NFIS2 Software

Software

- Image Group of the National Institute of Standards and Technology (NIST)
- NIST Fingerprint Image Software Version 2 (NFIS2)
- Developed for Federal Bureau of Investigation (FBI) and Department of Homeland Security (DHS)
- Aims to facilitate and support the automated manipulation and processing of fingerprint images.

NIST Fingerprint Image Software Version 2 (NFIS2)

- NFIS2 contains 7 general categories.
- We investigate 4 out of 7: PCASYS, MINDTCT, NFIQ and BOZORTH3.
- **PCASYS** is a neural-network based fingerprint classification system, which categorized a fingerprint image into the class of arch, left or right loop, scar, tented arch, or whorl.
- PCASYS is the only known no cost system of its kind.

NIST Fingerprint Image Software Version 2 (NFIS2)

- MINDTCT is a minutiae detector that automatically locates and records ridge ending and bifurcations in a fingerprint image.
- MINDTCT includes minutiae quality assessment based on local image conditions.
- The FBI's Universal Latent Workstation uses MINDTCT, and it too is the only known no cost system of its kind.

NIST Fingerprint Image Software Version 2 (NFIS2)

- NFIQ is a fingerprint image quality algorithm that analyses a fingerprint image and assigns a quality value of 1 (highest quality) – 5 (lowest quality) to the image.
- Higher quality images produce significantly better performance with matching algorithm.

NIST Fingerprint Image Software Version 2 (NFIS2)

- BOZORTH3 is a minutiae based fingerprint matching algorithm that will do both one-to-one and one-to-many matching operations.
- BOZORTH3 matching algorithm computes a match score between the minutiae from any two fingerprints to help determine if they are from the same finger.
- BOZORTH3 accepts minutiae generated by the MINDTCT algorithm.
- Written by Allan S. Bozorth while at the FBI.

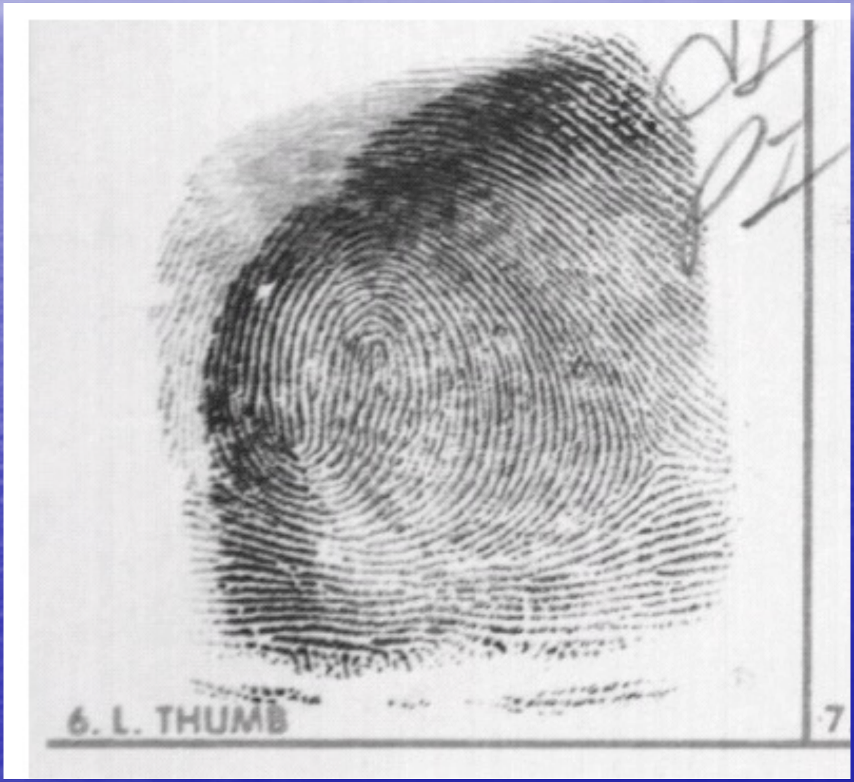
Fingerprint Classification (PCASYS)

- PCASYS is a prototype/demonstration pattern-level fingerprint classification program.
- It is provided in the form of a source code distribution and is intended to run on a desktop workstation.
- The program reads and classifies each of a set of fingerprint image files, optionally displaying the results of several processing stages in graphical form.
- This distribution contains 2700 fingerprint images that may be used to demonstrate the classifier; it can also be run on user-provided images.

Fingerprint Classification (PCASYS)

- The basic method used by the PCASYS fingerprint classifier consists of,
  - First, extracting from the fingerprint to be classified an array (a two-dimensional grid in this case) of the local orientations of the fingerprint's ridges and valleys.
  - Second, comparing that orientation array with similar arrays made from prototype fingerprints ahead of time.
- Refer to nbis_non_export_control.pdf for details

Fingerprint Classification (PCASYS)



Fingerprint used to demonstrate the fingerprint classification process (s0025236.wsq).

