Fingerprint Image Enhancement and Minutiae Extraction: Reproduction and Evaluation of Hong et al.

ELEN-4830 Digital Image Processing

Presenters:
Aditya Jabade
Rutvija Deshpande

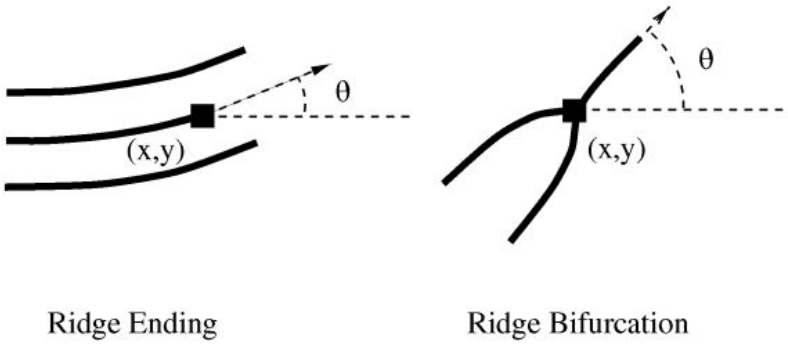
May 14, 2025

COLUMBIA | ENGINEERING
The Fu Foundation School of Engineering and Applied Science

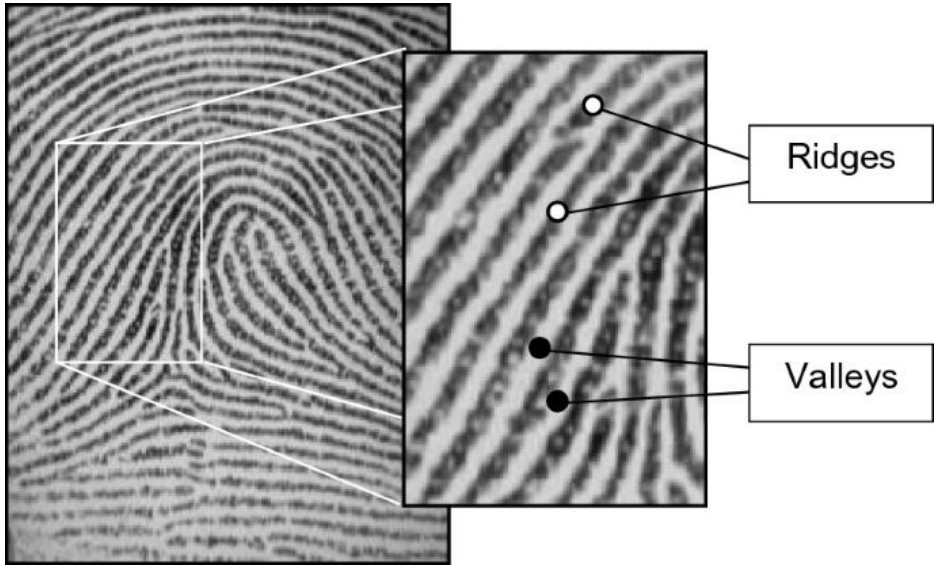
Pre-Motivation

What is a fingerprint!?

- A pattern of ridges and valleys.
- Uniqueness exclusively determined by local ridge characteristics.
- Prominent characteristics: Ridge ending, Bifurcation called ‘minutiae’ (*mi·noo·shee·uh*)



Ridge Ending Ridge Bifurcation



Reference: https://biometrics.cse.msu.edu/Publications/Fingerprint/MSU-CPS-97-35fenhance.pdf

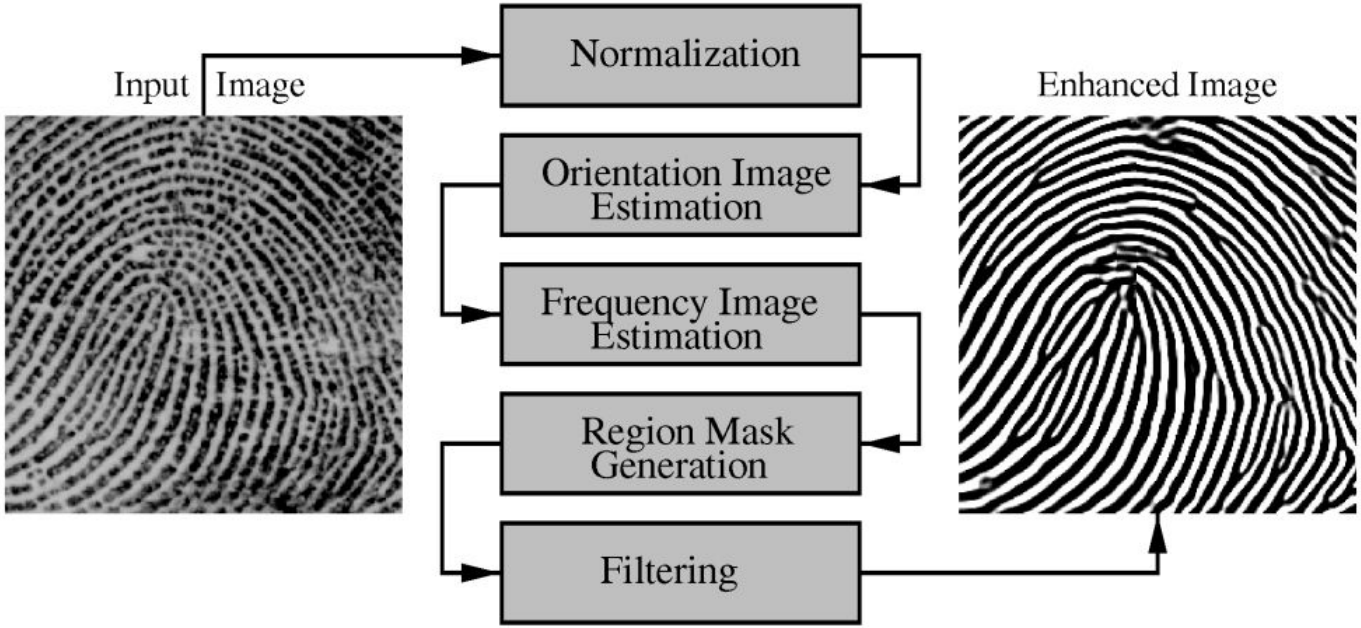
Motivation

- Fingerprints are widely used in biometric authentication Systems, because they are unique!
- **Reliable minutiae extraction** is key to fingerprint matching and requires clean ridge-valley structures.
- Raw fingerprint images often suffer from noise, low contrast, and broken or smudged ridge patterns.
- Standard image enhancement methods (e.g., histogram equalization, median filters) are not tailored to this oriented, periodic texture of fingerprint patterns.
- So we must attend to **local ridge orientation, continuity, tendency … (visual cues)**

Reference: https://biometrics.cse.msu.edu/Publications/Fingerprint/MSU-CPS-97-35fenhance.pdf

Proposed Algorithm

- Goal: Enhance clarity of ridges/valleys to aid minutiae extraction.



- Key Insight: Ridges form **local sinusoidal patterns** with **consistent orientation and frequency**

Reference: https://biometrics.cse.msu.edu/Publications/Fingerprint/MSU-CPS-97-35fenhance.pdf

Dataset

- **Hong et al.**
  - MSU fingerprint database
  - 700 live-scan images (10 per individual)
  - 500 dpi

- **Our work**
  - FVC2002 DB4
  - Size: 10 fingers x 8 impressions
  - Impression type: synthetic plain
  - Format: TIFF
  - 500 dpi, 288x384px

Reference: https://github.com/robertvazan/fingerprint-datasets

Algorithm Step 1: Normalization

- Brings intensity to a standard mean and variance.
- Improve gray-level variation across image regions.
- Doesn't alter ridge-valley structure.

$$G(i,j) = \begin{cases} M_0 + \sqrt{\dfrac{VAR_0\left(I(i,j)-M\right)^2}{VAR}} & \text{if } I(i,j) > M \\ M_0 - \sqrt{\dfrac{VAR_0\left(I(i,j)-M\right)^2}{VAR}} & \text{otherwise} \end{cases}$$

I : Input Image

M : Mean of Input Image

VAR : Variance of Input Image

G : Normalised Image

M_0 : Desired Mean

VAR_0 : Desired Variance

Reference: https://biometrics.cse.msu.edu/Publications/Fingerprint/MSU-CPS-97-35fenhance.pdf

Algorithm Step 1: Normalization

- Results

M = M VAR = V

$M_0 = 100$ $VAR_0 = 100$

Reference: https://biometrics.cse.msu.edu/Publications/Fingerprint/MSU-CPS-97-35fenhance.pdf

