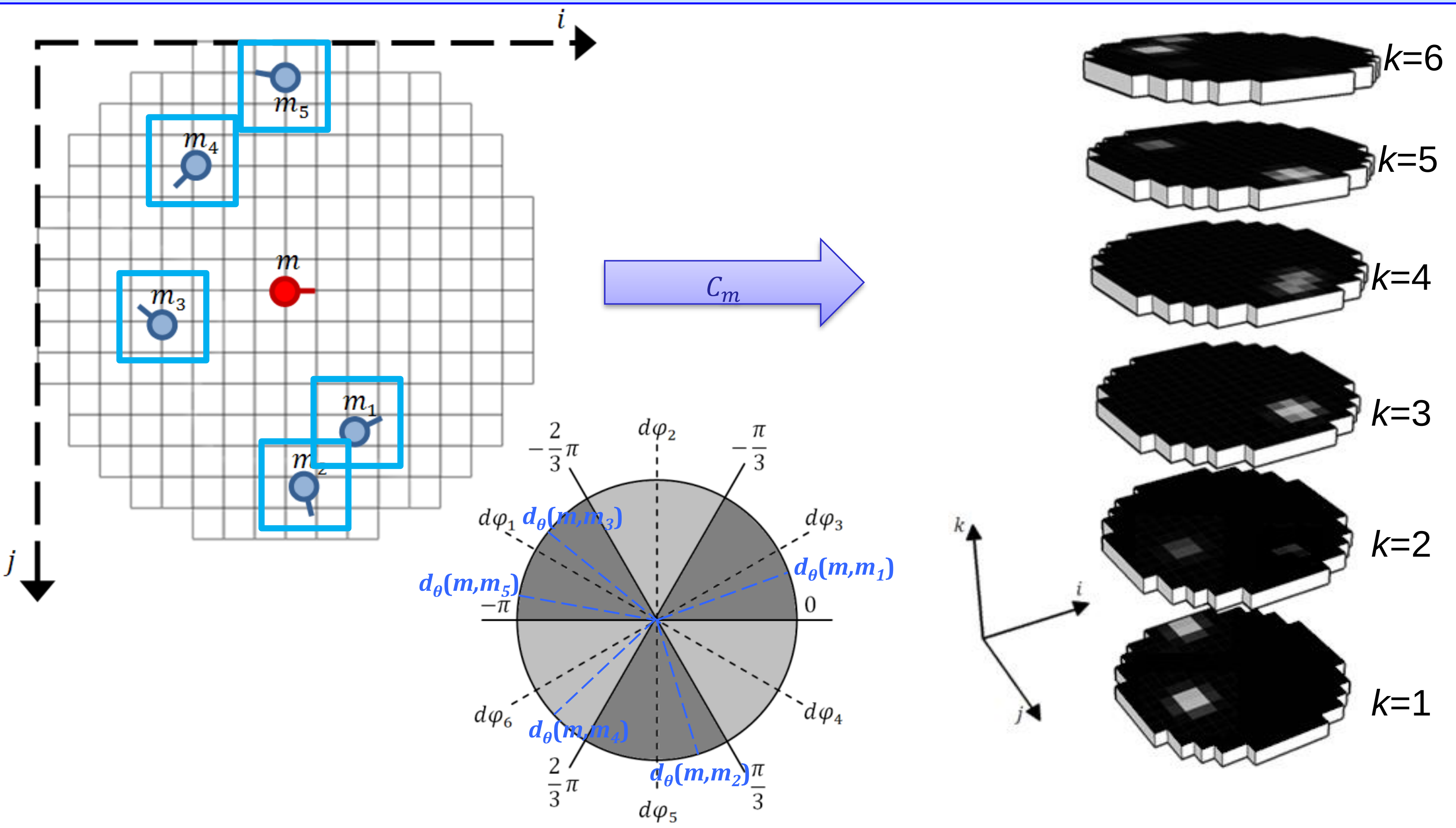
Minutia Cylinder-Code (MCC) (3)

The directional contribution depends on the angle differences.