PCASYS -- Segmentor

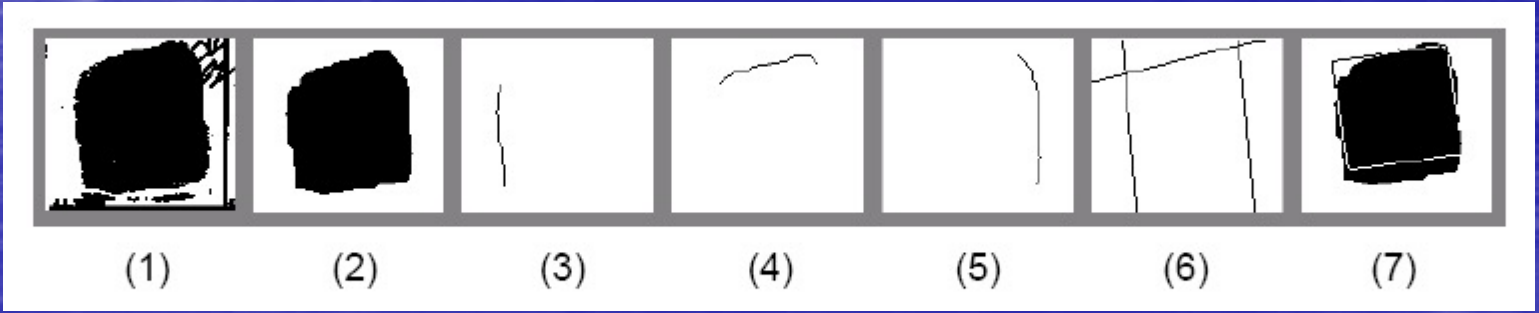
- **Segmentor** input an 8-bit grayscale raster of width at least 512 pixels and height at least 480 pixels (these dimensions, and scanned at about 19.69 pixels per millimeter (500 pixels per inch).
- The segmentor produces, as its output, an image that is 512·480 pixels in size by cutting a rectangular region of these dimensions out of the input image.

PCASYS -- Segmentor

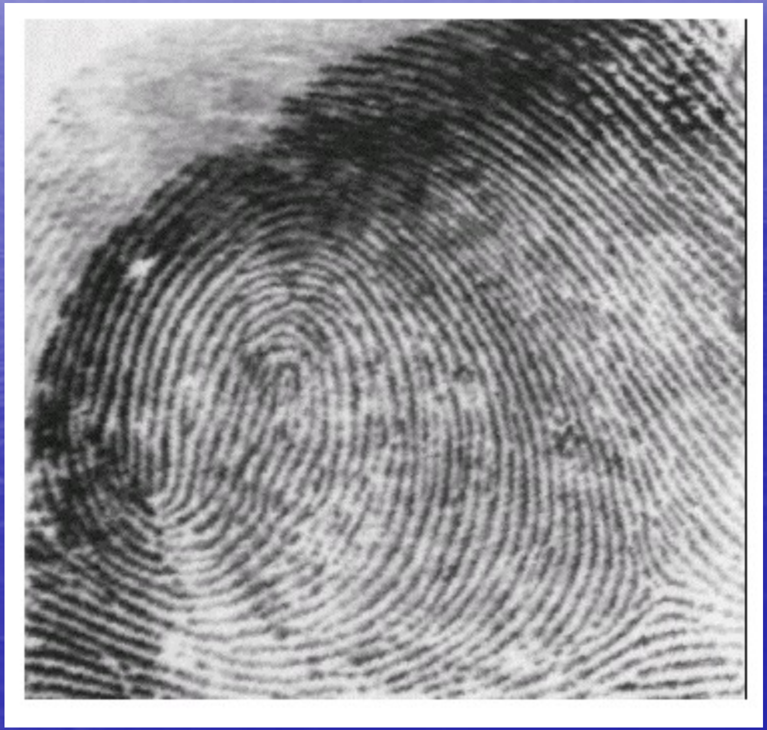
1. the segmentor produces a small binary (two-valued or logical-valued) image, and the part of the image that contains ink is called **foreground**.
2. the routine performs some **cleanup** work on the foreground-image, the main purpose of which is to delete those parts of the foreground that correspond to printing or writing rather than the finger impression.

3-6. the routine uses the edges to calculate the overall slope of the foreground and fits a straight line to each edge by linear regression.

7. last frame in Figure 7 is the (cleaned-up) foreground with an outline superimposed on it showing where the segmentor has decided to cut.