Algorithm Step 1: Normalization

- Results

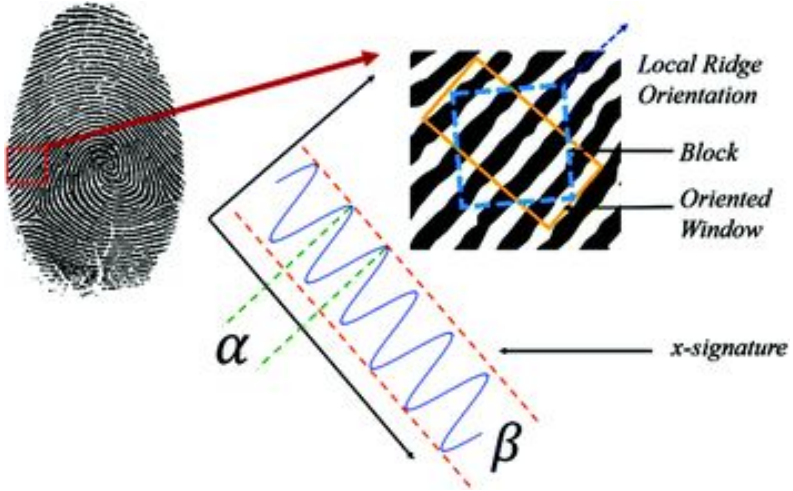
M = 127.5 VAR = 1545

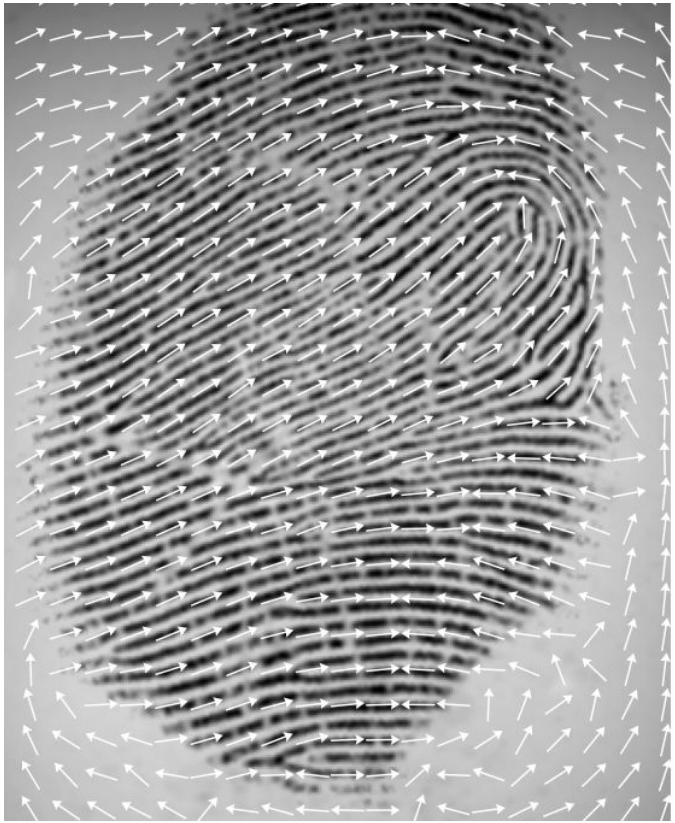
M_0 = 140 VAR_0 = 3000

Reference:

Algorithm Step 2: Orientation Image Estimation

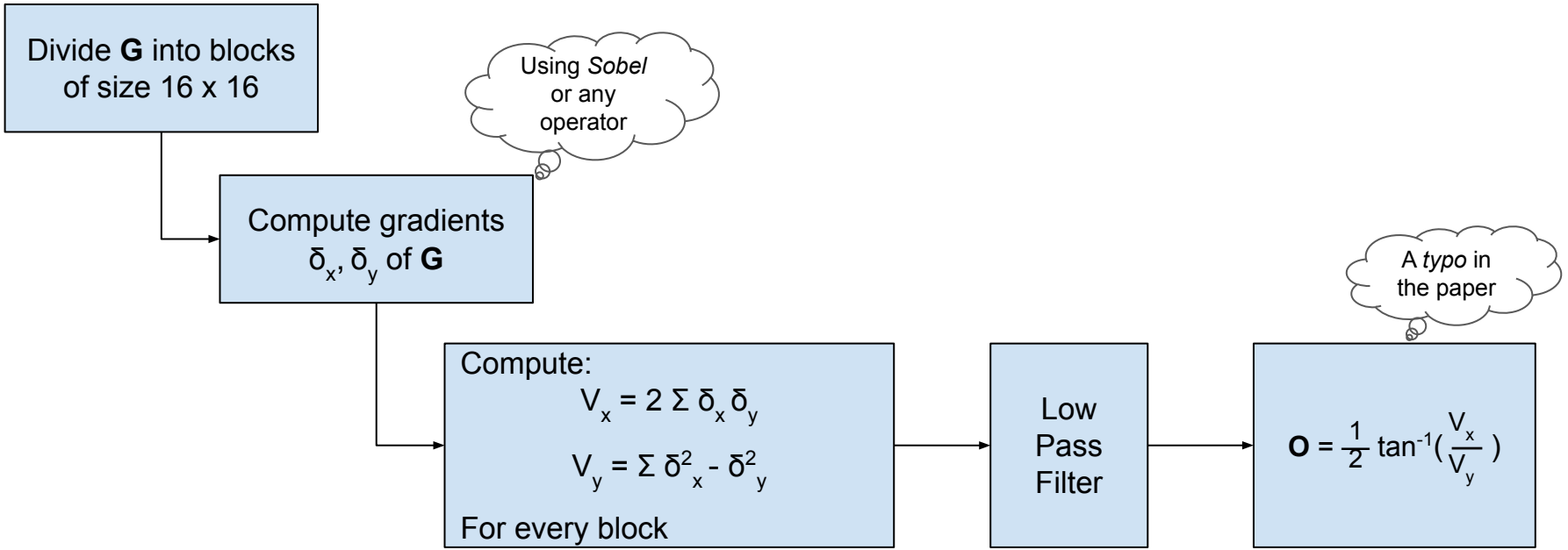
- **Intrinsic Property** of fingerprint images
- Defines **invariant coordinates** for ridge and valleys in local neighbourhood





Algorithm Step 2: Orientation Image Estimation

- **Least squares algorithm on gradients:**
  - Minimize the sum of squared projections of the gradient vectors onto the direction along the ridges



Reference: https://biometrics.cse.msu.edu/Publications/Fingerprint/MSU-CPS-97-35fenhance.pdf

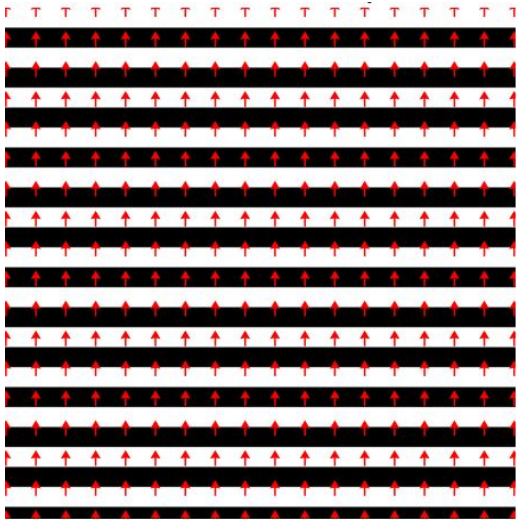
Algorithm Step 2: Orientation Image Estimation

