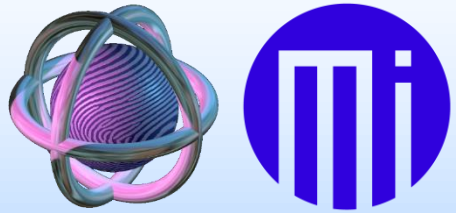
Hands on Fingerprint recognition

A practical solution with Python

Matteo Ferrara

Biometric System Lab. - https://biolab.csr.unibo.it
Machine Intelligence Lab. - https://miatbiolab.csr.unibo.it

University of Bologna - ITALY



IAPR/IEEE WINTER SCHOOL ON BIOMETRICS 2025
12-16 Jan 2025 | Shenzhen, China

Links to the Code

- These slides accompany a practical tutorial on a Jupyter notebook, which can be run locally or on Google Colab.

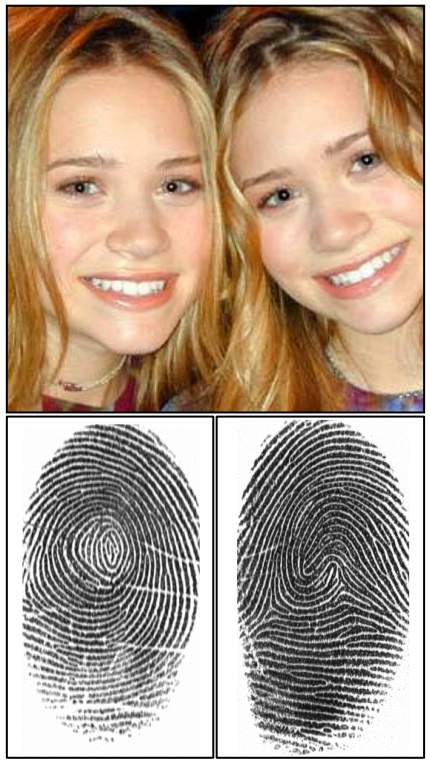
- How to run the notebook locally (recommended):
  - Download https://tinyurl.com/hands-on-fr

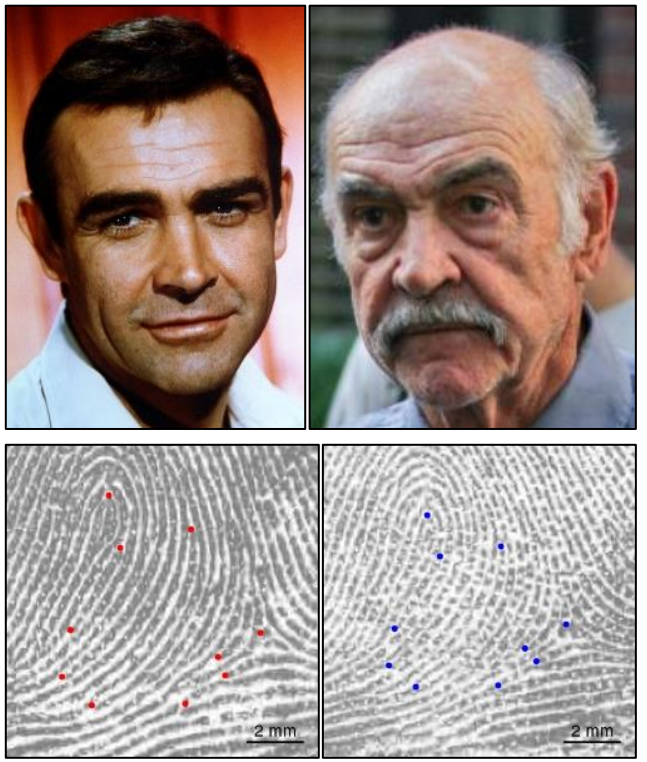
- How to run the notebook on Colab:
  - Open https://colab.research.google.com/drive/1IQYn3nXYJ5xIMyeYaQAhEb1YuF1-imFY?usp=sharing

Why fingerprints?

- Highly distinctive and unique
- Persistent
- Publicly accepted as reliable (evidence in a court of law)