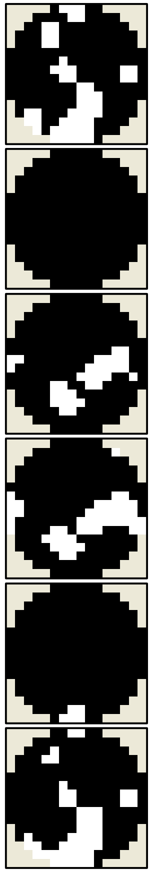
Minutia Cylinder-Code (MCC) (4)



Minutia Cylinder-Code (MCC) (5)

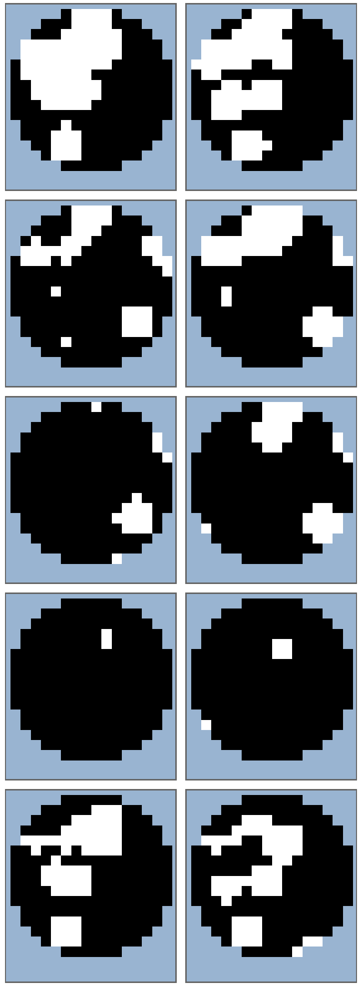
The cylinders can be conveniently converted into bit vectors by applying a unit step function.

Unit step function



Minutia Cylinder-Code (MCC) (6)

C_a C_b



$$\gamma(a,b) = 1 - \frac{\|C_a \text{ XOR } C_b\|}{\|C_a\| + \|C_b\|} = 0.64$$

Minutia Cylinder-Code (MCC) (7)

MCC speed performance

Test: 100 identification queries on a 1M database

Version	System configuration	Comparisons per second
MCC SDK Single core, no SSE optimizations Download: http://biolab.csr.unibo.it/mccsdk.html	Intel CPU E5-2650 @ 2GHz, 64 bit O.S.	18,000
SSE4 Optimized for CPU	Intel CPU E5-2650 @ 2GHz, 64 bit O.S. 2 processors, 32 cores	7 Millions
GPU (CUDA) and CPU Optimized	Intel CPU E5-2650 @ 2GHz, 64 bit O.S. 2 processors, 32 cores 4 Nvidia Tesla C2075 GPUs	42 Millions
GPU (CUDA) and CPU Optimized	Intel CPU Xeon E5-1660 @ 3.2GHz, 64 bit O.S. 1 processor, 8 cores 1 Nvidia Titan RTX GPU	117 Millions

Some references

- D. Maltoni, D. Maio, A.K. Jain and S. Prabhakar, “Handbook of Fingerprint Recognition,” *Springer*, 2009. The 3rd edition is coming (early 2022).
- A.M. Bazen and S.H. Gerez, "Systematic methods for the computation of the directional fields and singular points of fingerprints," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, July 2002.
- L. Hong, Y. Wan, A.K. Jain, “Fingerprint Image Enhancement Algorithms and Performance Evaluation”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 8, pp. 777-789, 1998.
- R. Cappelli, M. Ferrara and D. Maltoni, "Minutia Cylinder-Code: a new representation and matching technique for fingerprint recognition", *IEEE Transactions on Pattern Analysis Machine Intelligence*, vol.32, no.12, pp.2128-2141, December 2010.