PCASYS -- Segmentor

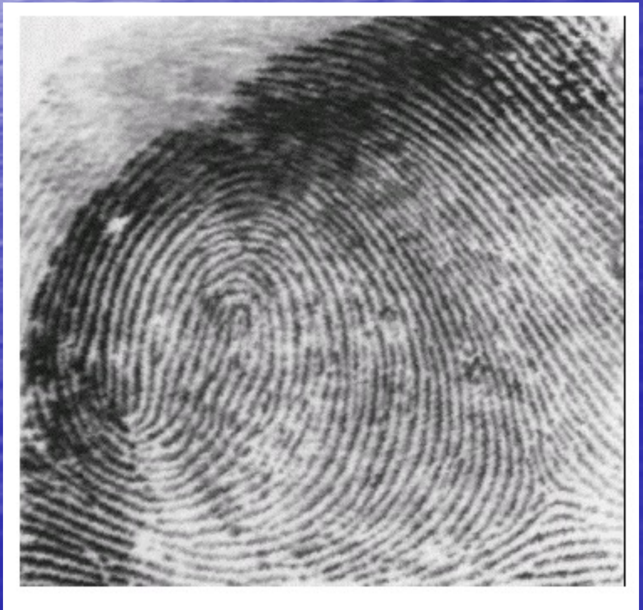

resulting segmented image

PCASYS - Image Enhancement

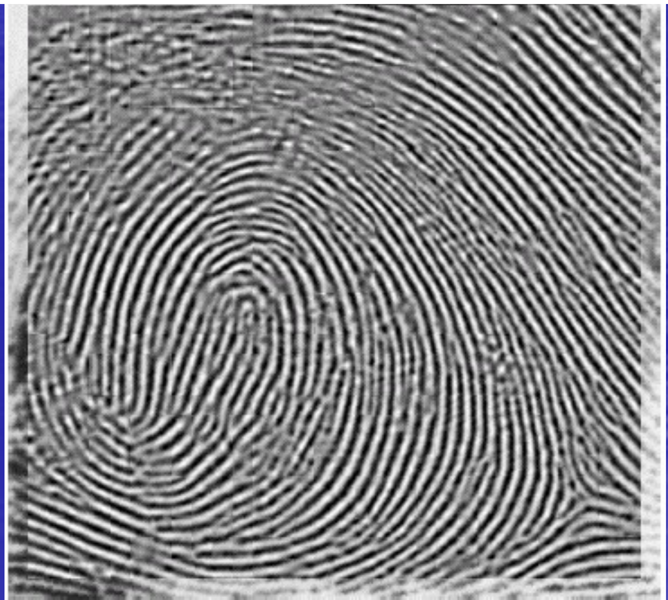
- Perform the forward two-dimensional Fast Fourier transform (FFT) to convert the data from its original (spatial) representation to a frequency representation.
- The backward 2-d FFT is done to return the enhanced data to a spatial representation before snipping out the middle 16×16 pixels and installing them into the output image.

PCASYS - Image Enhancement

- Noticeable difference seen between the original and enhanced versions is the increase in contrast.
- The more important change caused by the enhancer is the improved smoothness and stronger ridge/valley structure of the image.
- Discontinuities are visible at the boundaries of some output squares have no major harmful effect on subsequent processing.



Sample fingerprint after enhancement

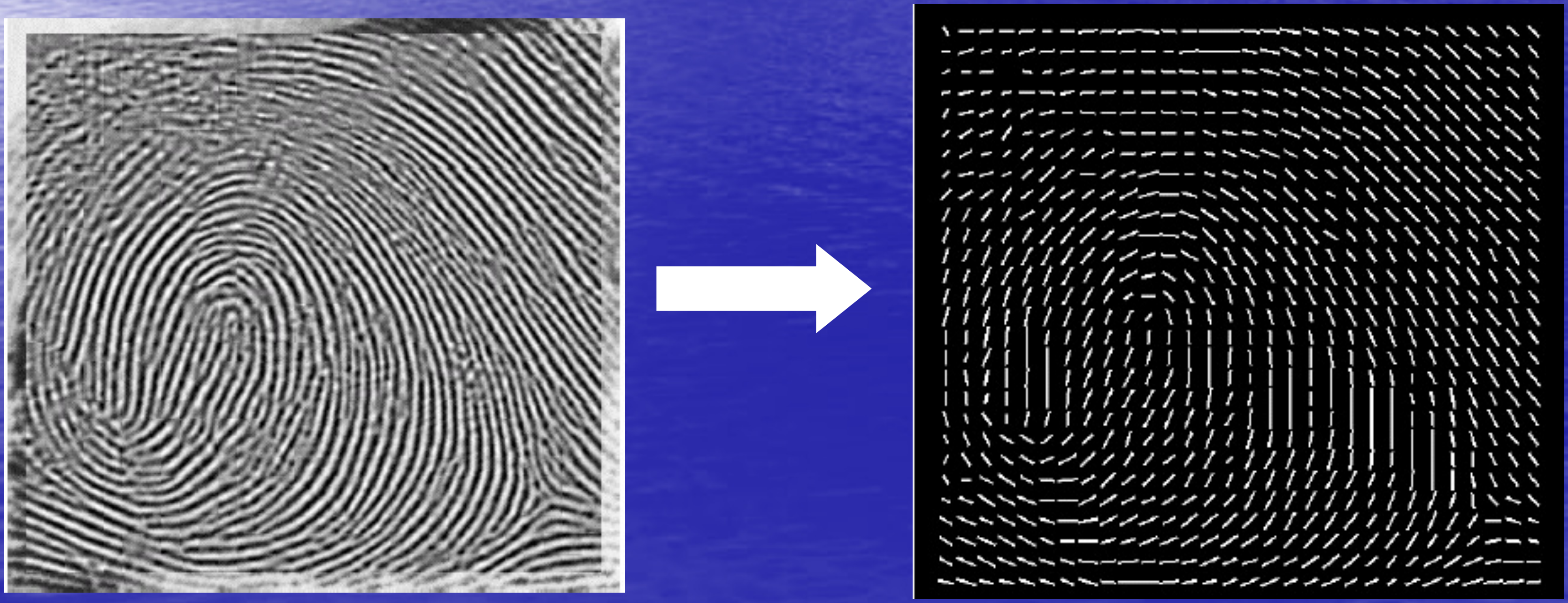


PCAYSIS - Ridge-Valley Orientation Detector

- This step detects the local orientation of the ridges and valleys of the finger surface, and produces an array of regional averages of these orientations.

PCAYSIS - Ridge-Valley Orientation Detector

- depicts the local average orientations that were detected in the segmented and filtered image from the example fingerprint.
- Array of local average orientations of the example fingerprint. Each bar, depicting an orientation, is approximately parallel to the local ridges and valleys.