Identical twins have different fingerprints

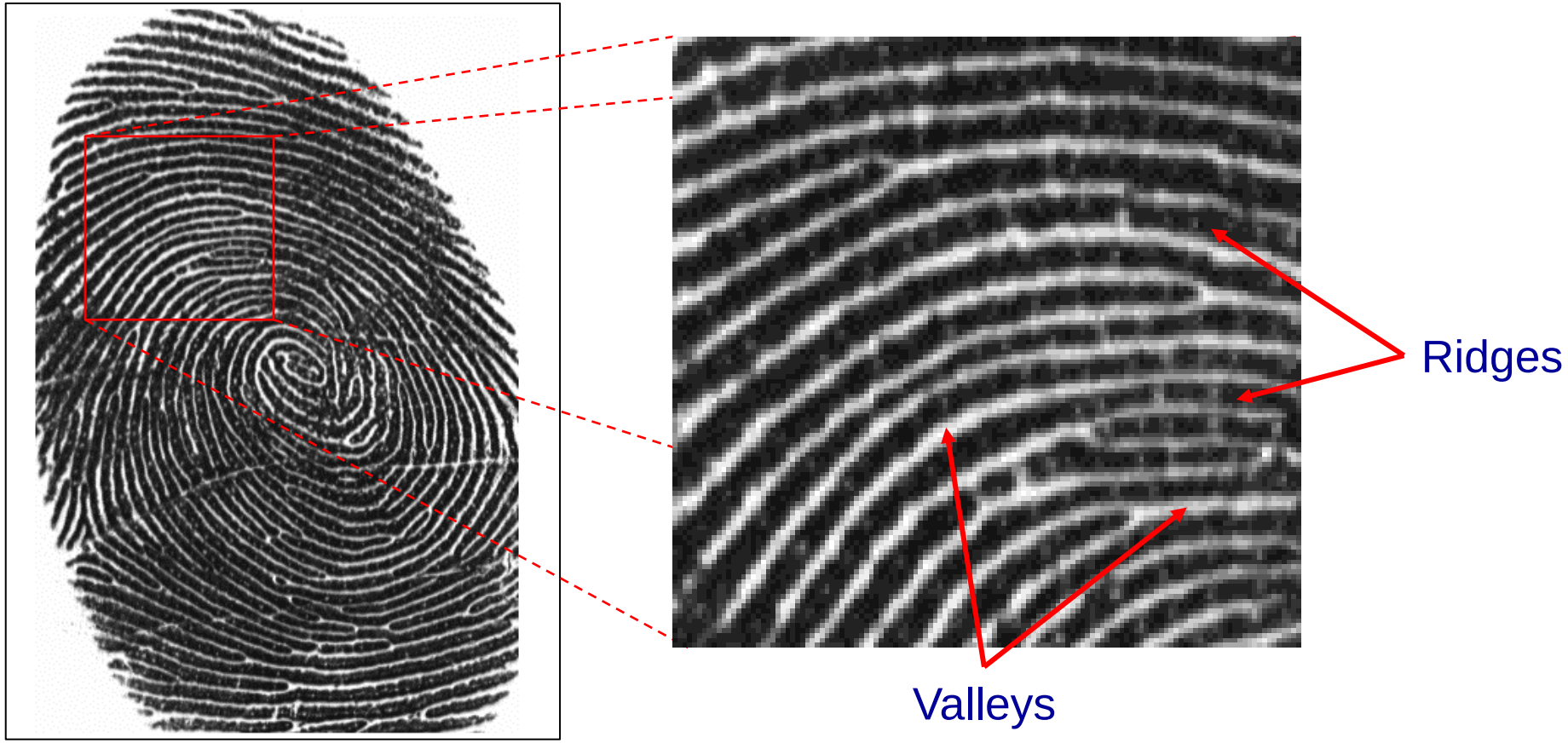




Do not change during the lifetime of a person

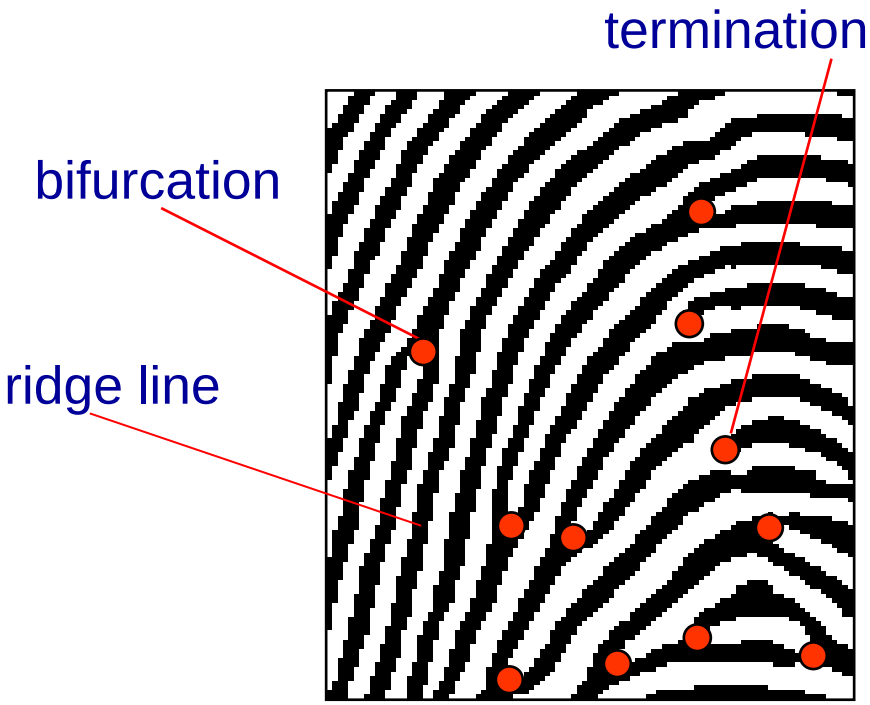
Fingerprint anatomy

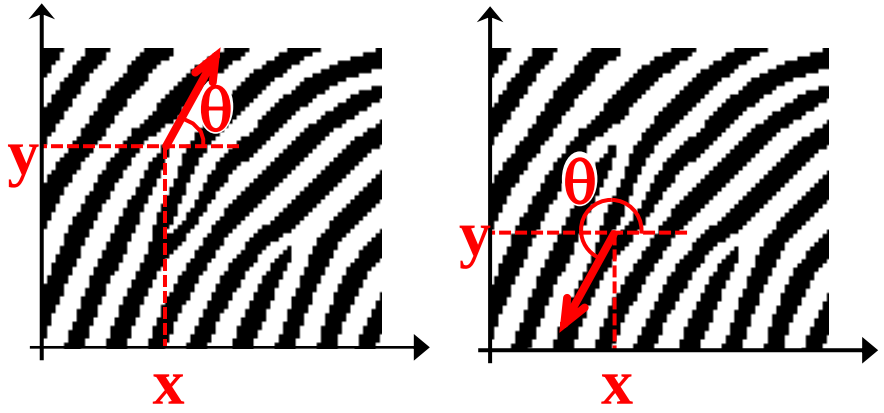
A fingerprint is composed of a set of lines (ridge lines), which mainly flow parallel, making a pattern (ridge pattern).