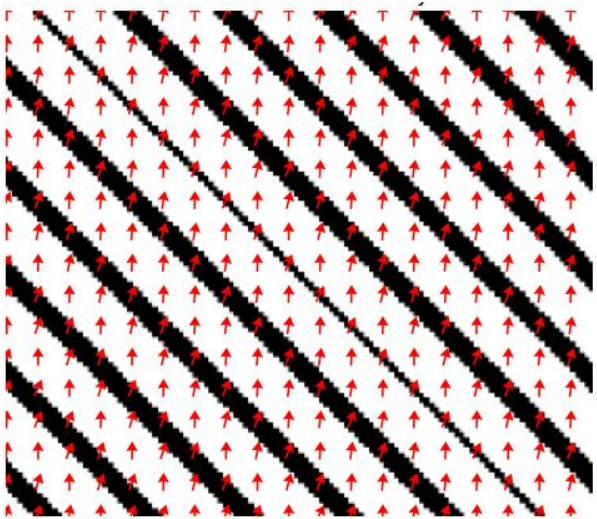
- Initial runs using Sobel

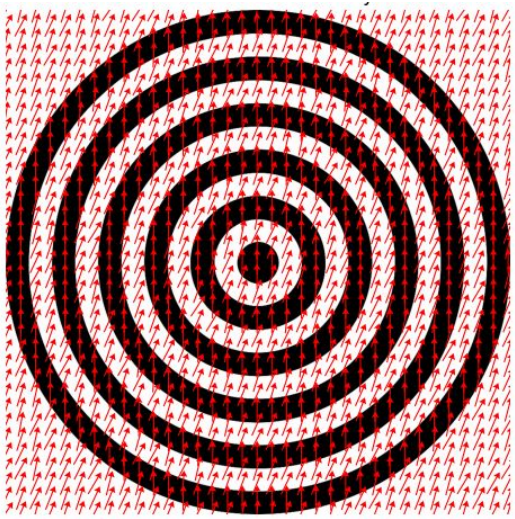
Reference: 102_1.tiff, 102_5.tiff

Algorithm Step 2: Orientation Image Estimation

- What is wrong?
  - Test inputs





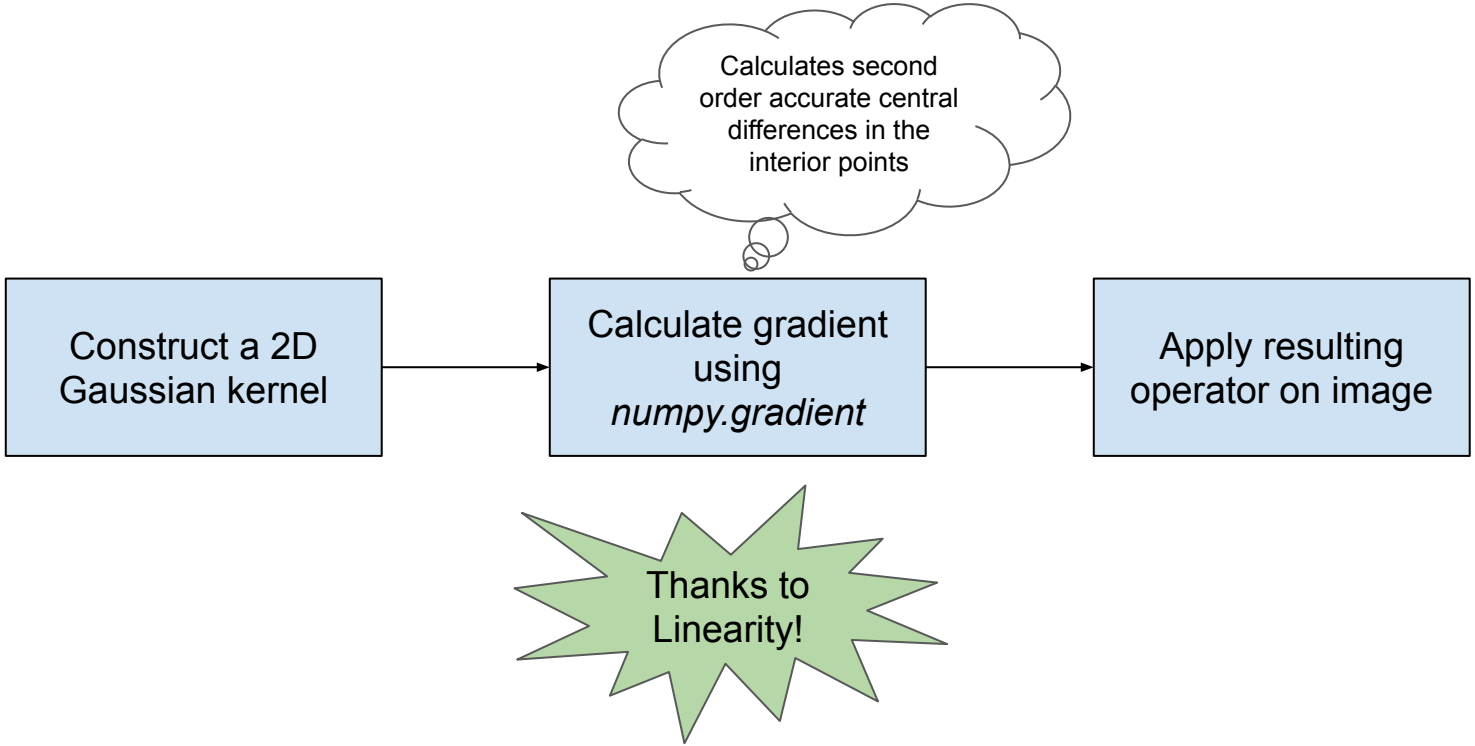


- Issues noted: overflow while computing V_x , V_y

Reference:

Algorithm Step 2: Orientation Image Estimation

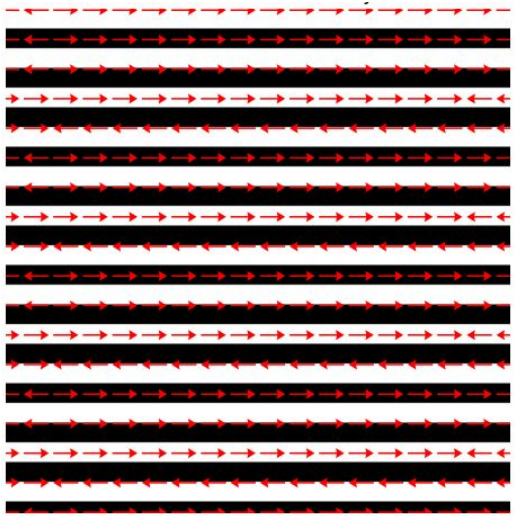
- A fresh approach (after ages of trouble-shooting)

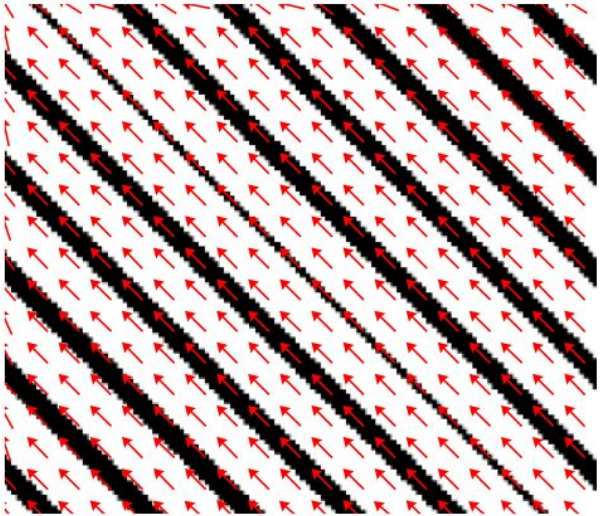


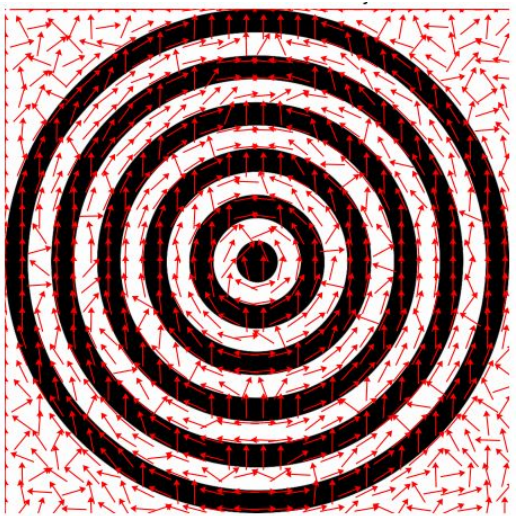
Reference: https://github.com/Utkarsh-Deshmukh

Algorithm Step 2: Orientation Image Estimation

- Results
  - Test inputs



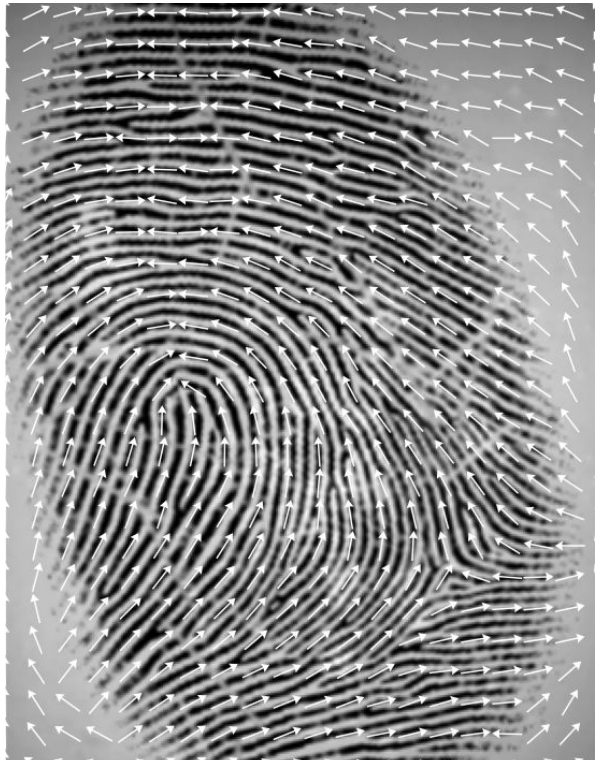




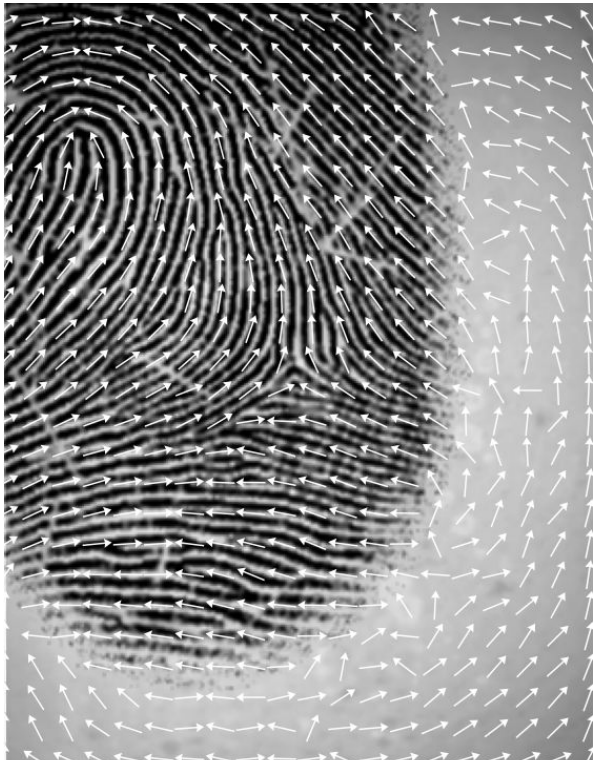
Reference:

Algorithm Step 2: Orientation Image Estimation

- Results



O



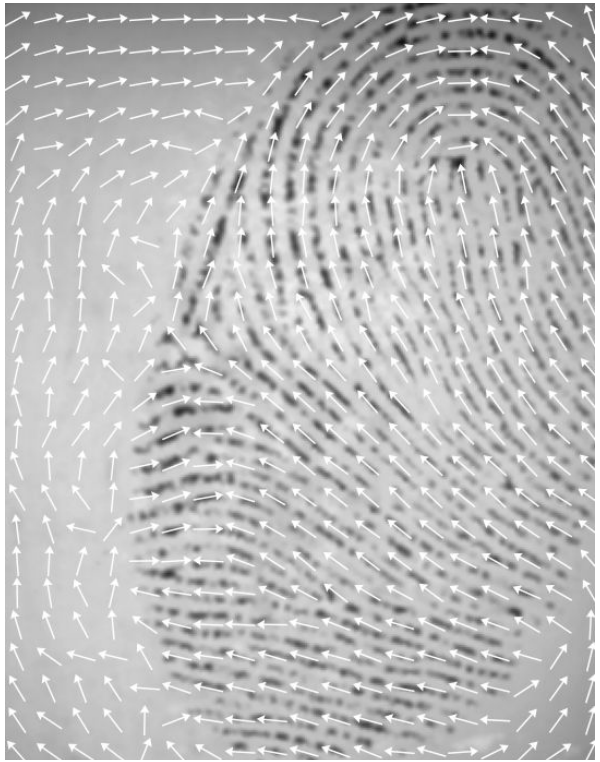
O

Reference: 102_1.tiff , 102_5.tiff

Algorithm Step 2: Orientation Image Estimation

- More

O

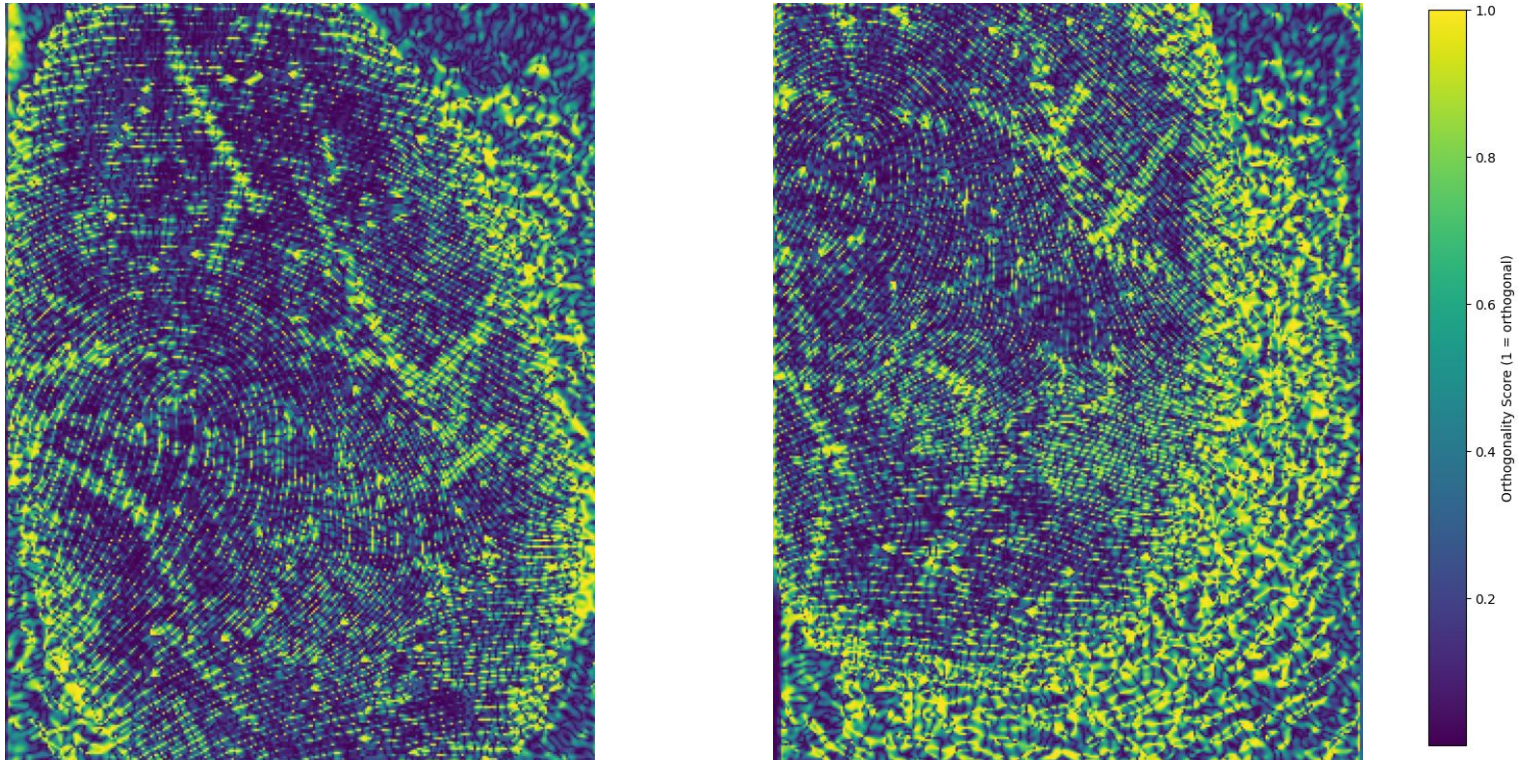


O

Reference:

Algorithm Step 2: Orientation Image Estimation

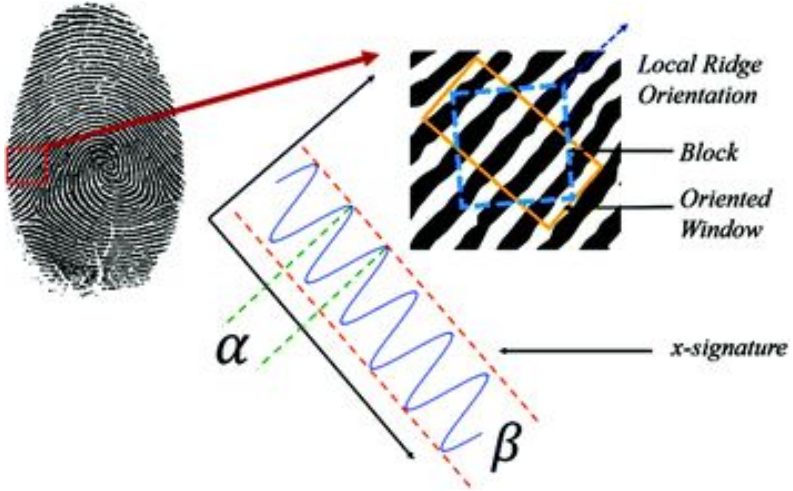
- Ridge-Gradient orthogonality : cosine(angle - π/2)

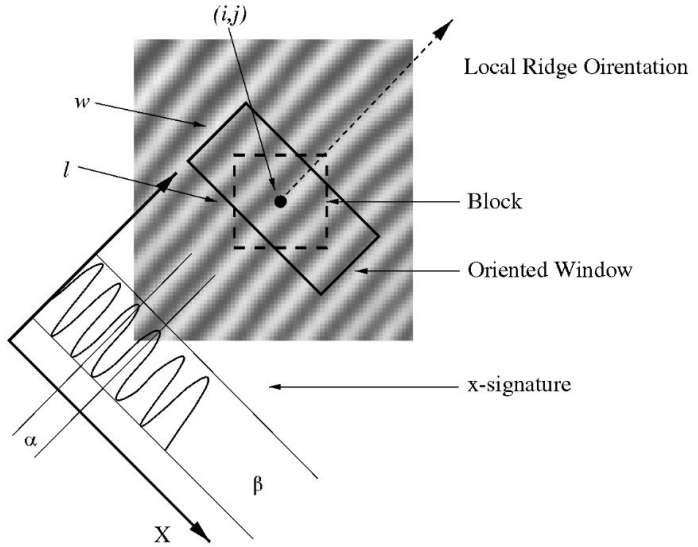


Reference: 102_1.tiff , 102_5.tiff

Algorithm Step 3: Ridge Frequency Image Estimation

- **Intrinsic Property** of fingerprint images
- In local neighbourhood, gray levels along ridges & valleys can be modeled as a **sinusoidal wave**, **normal to ridges**

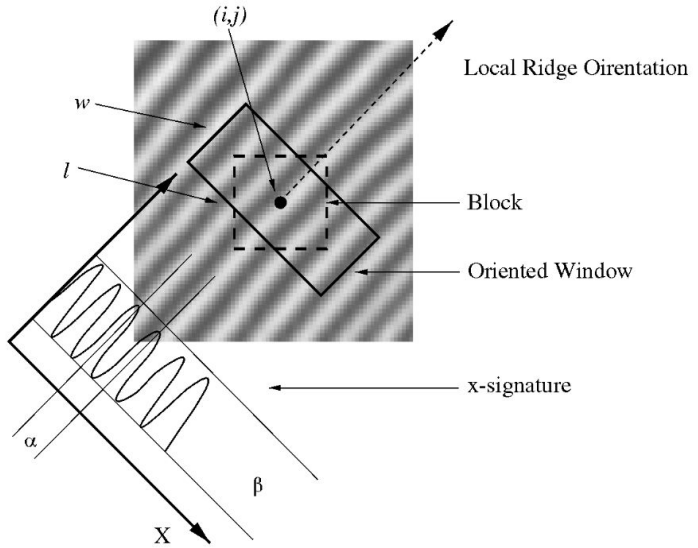




Reference: https://biometrics.cse.msu.edu/Publications/Fingerprint/MSU-CPS-97-35fenhance.pdf

Algorithm Step 3: Ridge Frequency Image Estimation

- **X-Signature mimics the ridge-valley wave pattern**



Divide **G** into blocks of size 16 x 16

For each block compute an **orientation window** in the ridge coordinate frame

A fancy formula

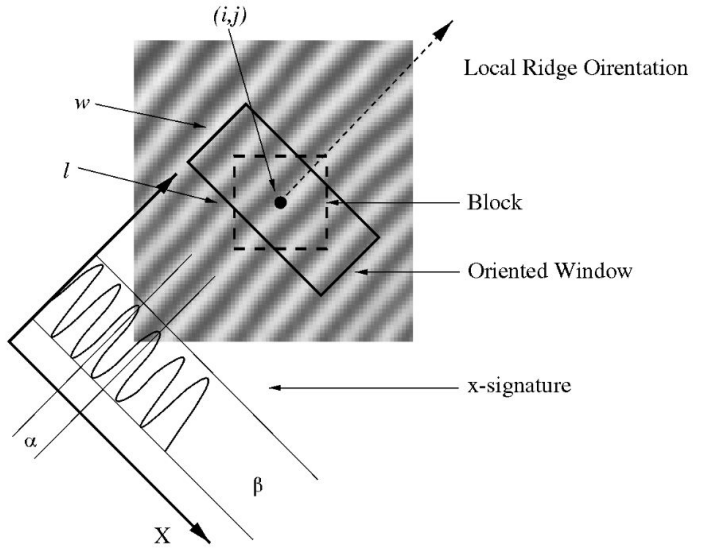
For each block compute the **X signature** within oriented window

Compute frequency of X-signature as **(average distance between consecutive peaks)$^{-1}$**

Reference: https://biometrics.cse.msu.edu/Publications/Fingerprint/MSU-CPS-97-35fenhance.pdf

Algorithm Step 3: Ridge Frequency Image Estimation

- **X-Signature mimics the ridge-valley wave pattern**



Compute frequency of X-signature as **(average distance between consecutive peaks)$^{-1}$**

If no peaks detected
OR
estimated frequency outside (**0.04,0.333**):
Assign *invalid* value = -1