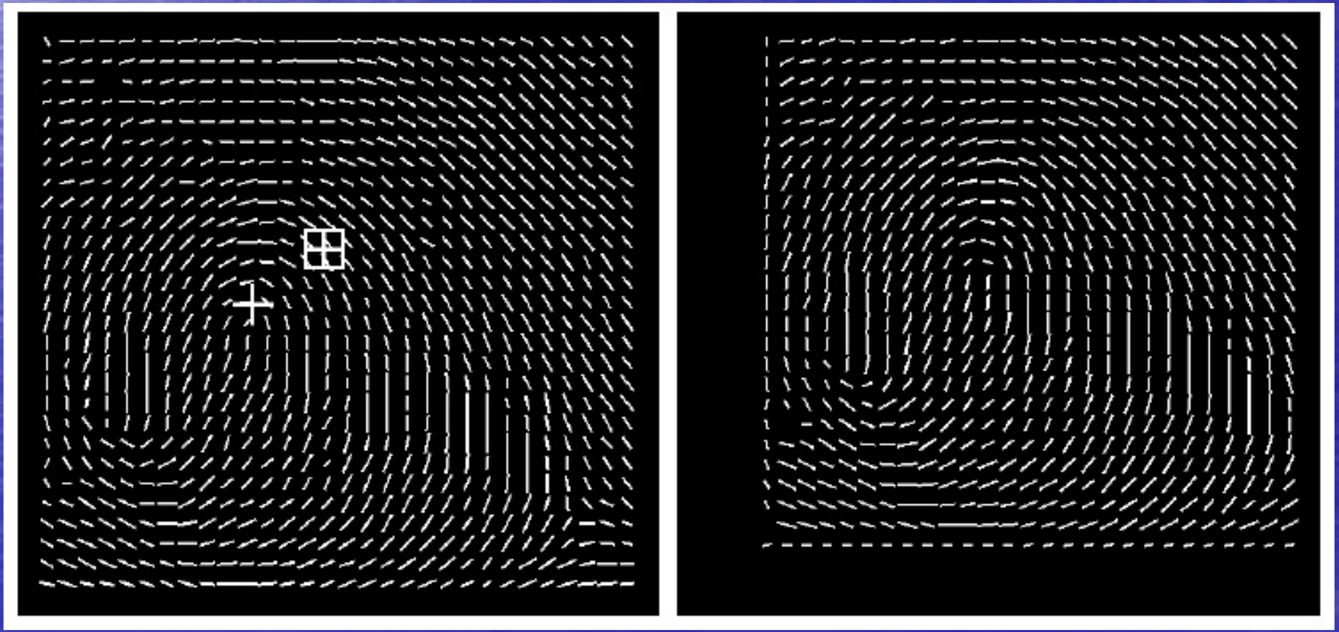


PCASYS -- Registration

- Registration is a process that the classifier uses in order to reduce the amount of translation variation between similar orientation arrays.
- registration improves subsequent classification accuracy.

PCASYS -- Registration

- Left: Orientation array Right: Registered orientation array. The plus sign is registration point (core) found by R92 (an algorithm), and plus sign in a square is standard (median) registration point.

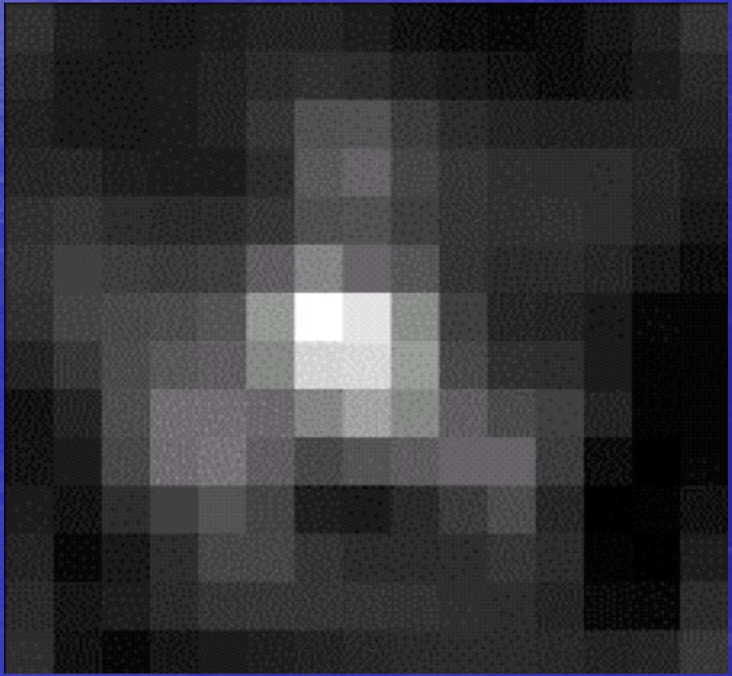


PCASYS -- Regional Weights

- In order to allow the important central region of the fingerprint to have more weight than the outer regions; what we call *regional weights*.
- Involving the application of linear transforms prior to PNN distance computations, we obtained the best results by using regional weights.

PCASYS -- Regional Weights

- Absolute values of the optimized regional weights. Each square represents one weight, associated with a 2×2 block from the registered orientation array.



PCASYS --Probabilistic Neural Network Classifier

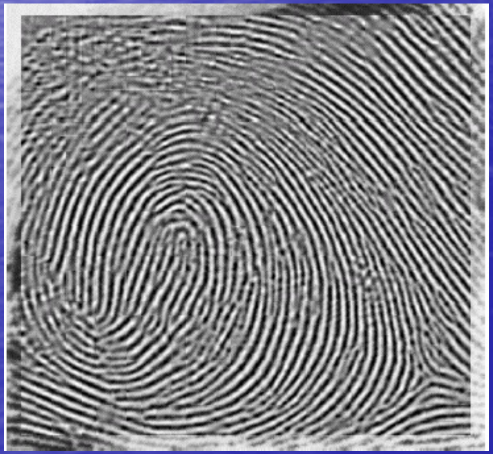
- Input the low-dimensional feature vector that is the output of the transform (PCA) and determine the class of the fingerprint.
- Probabilistic Neural Network (PNN) classifies an incoming feature vector by computing the value of spherical Gaussian kernel functions centered at each of a large number of stored prototype feature vectors.
- These prototypes were made ahead of time from a training set of fingerprints of known class by using the same preprocessing and feature extraction that was used to produce the incoming feature vector.