Minutiae

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

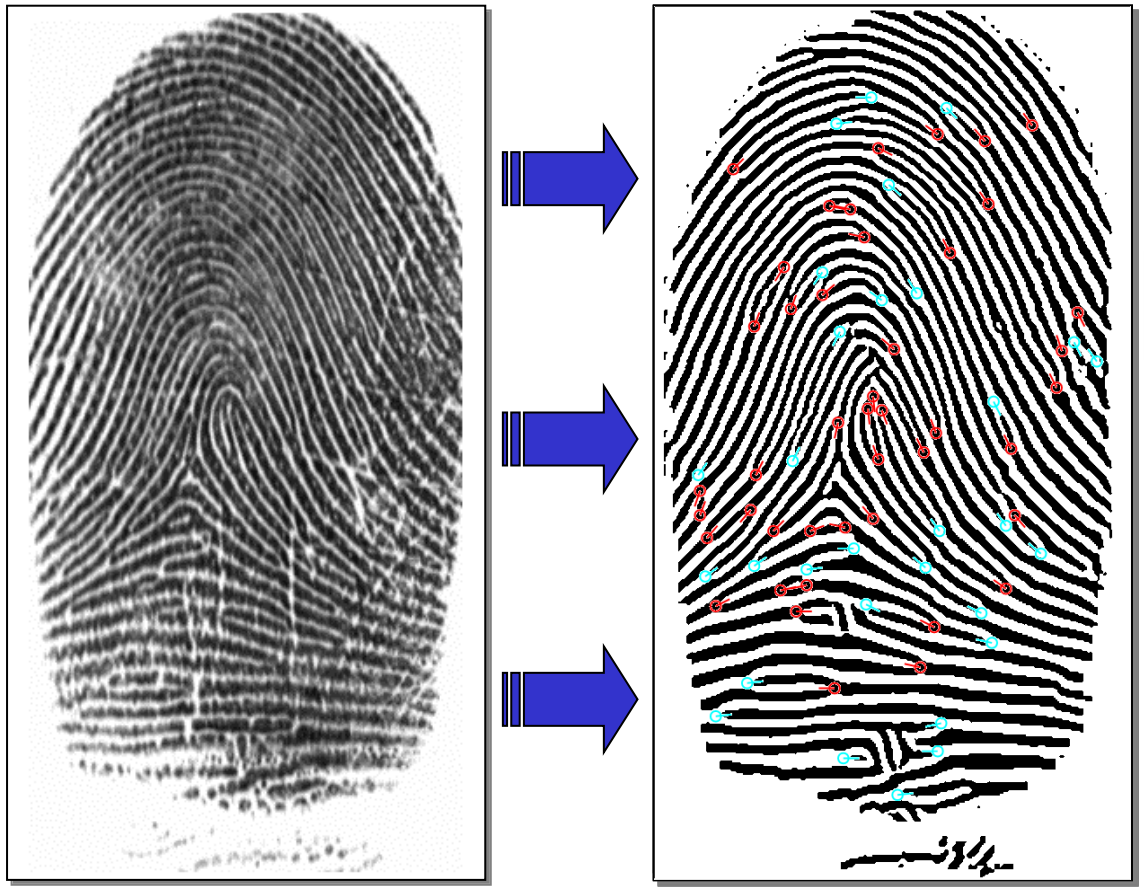


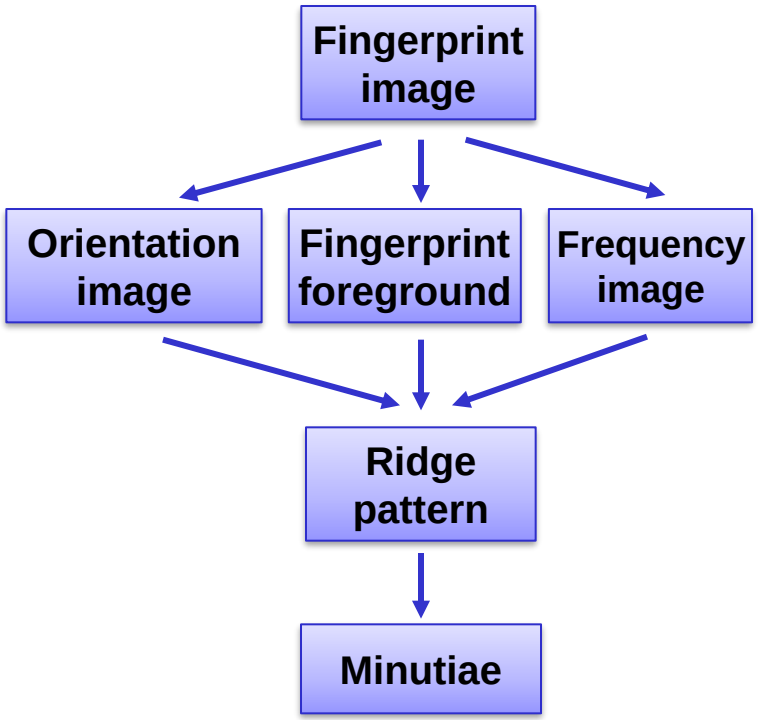


termination bifurcation

Video

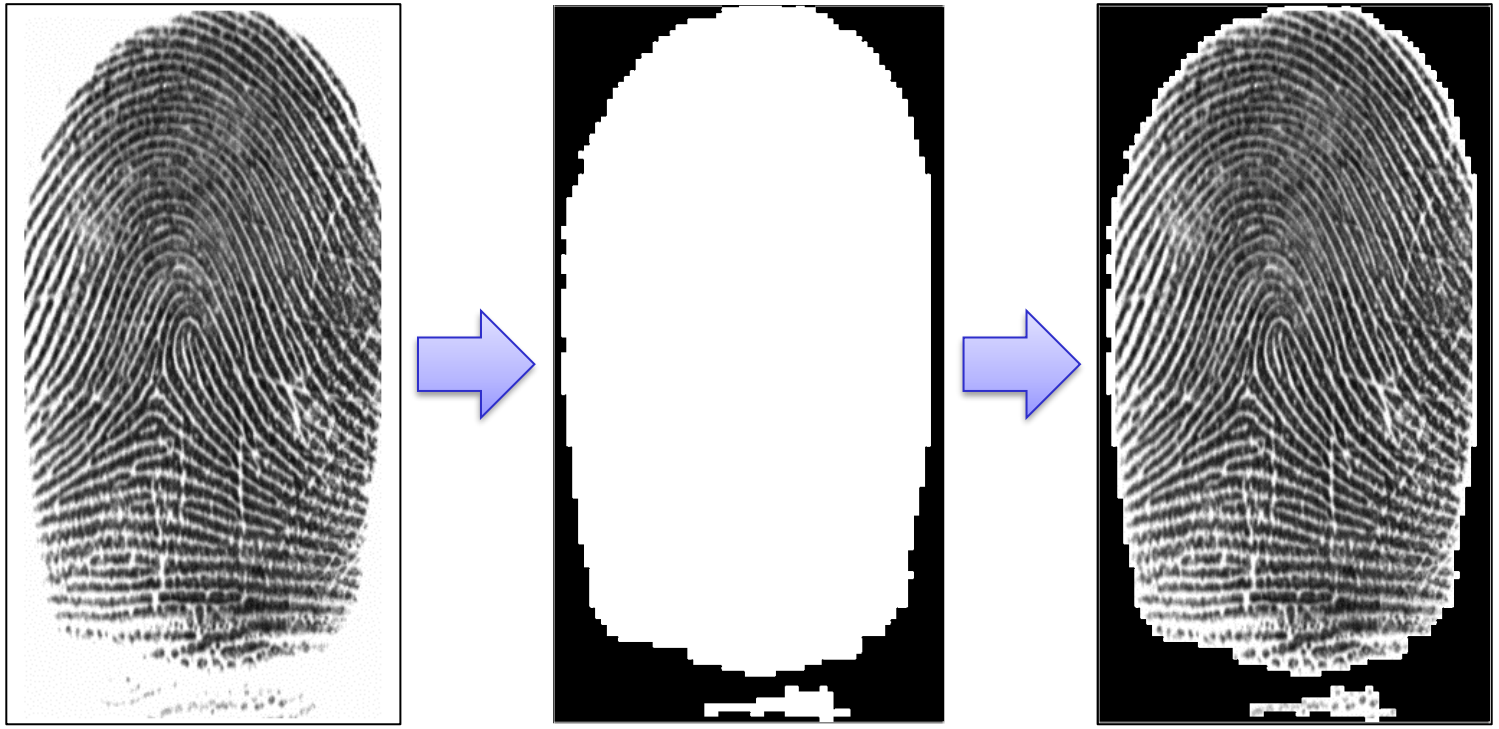
Feature extraction: main steps





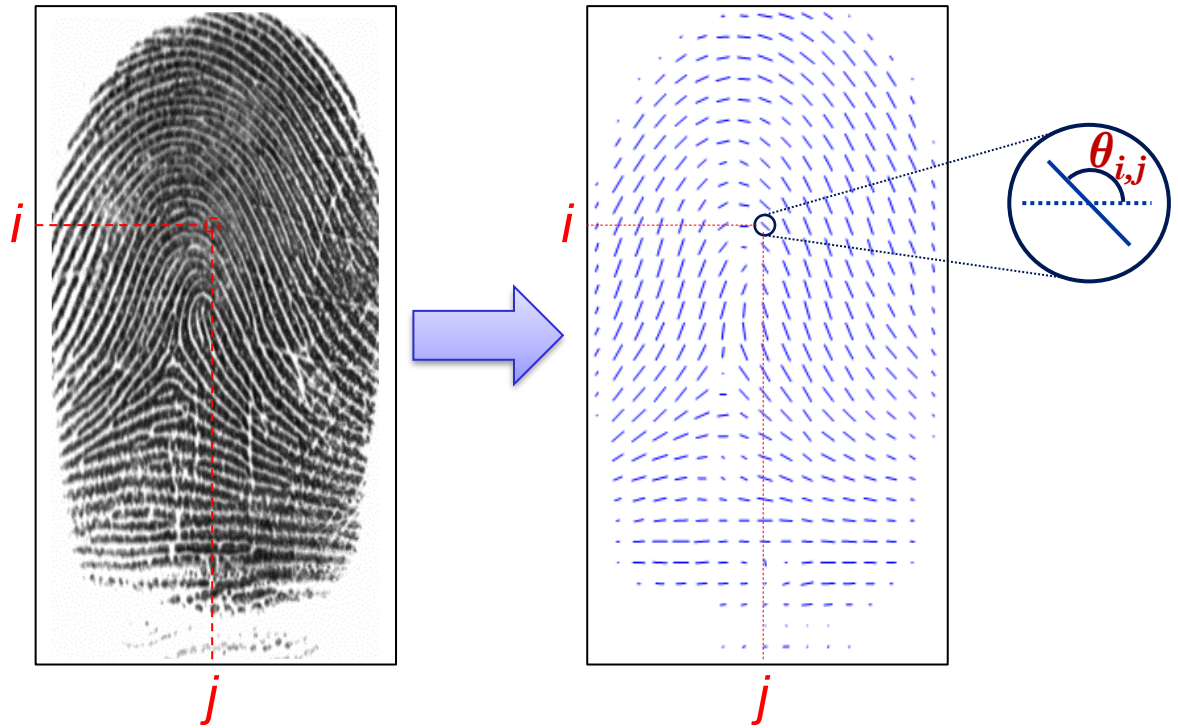
Segmentation

- The segmentation stage is aimed to separate the fingerprint area (foreground) from the image background.
- Foreground is characterized by the presence of a striped and oriented pattern; background presents a uniform pattern.



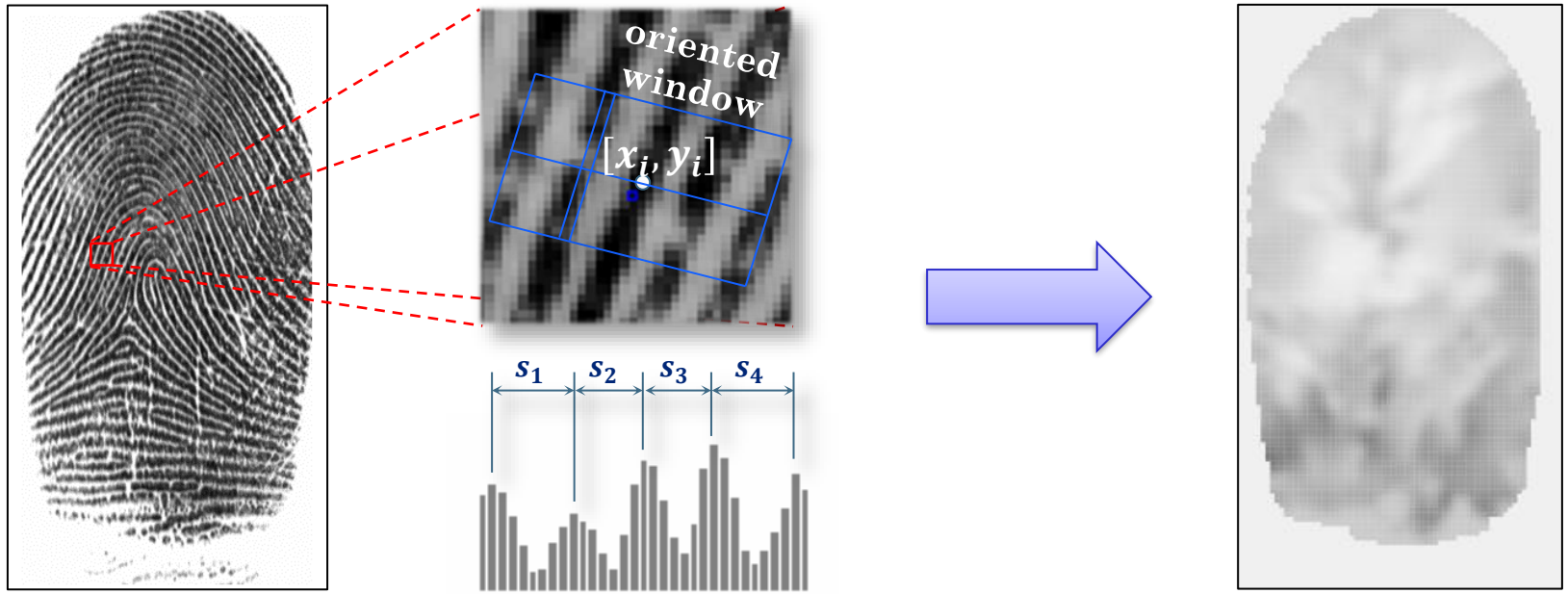
Local ridge orientation

- 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.

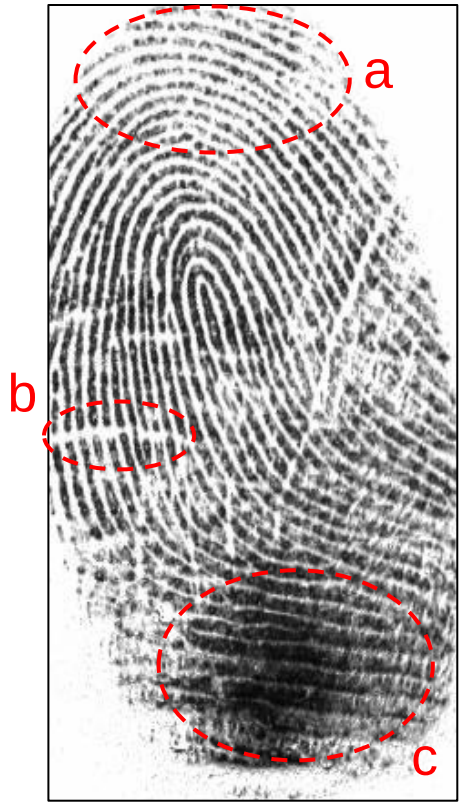


Enhancement (1)

- 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.

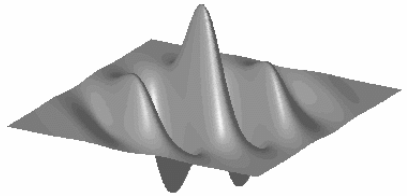
- Typical degradations:
  a. ridge lines are not continuous;
  b. cuts, creases and bruises on the finger;
  c. parallel ridges are not well separated.

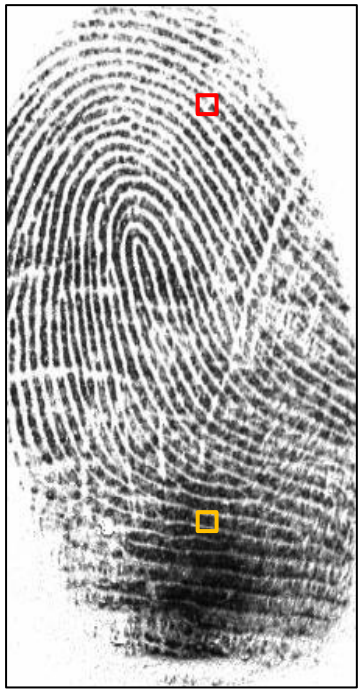
- The most widely used technique for fingerprint enhancement is based on contextual filters.
- In contextual filtering, the characteristics of the filter used change according to the local context.