Iteratively **interpolate** for all *invalid* blocks using neighbouring values

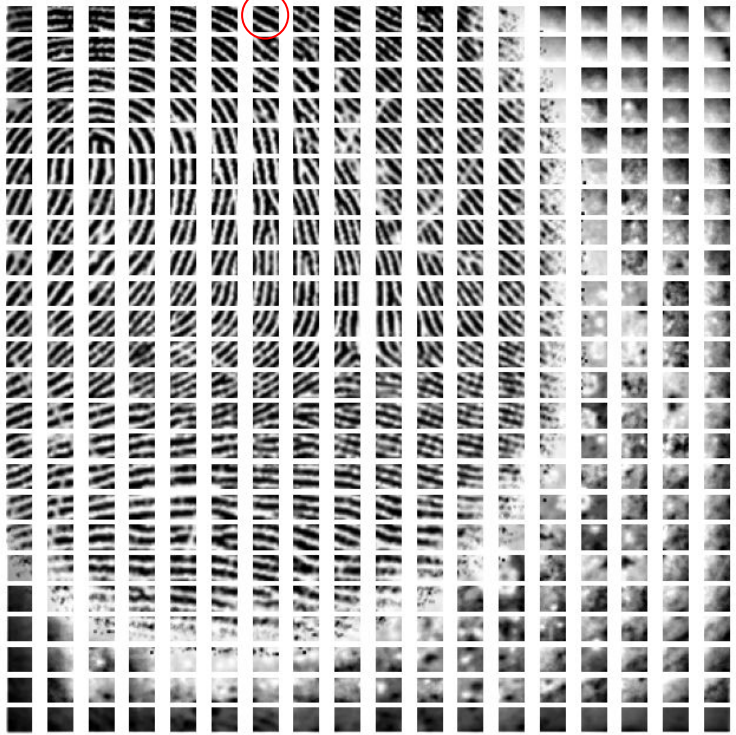
Gaussian Smoothing

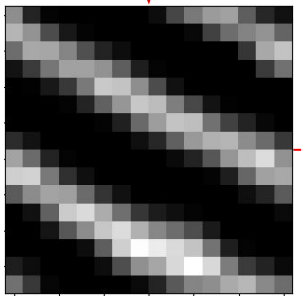
Frequency Image **F**

Reference: https://biometrics.cse.msu.edu/Publications/Fingerprint/MSU-CPS-97-35fenhance.pdf

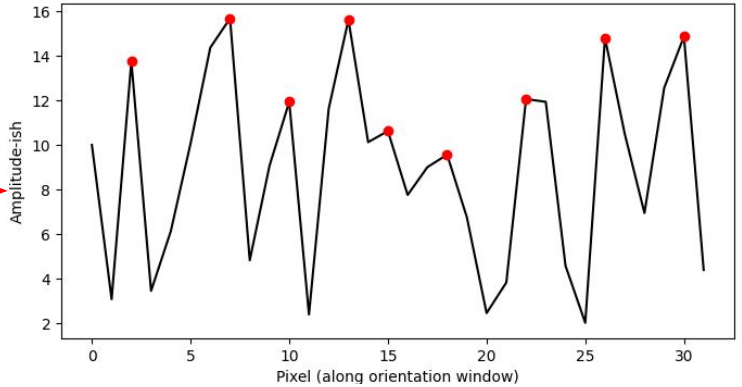
Algorithm Step 2: Orientation Image Estimation

- Blocks (16 x 16)





Block

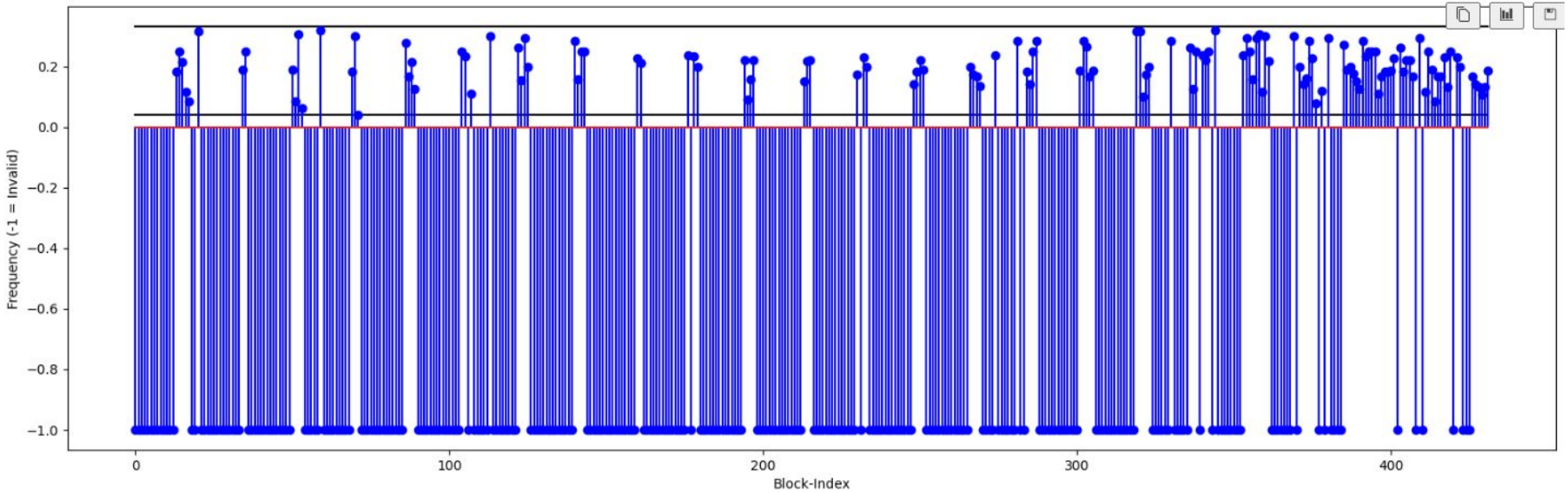


X-Signature

Reference: 102_5.tiff

Algorithm Step 2: Orientation Image Estimation

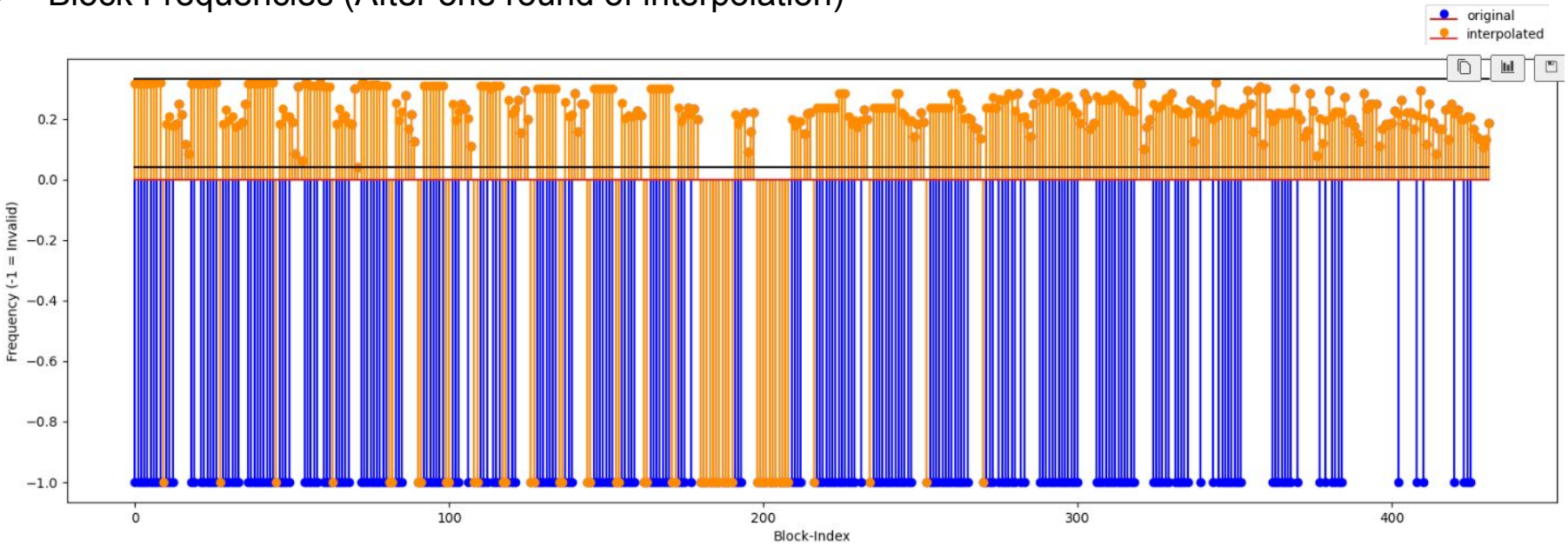
- Block Frequencies (before interpolation)



Reference:

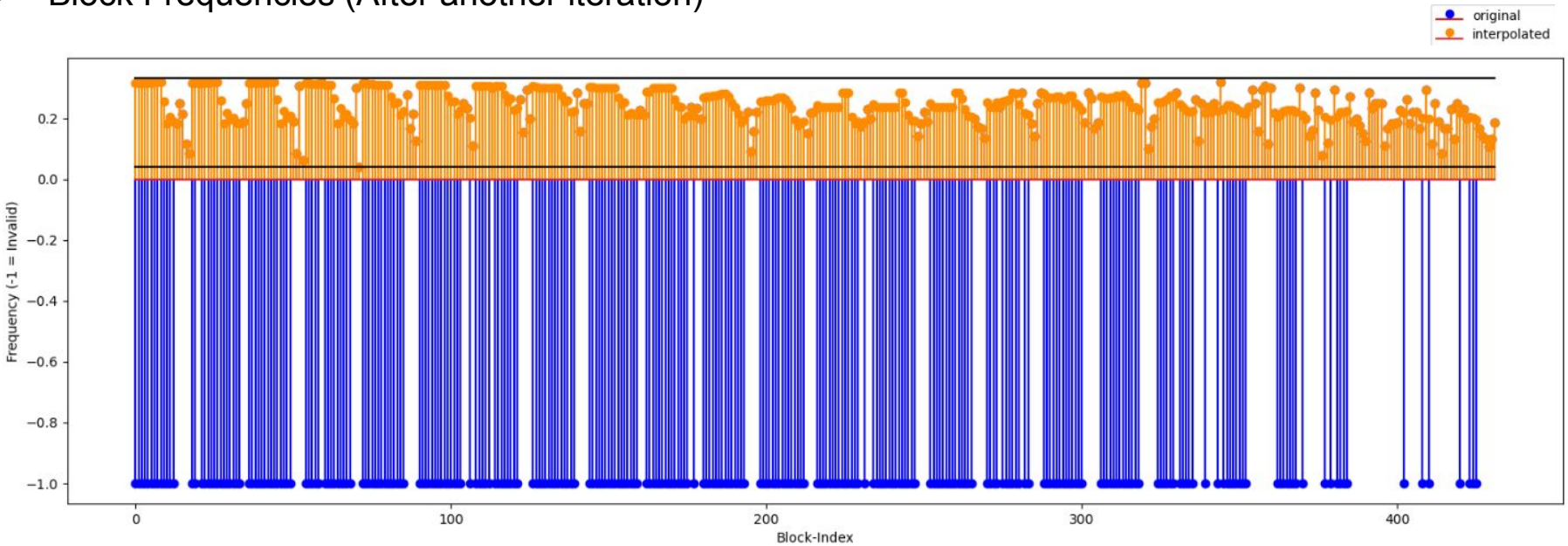
Algorithm Step 2: Orientation Image Estimation