PCASYS --Probabilistic Neural Network Classifier

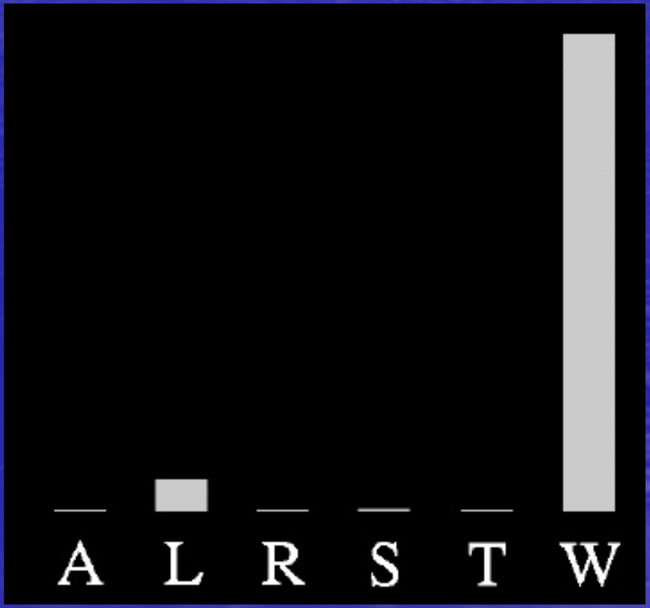
- For each class, an activation is made by adding up the values of the kernels centered at all prototypes of that class; the hypothesized class is then defined to be the one whose activation is largest.

PCASYS --Probabilistic Neural Network Classifier

- This is a bar graph of the normalized activations produced for the example fingerprint.
- All 6 normalized activations are shown here.
- The whorl (W) class has won and so is the hypothesized class (correctly as it turns out), but the left loop (L) class has also received a fairly large activation and therefore the confidence is only moderately high.