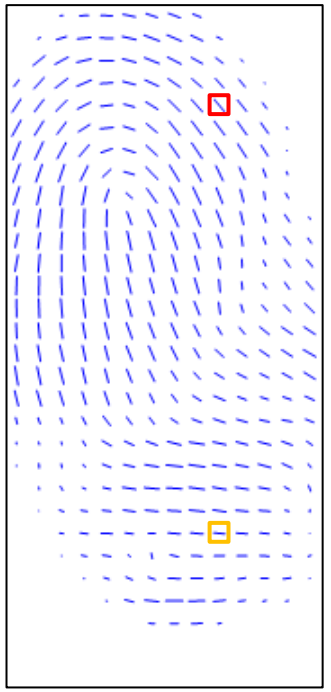


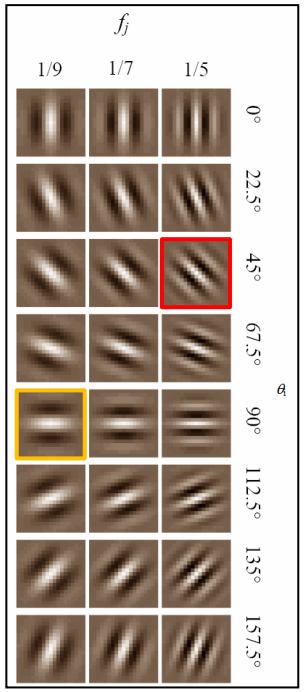
Enhancement (2)

- The local context of a fingerprint is represented by the ridge orientation and frequency.
- Gabor filter: sinusoidal plane wave tapered by a Gaussian.

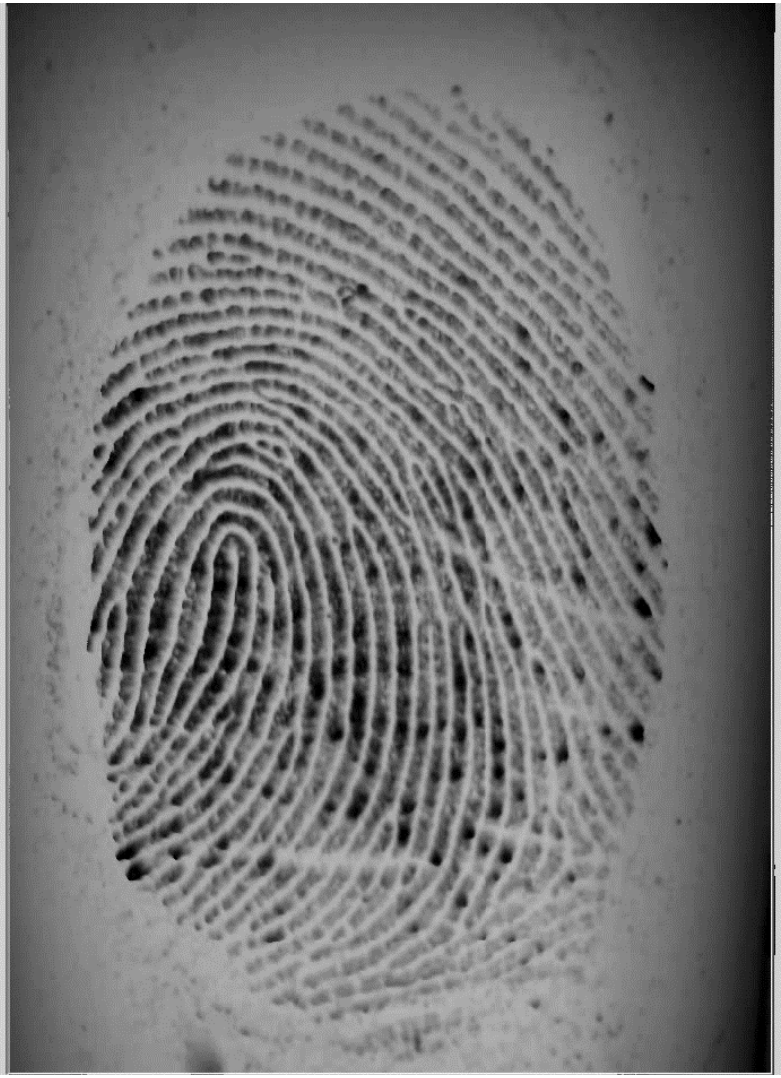








Enhancement (3)



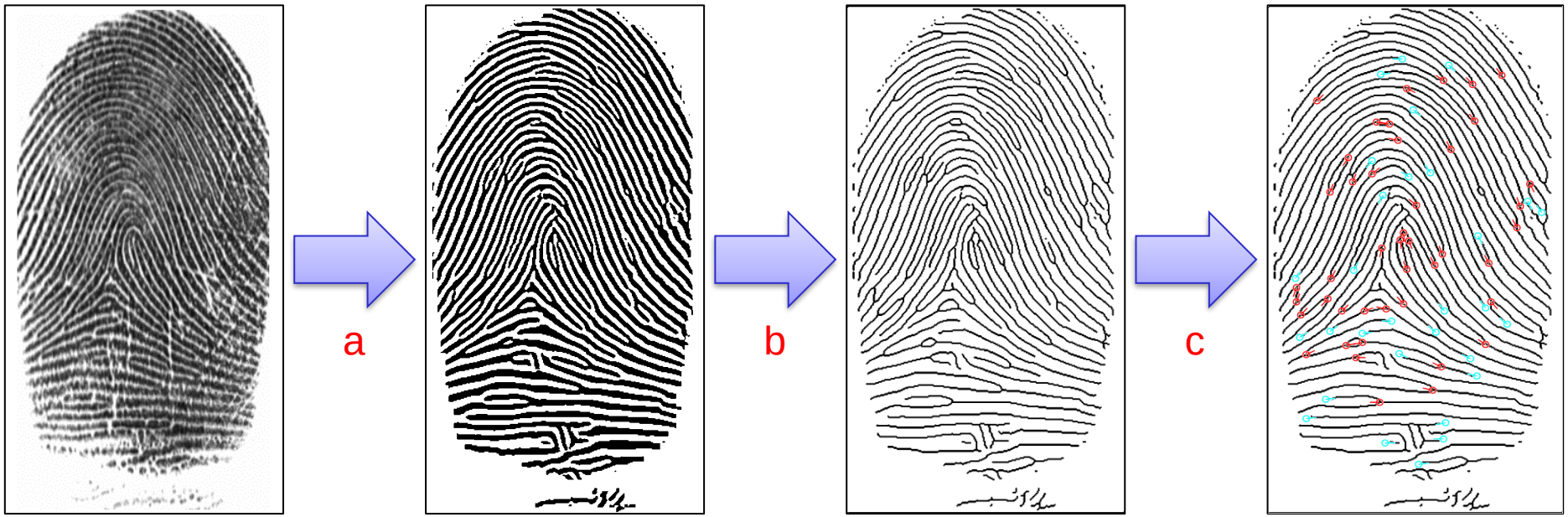
Minutiae detection (1)

Traditional approach:

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

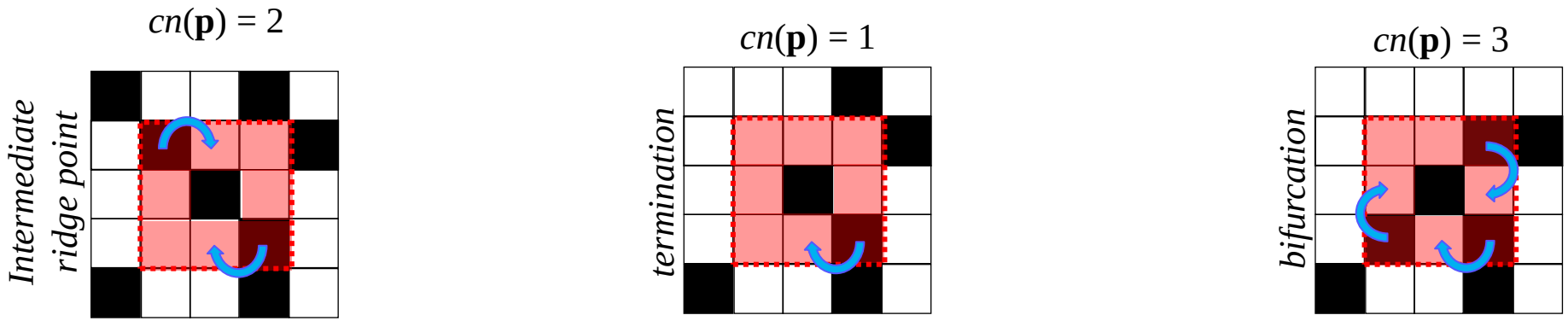
b. Thinning: the binary image is submitted to a thinning stage aimed to reduce the ridge thickness to one pixel;

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*):