- Block Frequencies (After one round of interpolation)



Algorithm Step 2: Orientation Image Estimation

- Block Frequencies (After another iteration)

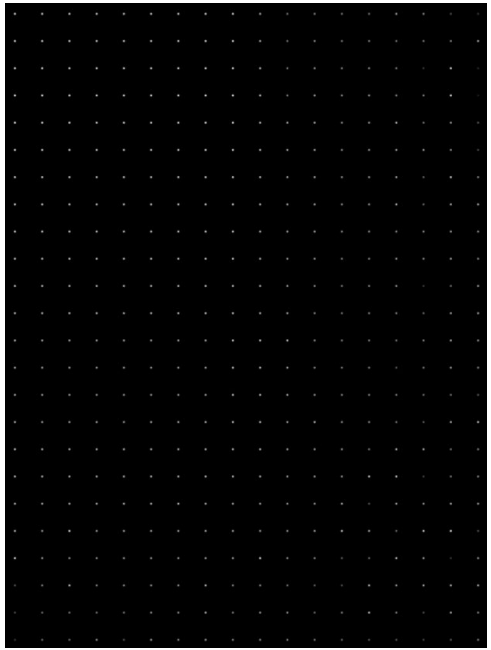


Reference:

Algorithm Step 2: Orientation Image Estimation

- Results (Frequency Image **F**)

I



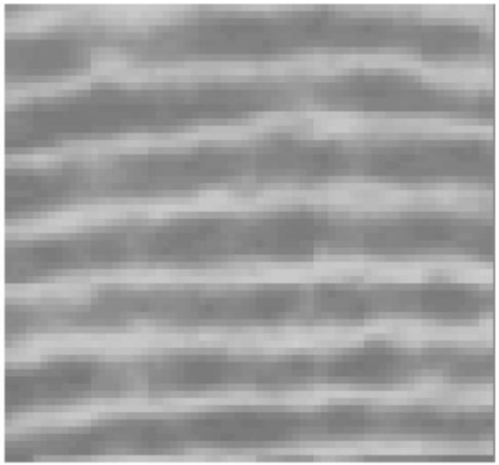
After interpolation

F

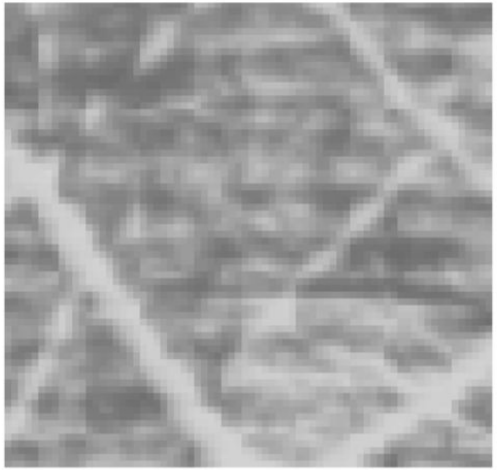
Reference: 102_5.tiff

Algorithm Step 4: Region Masking

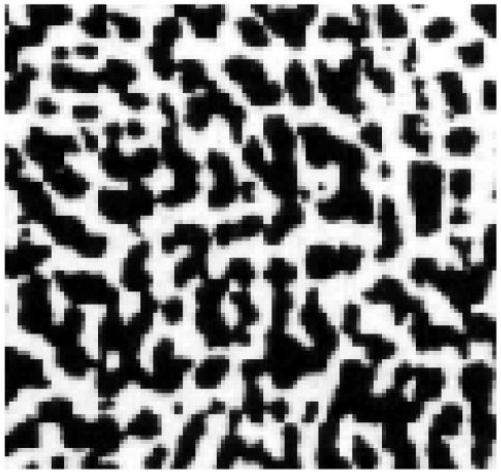
- A filter to **discard beyond-repair** images.
- Use local wave attributes to classify regions as **recoverable** and **non-recoverable**



Well-Defined

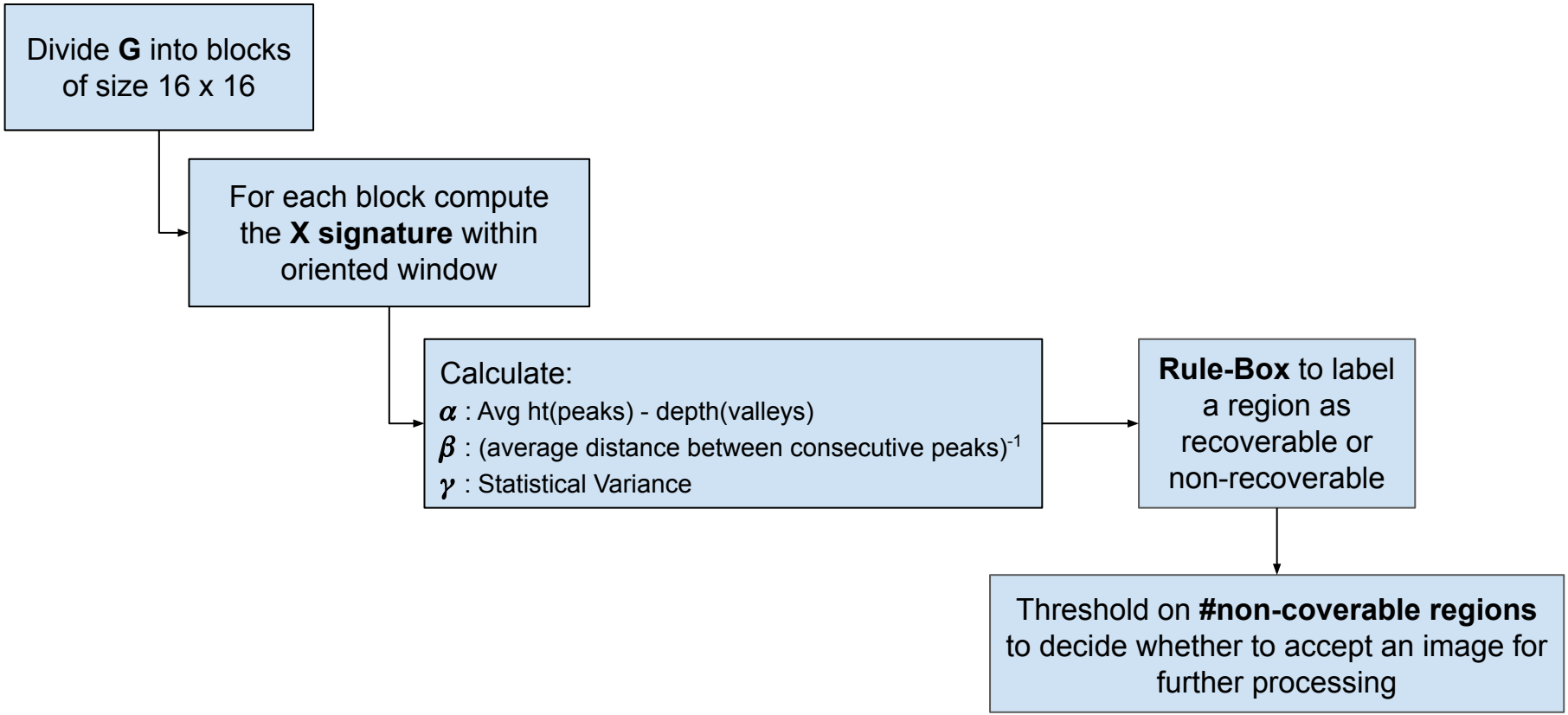


Recoverable



Non-recoverable

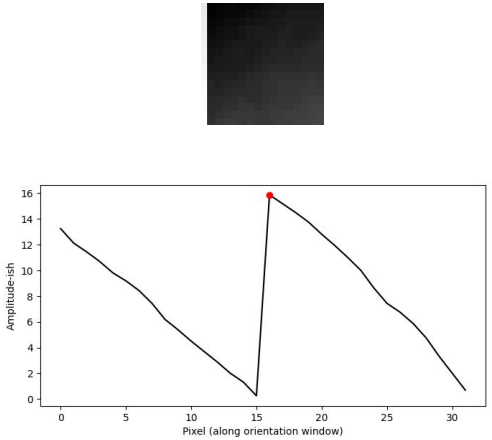
Algorithm Step 4: Region Masking



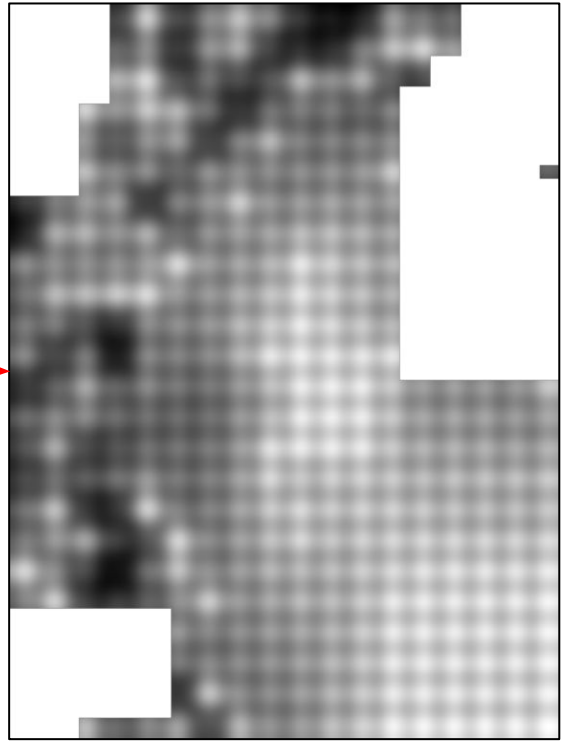
Reference:

Algorithm Step 4: Region Masking

- Irrecoverable regions

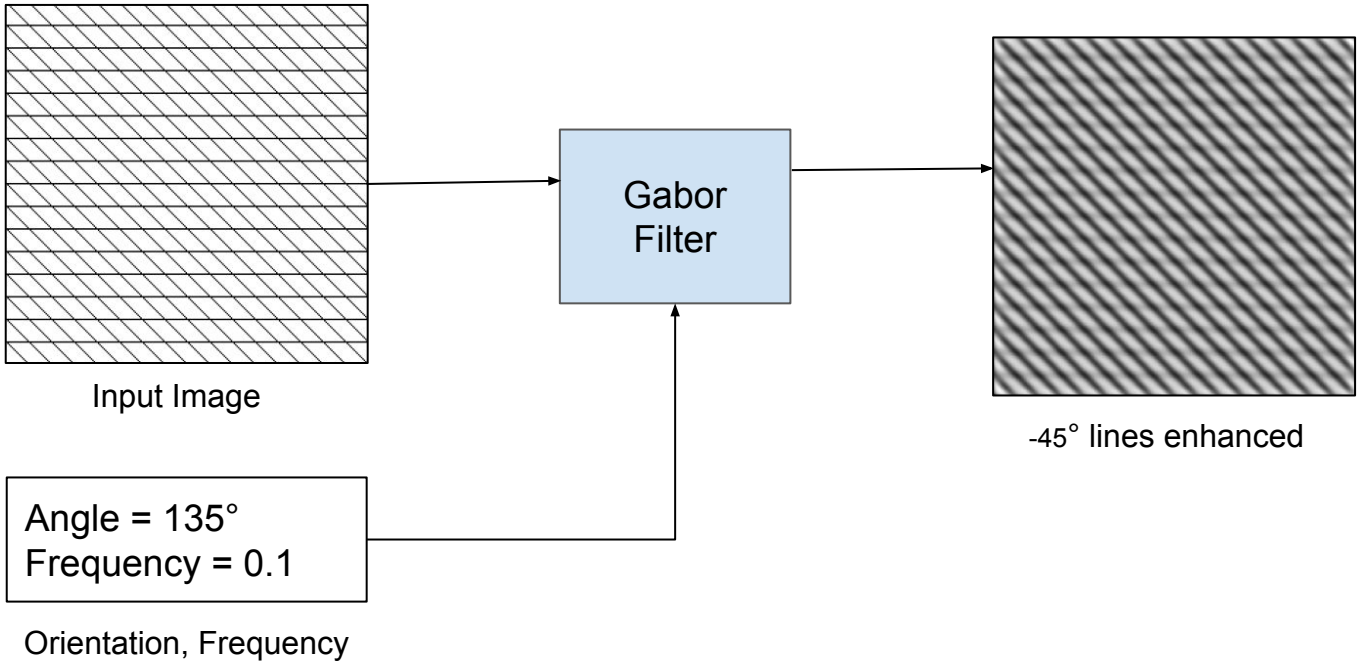


Interpolation doesn't converge hence those white blocks



Algorithm Step 5: (Gabor) Filtering

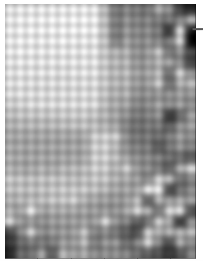
- **Frequency** and **orientation** selective **band-pass filters**
- Exploit the local frequency and orientation to **enhance true ridge-valleys** and **suppress noise**



Algorithm Step 5: (Gabor) Filtering

Normalized Image
G

Orientation Image
O



Frequency Image
F

Gabor
Filter

Enhanced Image
E

Reference: 102_5.tiff

Algorithm Step 5: (Gabor) Filtering

- Results

E

I

Reference: 102_5.tiff

Algorithm Step 5: (Gabor) Filtering

- Results

E

I

Algorithm Step 5: (Gabor) Filtering

- Results

E

I

Algorithm Step 5: (Gabor) Filtering

- Results

E

I

Discussion

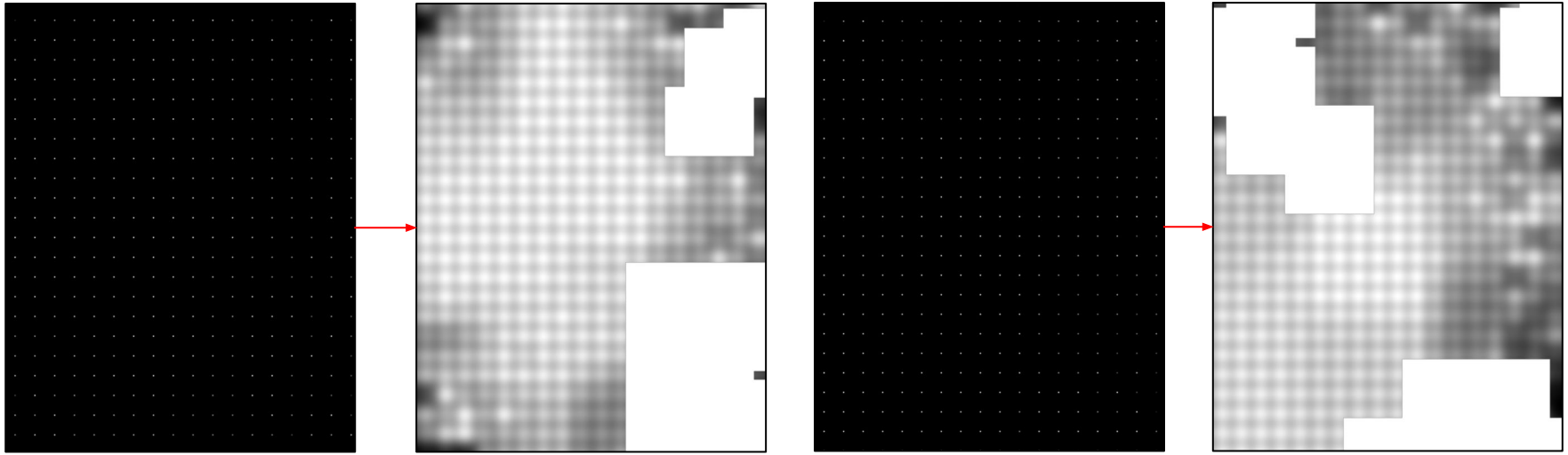
- When successful enhancement is not **visually evident**
- In many cases, **white blocks mask** the (gabor) filtered **output**