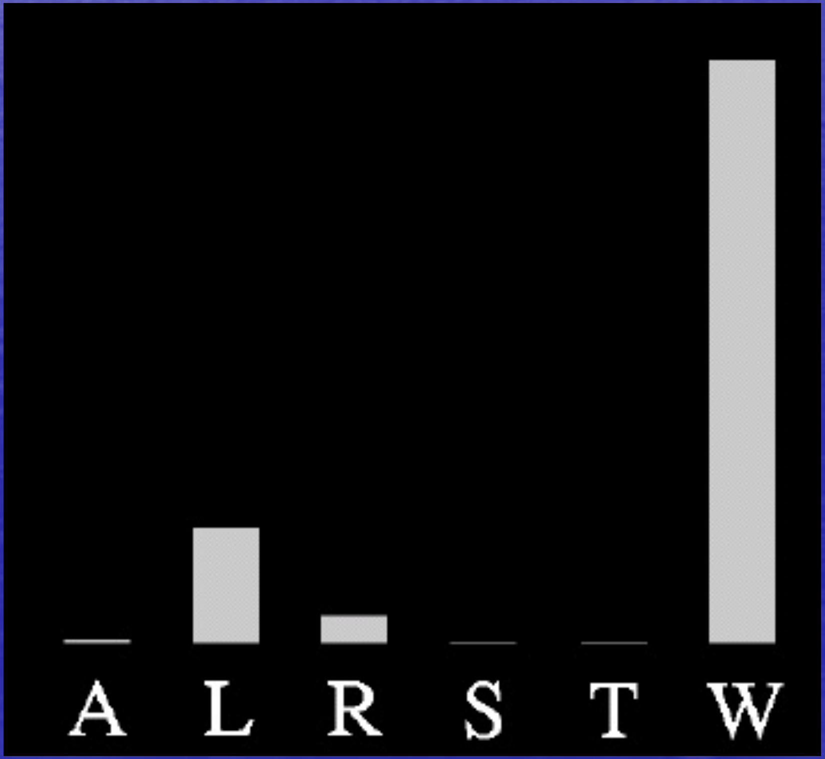


PNN output activations for the example fingerprint



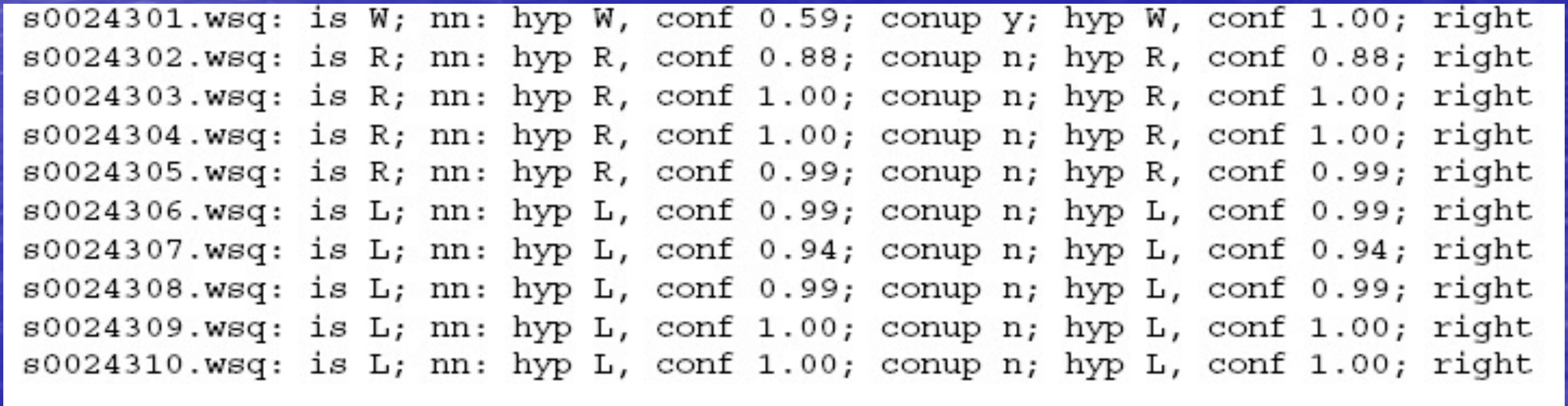
PCASYS -- Multi-Layer Perceptron Neural Network Classifier

- MLP output activations for the example fingerprint



PCASYS - Output

- Each line shows: the fingerprint filename; the actual class (A, L, R, S, T, and W stand for the pattern-level classes arch, left loop, right loop, sear, tented arch, and whorl); the output of the classifier (a hypothesized class and a confidence); the output of the auxiliary pseudo-ridge tracing whorl detector (whether or not a concave-upward shape, a "conup," was found); the final output of the hybrid classifier, which is a hypothesized class and a confidence; and whether this hypothesized class was right or wrong.



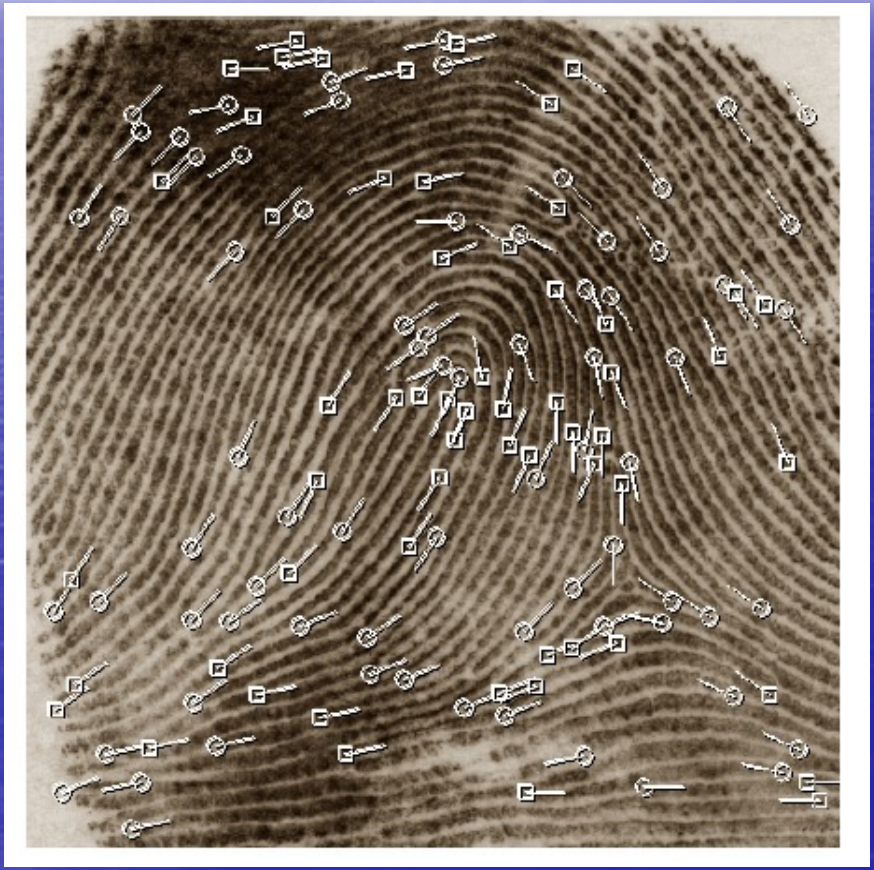
```
s0024301.wsq: is W; nn: hyp W, conf 0.59; conup y; hyp W, conf 1.00; right
s0024302.wsq: is R; nn: hyp R, conf 0.88; conup n; hyp R, conf 0.88; right
s0024303.wsq: is R; nn: hyp R, conf 1.00; conup n; hyp R, conf 1.00; right
s0024304.wsq: is R; nn: hyp R, conf 1.00; conup n; hyp R, conf 1.00; right
s0024305.wsq: is R; nn: hyp R, conf 0.99; conup n; hyp R, conf 0.99; right
s0024306.wsq: is L; nn: hyp L, conf 0.99; conup n; hyp L, conf 0.99; right
s0024307.wsq: is L; nn: hyp L, conf 0.94; conup n; hyp L, conf 0.94; right
s0024308.wsq: is L; nn: hyp L, conf 0.99; conup n; hyp L, conf 0.99; right
s0024309.wsq: is L; nn: hyp L, conf 1.00; conup n; hyp L, conf 1.00; right
s0024310.wsq: is L; nn: hyp L, conf 1.00; conup n; hyp L, conf 1.00; right
```

Minutiae Detection (MINDTCT)

Minutiae Detection (MINDTCT)