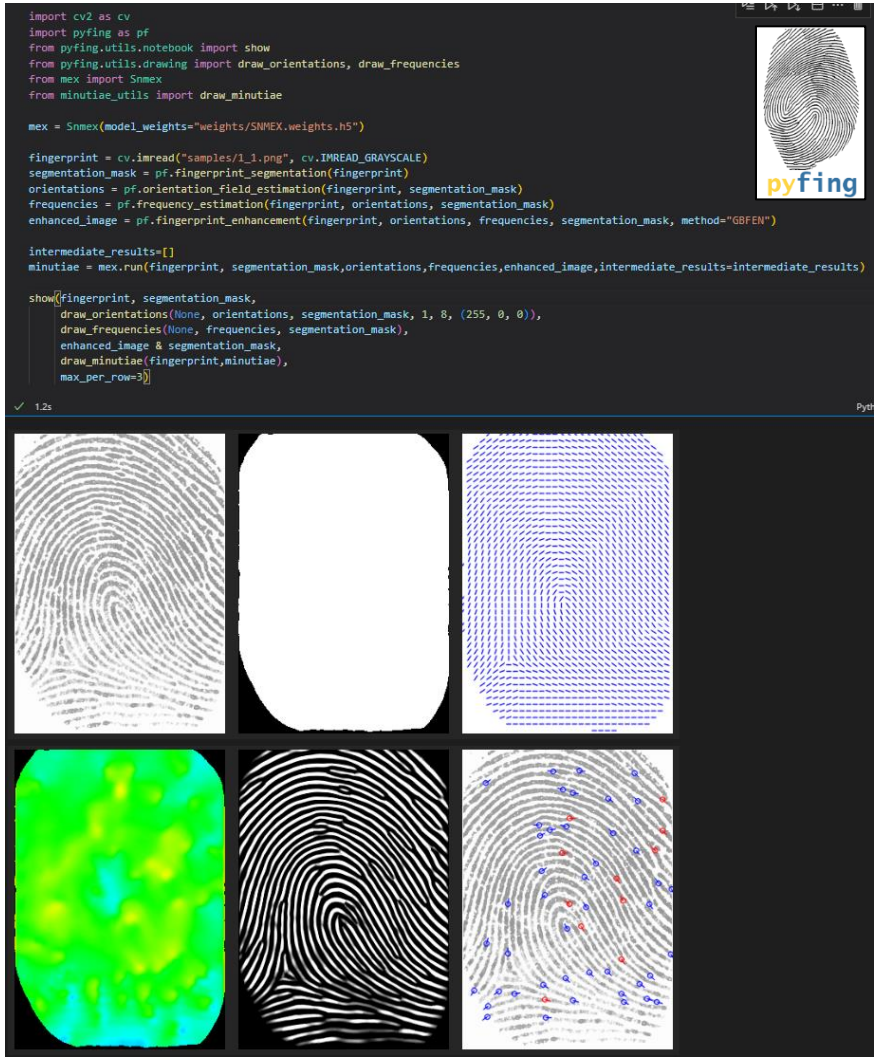
It is simple to note that a black pixel **p**:

- is an intermediate ridge point if *cn*(**p**)=2;
- is a termination if *cn*(**p**)=1;
- is bifurcation if *cn*(**p**)=3;
- part of a more complex minutiae if *cn*(**p**)> 3.

The pyfing library: A Python Library for Fingerprint Analysis

- Open-source fingerprint recognition building blocks in Python
- Includes both traditional and deep-learning methods
- Based on well-known libraries (Numpy, OpenCV, Keras)
- State-of-the-art methods (as confirmed by the results reported on the corresponding papers)
- Includes simple tools for visualization of the main fingerprint features
- Modular and extensible: the library is designed to be easily extended with new algorithms

https://github.com/raffaele-cappelli/pyfing



The pyfing library: Status of the Project

Fingerprint Segmentation

Traditional method: GMFS

Deep-learning method: SUFS

Paper published in 2023

State-of-the-art results on the FVC fingerprint segmentation benchmark

Orientation field estimation

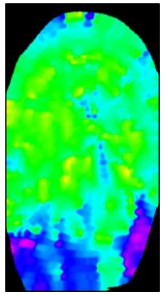
Traditional method: GBFOE

Deep-learning method: SNFOE

Paper published in 2024

State-of-the-art results on the FOE and NIST SD27 benchmarks

Frequency estimation



Traditional method: XSFFE

Deep-learning method: SNFFE

Paper published in 2024

State-of-the-art results on the new FFE benchmark

Fingerprint Enhancement

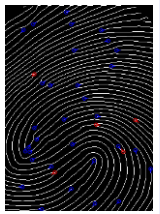
Traditional method: GBFEN

Deep-learning method: SNFEN

Paper under review

State-of-the-art results on the NIST SD27 benchmark

Minutia extraction

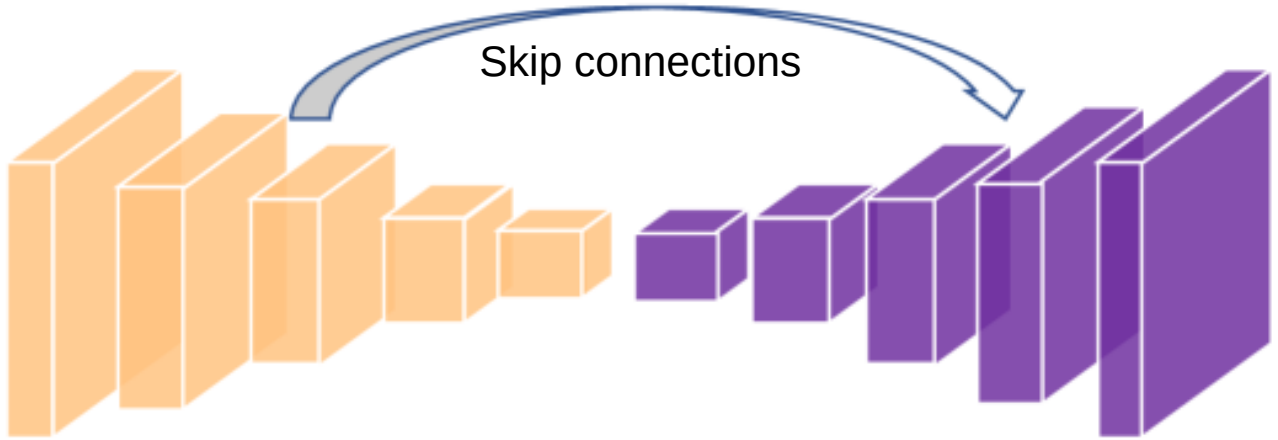


Methods under development

…

The pyfing library: General Architecture

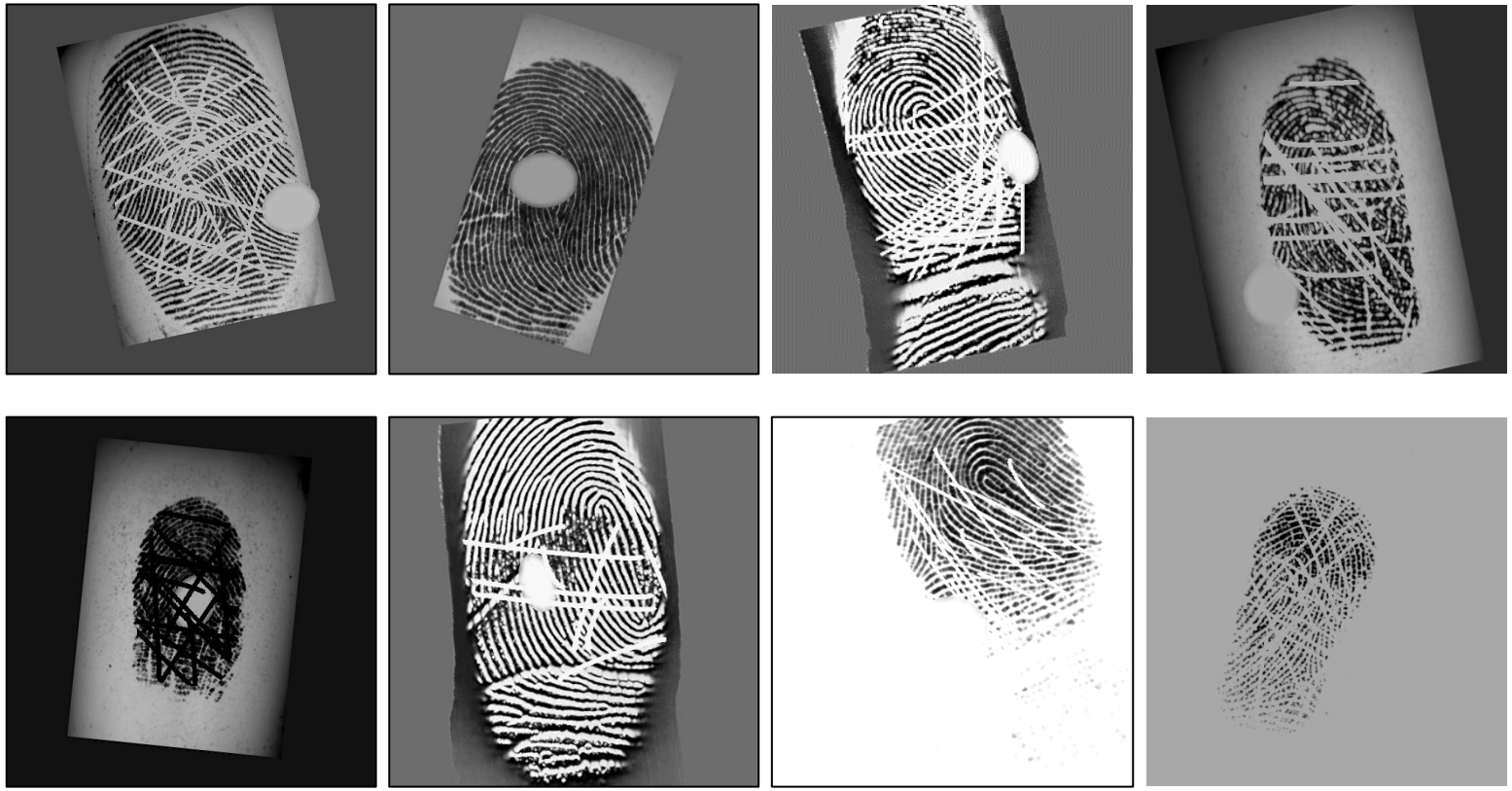
- All deep learning solutions in pyfing share the same general architecture: a fully convolutional encoder-decoder.
- It is characterized by the following features:
  - an encoding path to capture contextual information
  - a decoding path to achieve precise positioning
  - skip connections to link the encoding path with the decoding path, retrieving spatial information lost during down-sampling.