Reference: 102_6.tiff,102_7.tiff, 105_1.tiff, 108_8.tiff

Discussion

- These masks grew because of **Gaussian smoothing** on the frequency image

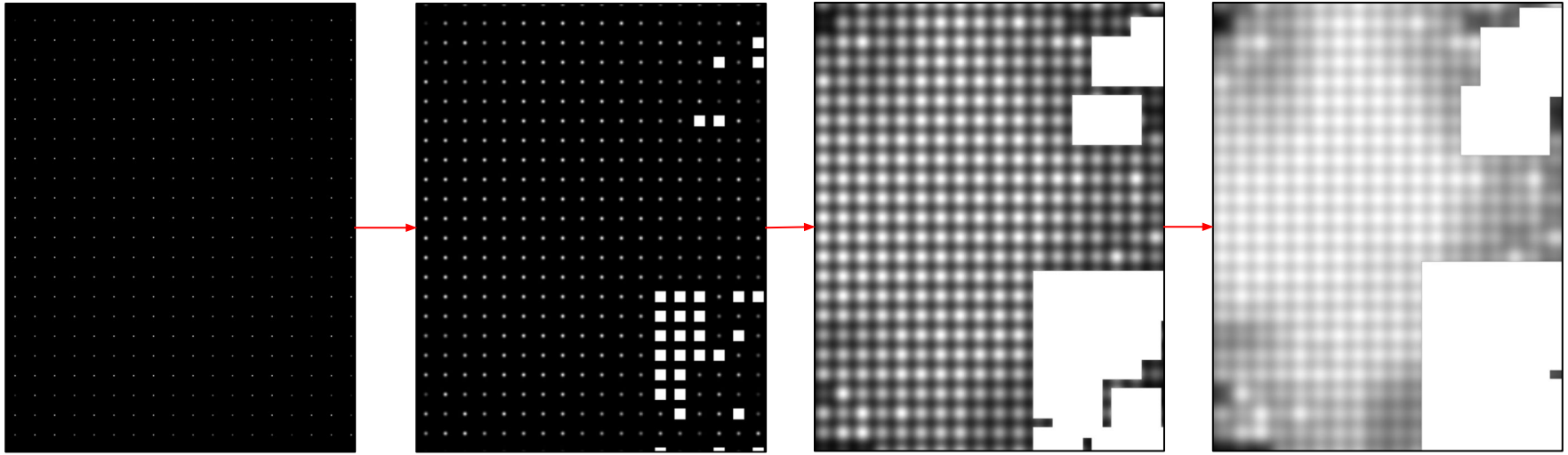


Interpolation result | Smoothened | Interpolation result | Smoothened

Reference: 102_6.tiff,102_7.tiff

Discussion

- These masks grew because of **Gaussian smoothing** on the frequency image

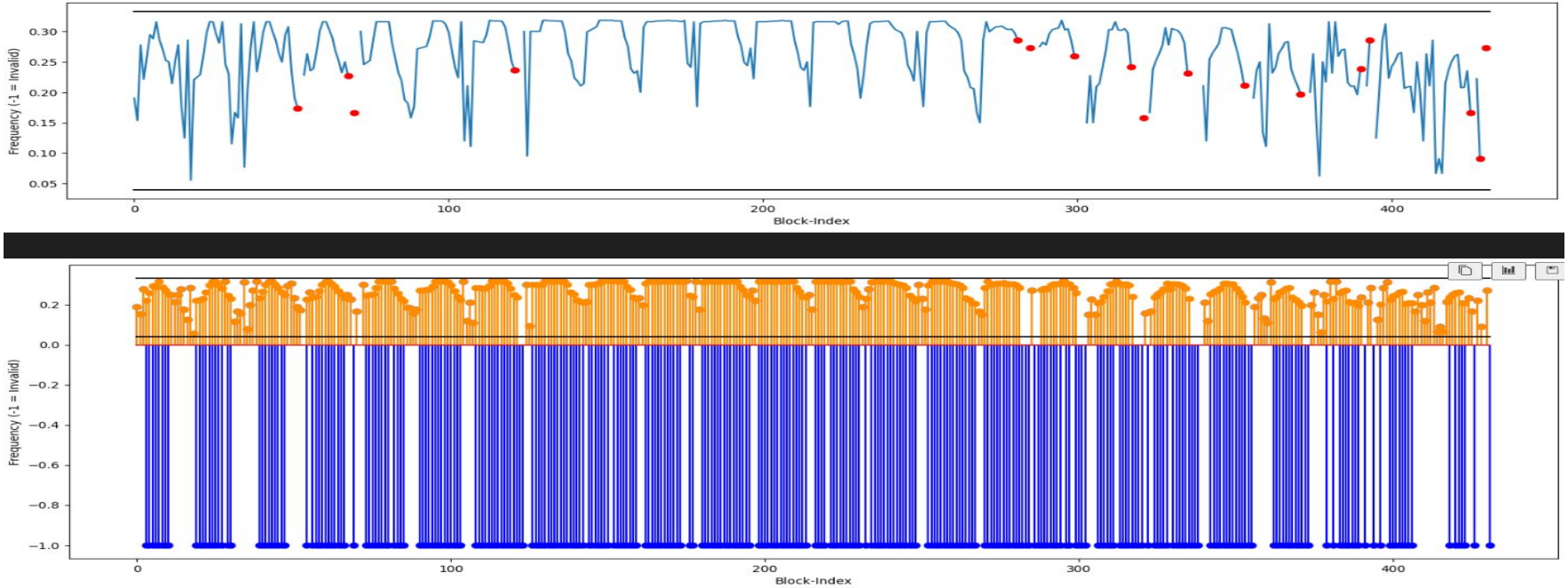


Interpolation result | σ = 1 | σ = 5 | σ = 7

Reference: 102_6.tiff

Discussion

- These masks emerge because the corresponding **block-frequency** was **NaN**
- Interpolation **did not converge**

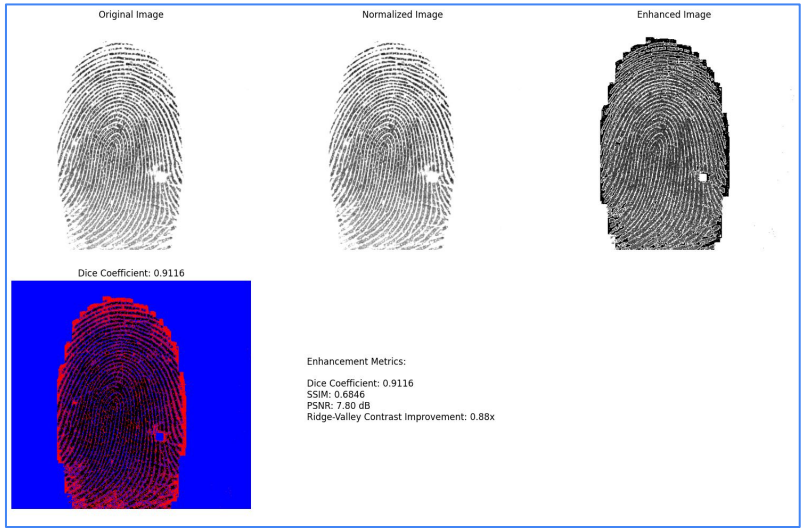


Reference: 102_6.tiff

Performance Evaluation

- **Dice Coefficient**

What is it: Measures how similar two binary images are to each other. Ranges from 0 to 1, where 1 implies the images are identical and 0 implies they have nothing in common.

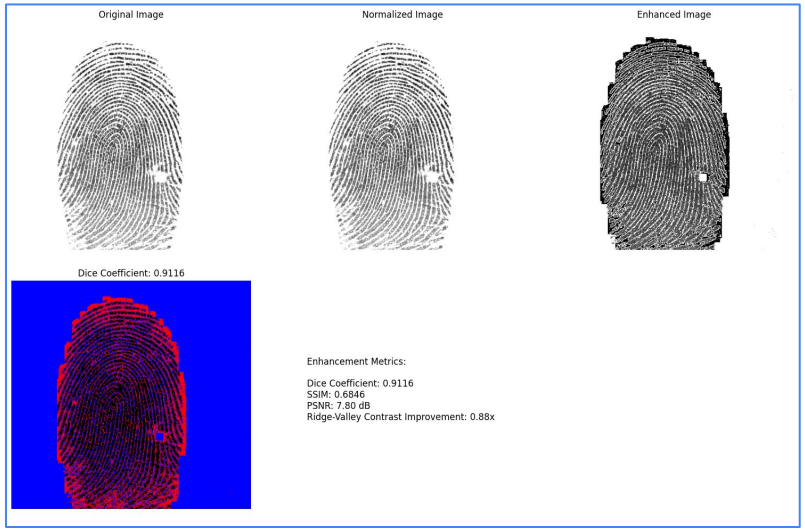


Reference:

Performance Evaluation

- **SSIM (Structural Similarity Index Measure)**

  What is it: Evaluates the similarity between two images based on luminance, contrast, and structure. It also ranges from -1 to 1, where 1 indicates perfect similarity.

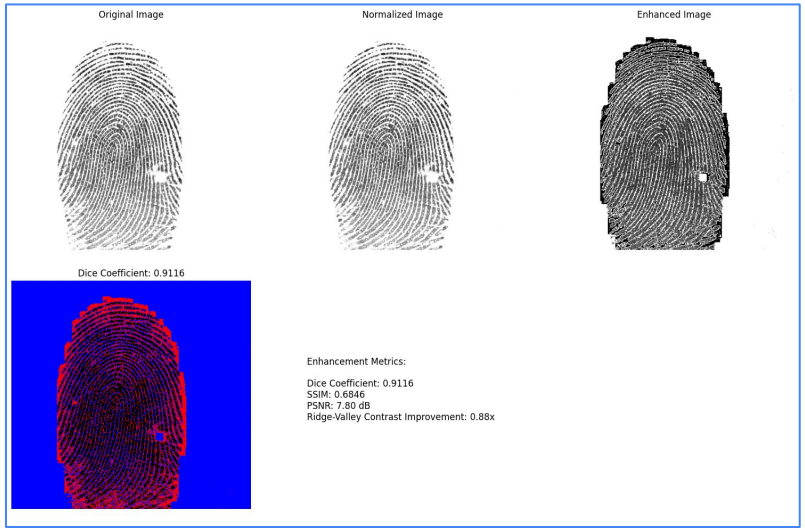


Reference:

Performance Evaluation

- **PSNR (Peak Signal to Noise Ratio)**

  What is it: Measures the ratio between the maximum possible power of a signal and the power of corrupting noise. Higher the better.

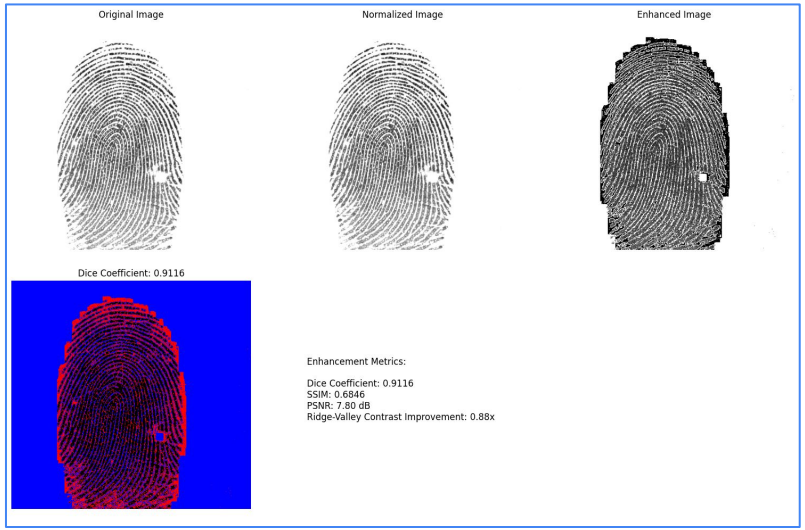


Reference:

Performance Evaluation

- **Ridge Valley Contrast Improvement**

  What is it: Measures how well our enhancement improves the difference in brightness between ridges (the dark lines) and valleys (the light spaces between).



Future Directions

- **Fix the frequency interpolation convergence**

- **Quantitative assessment of developed pipeline**
    - Goodness Index (used by Hong et al.)
    - DICE Score
    - PSNR
    - SSIM
    - Ridge Valley Contrast Improvement

Future Directions

- **Deep Learning-Based Enhancement**
  - Implement **U-Net** architectures tailored for fingerprint enhancement.
  - Other **CNN** models that can handle extremely low-quality fingerprints
  - Explore **GANs** (Generative Adversarial Networks) for fingerprint restoration.

- **Adaptive Enhancement**
  - Develop algorithms that **dynamically adjust** parameters based on local fingerprint quality
  - Implement quality-based **region-adaptive enhancement** (different techniques for different regions)
  - New methods for handling latent (partial, smudged) fingerprints found at crime scenes

Future Directions

- **Advanced Minutiae Detection**
  - Implement **Crossing Number (CN)** method for minutiae extraction
  - Add **ridge counting** between minutiae for richer feature sets
  - Develop algorithms for detecting **more complex minutiae types** beyond bifurcations and endings

- **Minutiae Quality Assessment**
  - Develop **confidence scores** for each detected minutia
  - **Filter false minutiae** based on structural and contextual rules
  - Implement **ridge flow-based validation** to remove spurious minutiae

Any Questions …