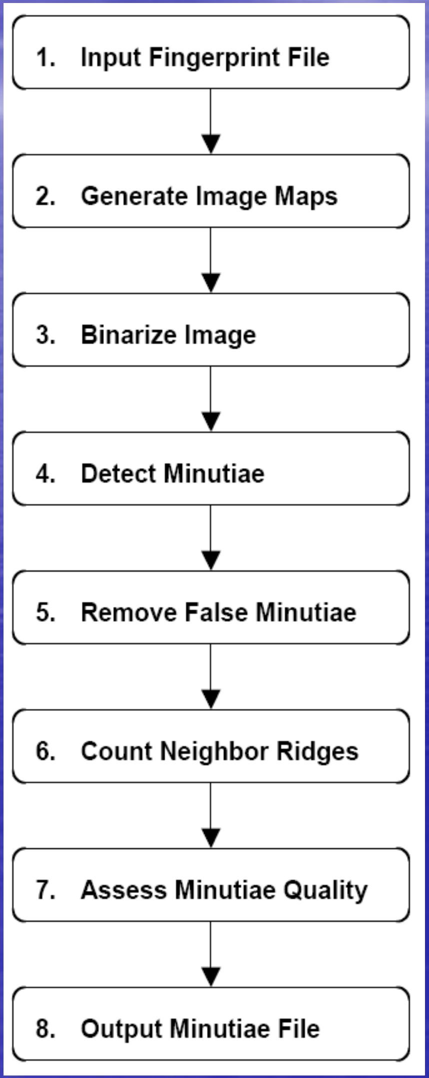
- MINDTCT takes a fingerprint image and locates all minutiae in the image, assigning to each minutia point its location, orientation, type, and quality.
- The command, mindtct, reads a fingerprint image from an ANSI/NIST, WSQ, baseline JPEG, lossless JPEG file, or IHead formatted file.
- Mindtct outputs the minutiae identification based on the ANSI/NIST standard or the M1 (ANSI INCITS 378-2004) representation.

Detected Minutiae



Minutiae results

MINDTCT



Minutiae detection process

MINDTCT -- Generate Image Quality Maps

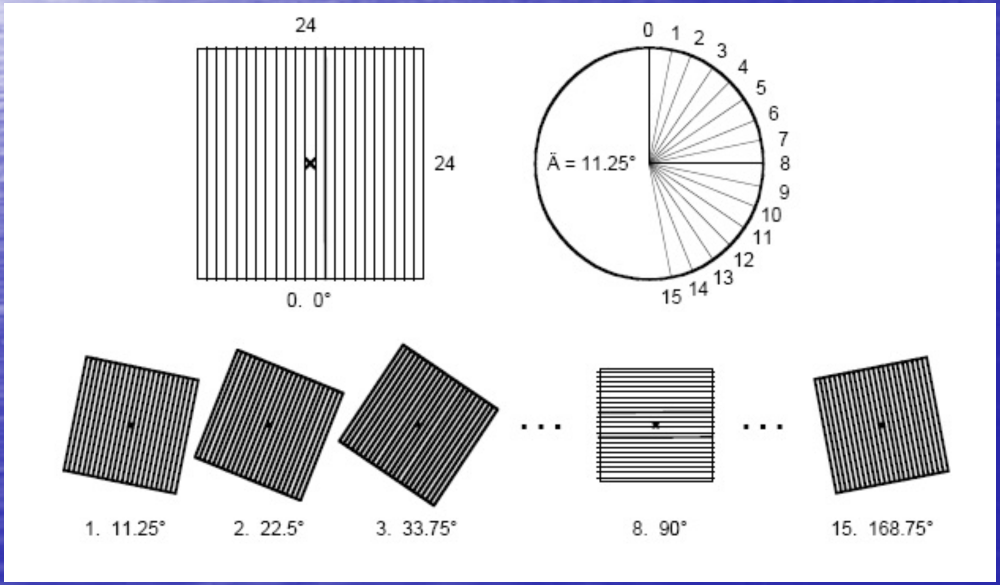
- The image quality of a fingerprint may vary, especially in the case of latent fingerprints, it is critical to be able to analyze the image and determine areas that are degraded and likely to cause problems.
- Several characteristics can be measured that are designed to convey information regarding the quality of localized regions in the image.
- These include determining the directional flow of ridges in the image and detecting regions of low contrast, low ridge flow, and high curvature.
- These conditions represent unstable areas in the image where minutiae detection is unreliable, and together they can be used to represent levels of quality in the image.

MINDTCT -- Direction Map

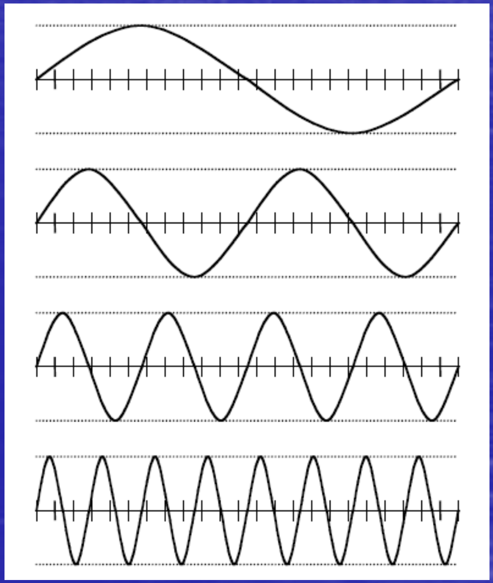
- One of the fundamental steps in this minutiae detection process is deriving a directional ridge flow map, or *direction map*.
- The purpose of this map is to represent areas of the image with sufficient ridge structure. Well-formed and clearly visible ridges are essential to reliably detecting points of ridge ending and bifurcation.
- In addition, the direction map records the general orientation of the ridges as they flow across the image.