The pyfing library: Data Augmentation

Due to the limited availability of fingerprints in the training datasets, fingerprint-specific augmentation techniques were implemented, including:

- random translation
- rotation
- scale
- horizontal flip
- gamma correction
- contrast reduction
- morphological operations
- simulated scratches and abrasions

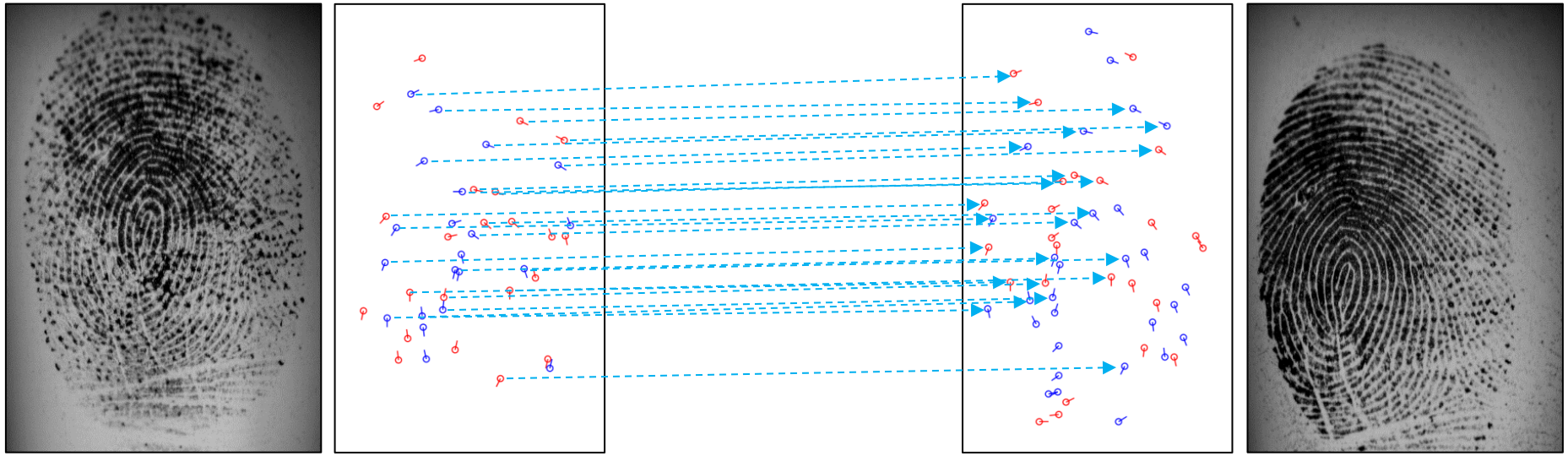


Fingerprint Matching

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.

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



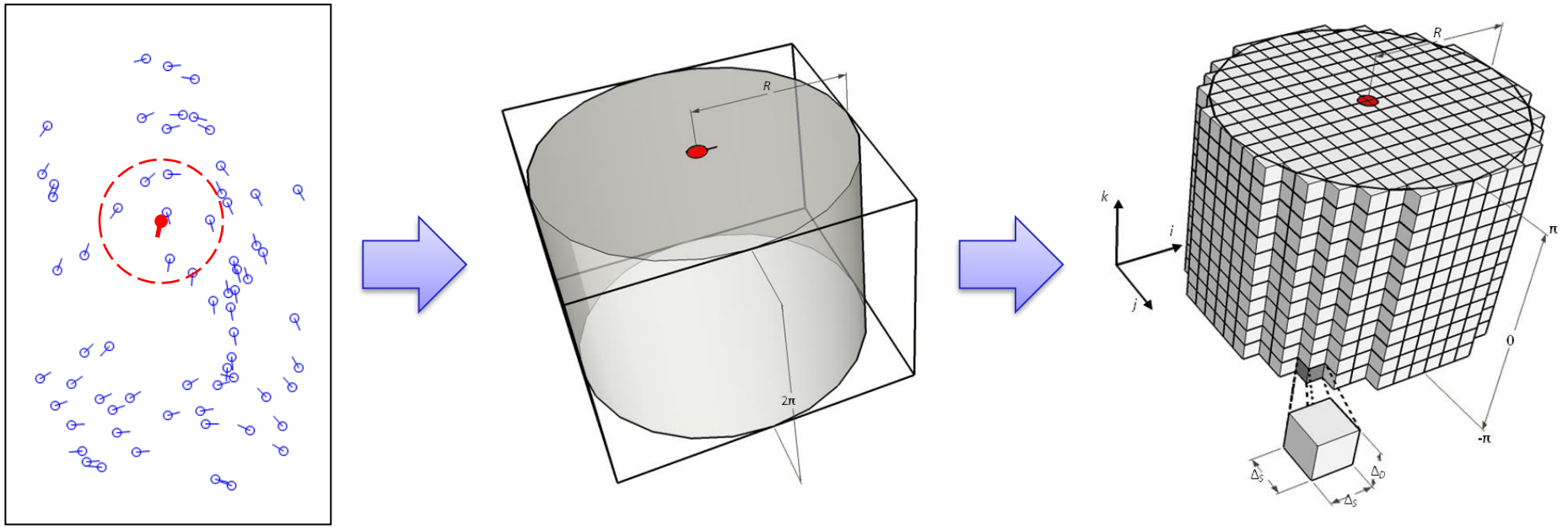
$$score = \frac{\#pairs}{(u+v)/2}$$

Minutia Cylinder-Code (MCC)

The main advantages of MCC are:

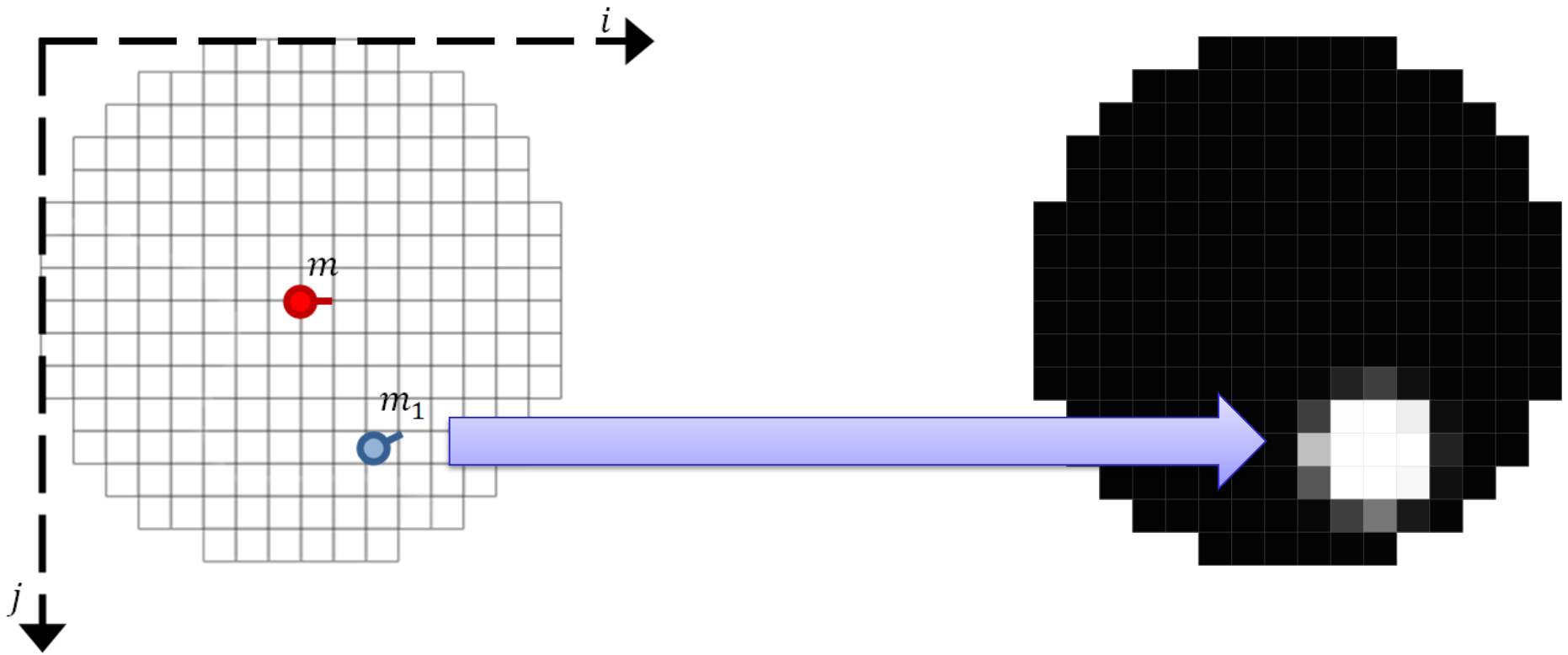
- fixed radius structure;
- fixed-length descriptors;
- toleration of local distortion and small feature extraction errors;
- bit-oriented coding;
- fast and simple local structure comparison phase;

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

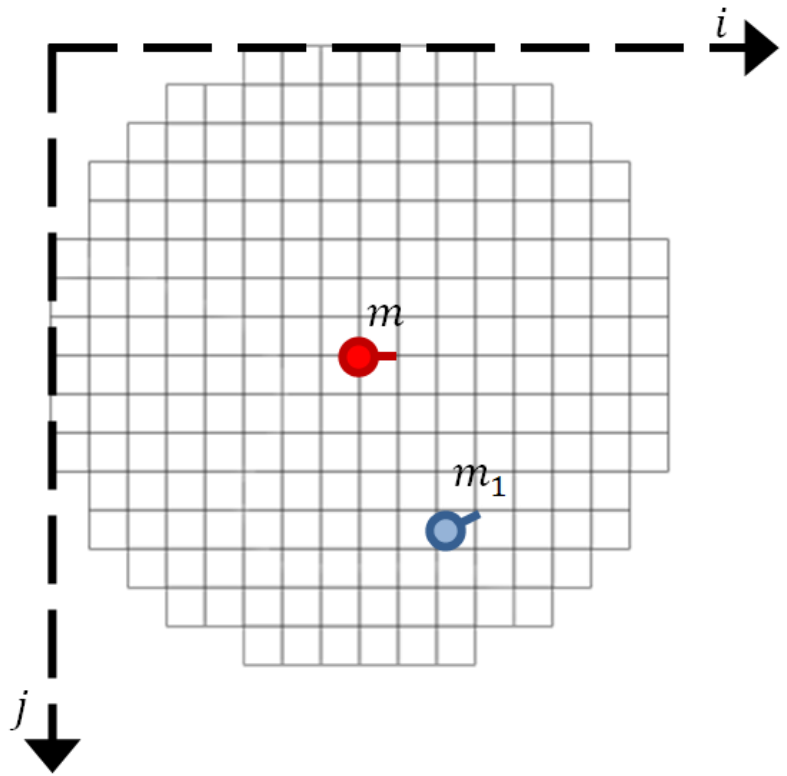


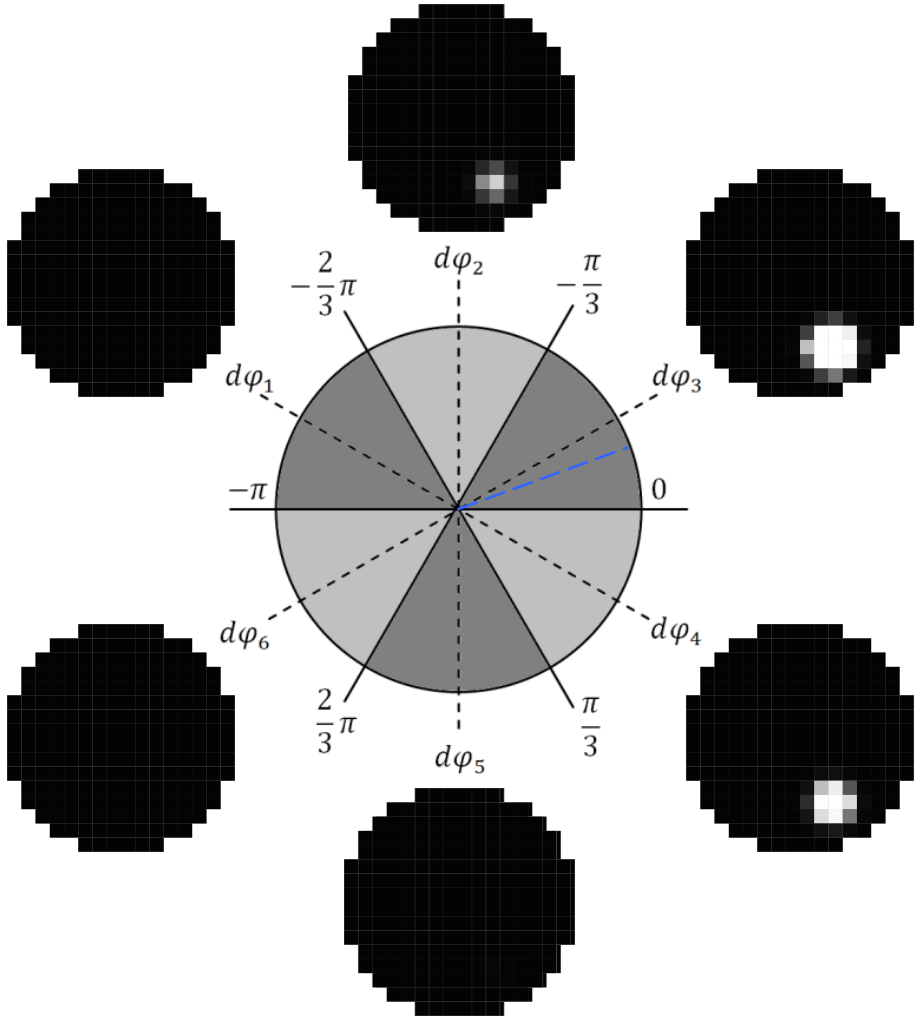
Minutia Cylinder-Code (MCC) (2)

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

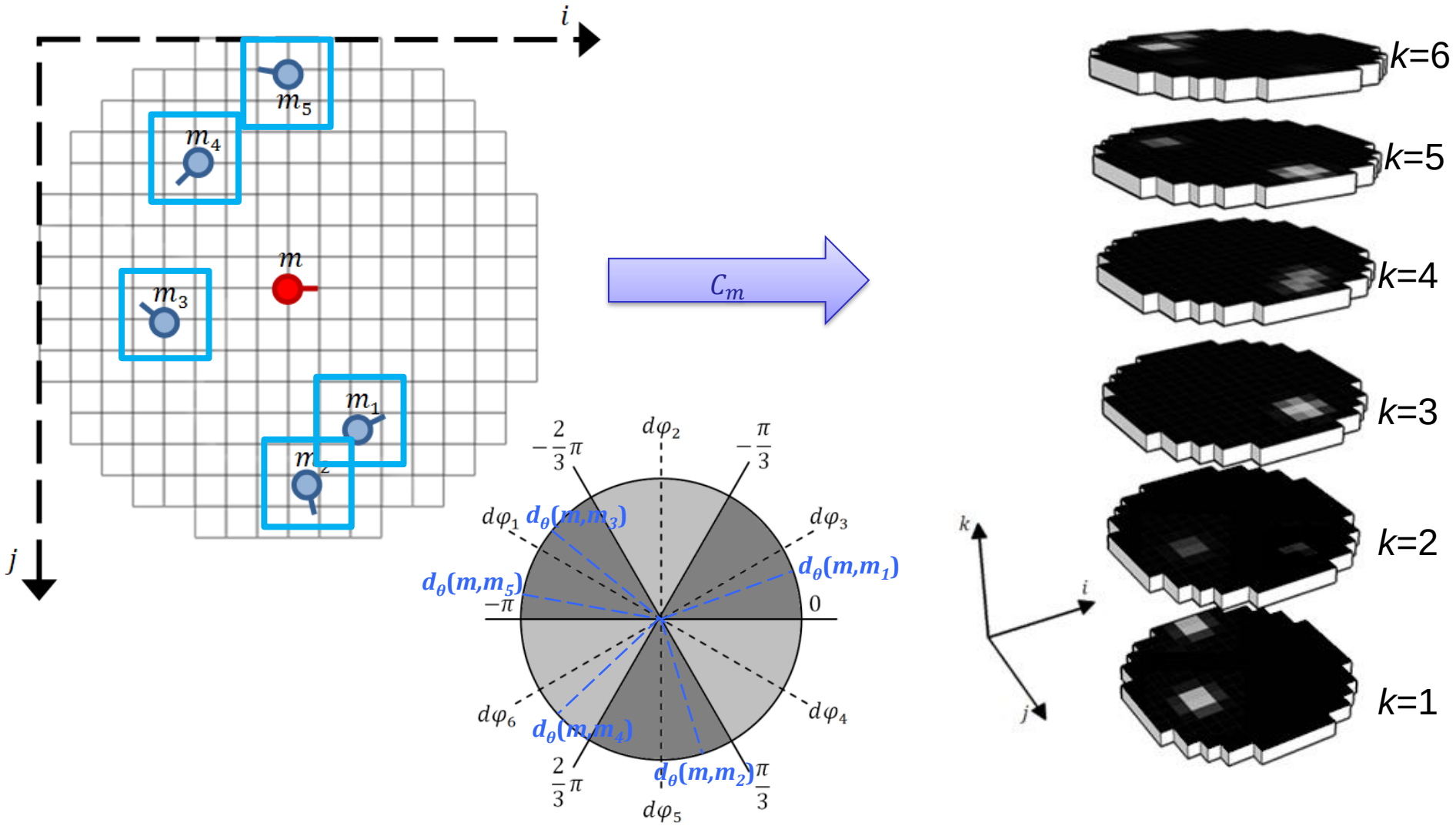


Minutia Cylinder-Code (MCC) (3)



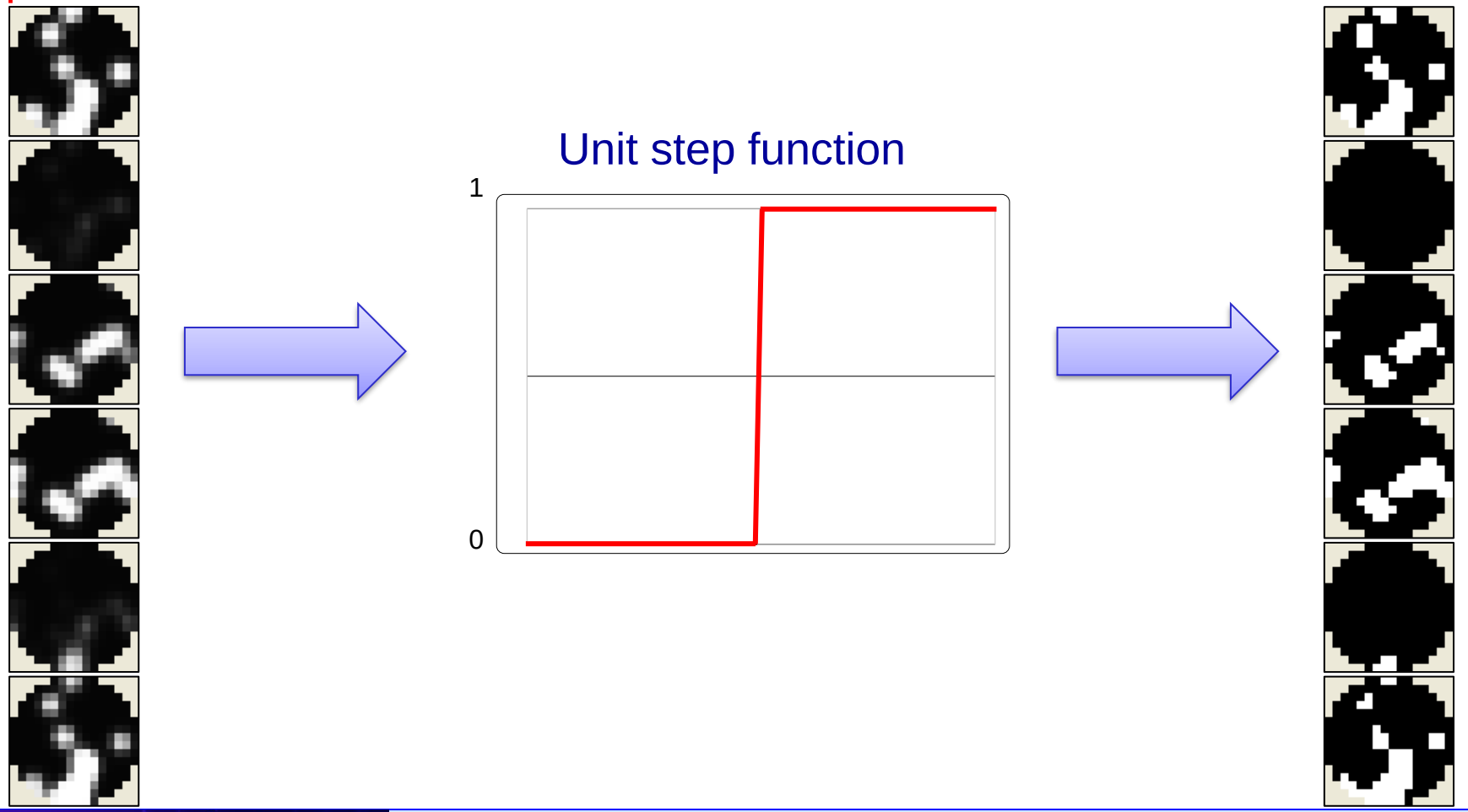


Minutia Cylinder-Code (MCC) (4)

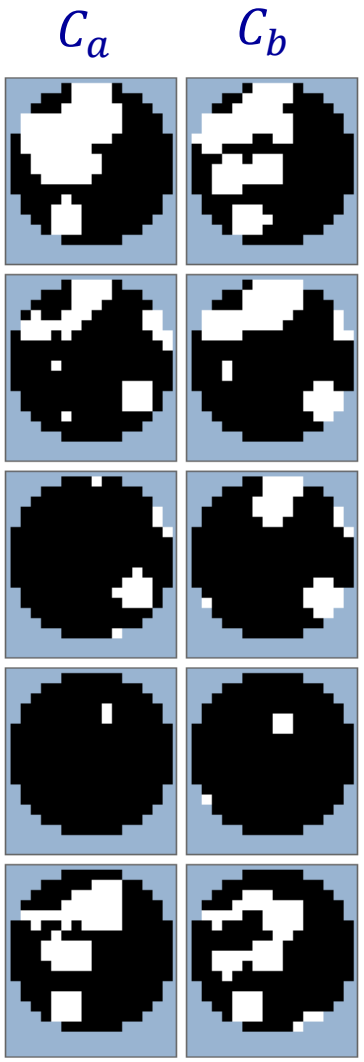


Minutia Cylinder-Code (MCC) (5)

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



Minutia Cylinder-Code (MCC) (6)



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

Minutia Cylinder-Code (MCC) (7)

MCC speed performance

Test: 100 identification queries on a 1M database

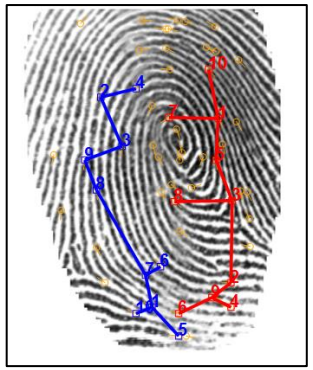
Version	System configuration	Comparisons per second
MCC SDK - Single core, no SSE optimizations Download: http://biolab.csr.unibo.it/mccsdk.html	Intel CPU E5-2650 @ 2GHz, 64 bit O.S.	18,000
SSE4 Optimized for CPU	Intel CPU E5-2650 @ 2GHz, 64 bit O.S. 2 processors, 32 cores	7 Millions
GPU (CUDA) and CPU Optimized	Intel CPU E5-2650 @ 2GHz, 64 bit O.S. 2 processors, 32 cores 4 Nvidia Tesla C2075 GPUs	42 Millions
	Intel CPU Xeon E5-1660 @ 3.2GHz, 64 bit O.S. 1 processor, 8 cores 1 Nvidia Titan RTX GPU	117 Millions
	Intel CPU Xeon W-2133 @ 3.6GHz, 64 bit O.S. 1 processor, 12 cores 1 NVIDIA GeForce RTX 4090 GPU	204 Millions

Main challenges

Nowadays research on fingerprints is mainly active on:

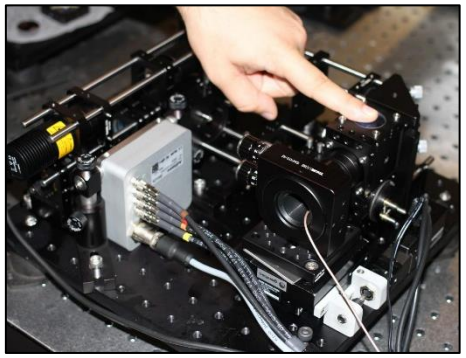
Fake fingerprints

Double-identity fingerprints



Altered fingerprints

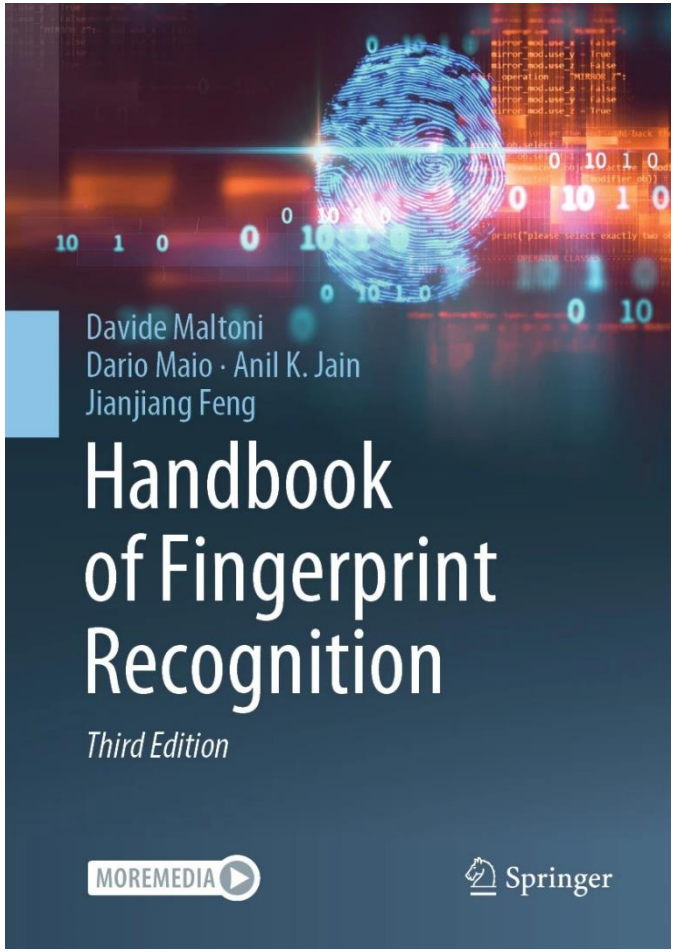
New sensors



Latent fingerprints

Template protection

Handbook of Fingerprint Recognition



D. Maltoni, D. Maio, A.K. Jain and J. Feng, Handbook of Fingerprint Recognition (Third Edition), *Springer Nature*, 2022.

Thank you for your attention

matteo.ferrara@unibo.it
https://biolab.csr.unibo.it
https://miatbiolab.csr.unibo.it