MINDTCT -- Direction Map

- Within an orientation, the pixels along each rotated row of the window are summed together, forming a vector of 24 pixel row sums.
- The resonance coefficients produced from convolving each of the 16 orientation's row sum vectors with the 4 different discrete waveforms are stored and then analyzed.
- Generally, the dominant ridge flow direction for the block is determined by the orientation with maximum waveform resonance.

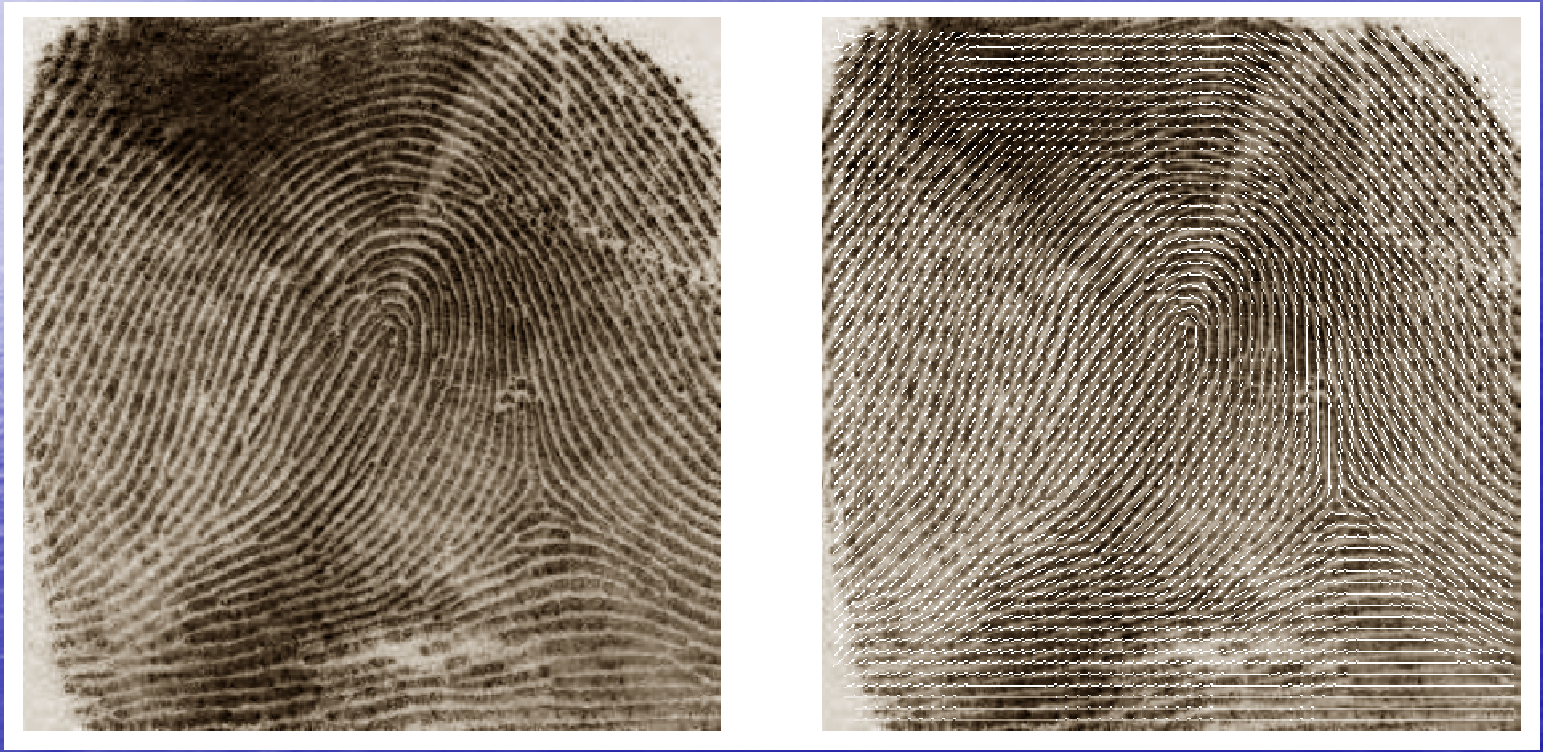


Window rotation at incremental orientations



waveform frequencies

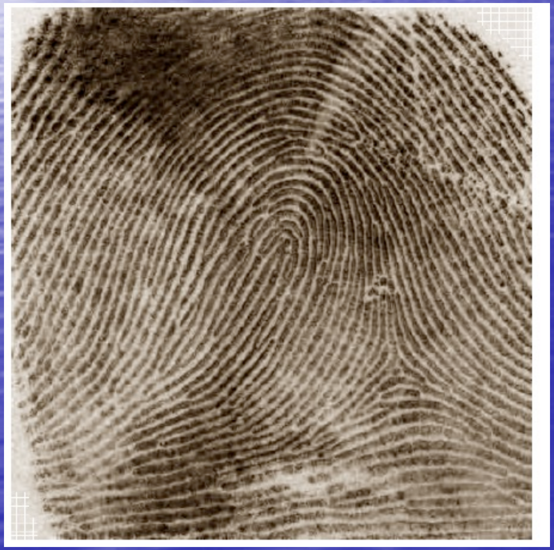
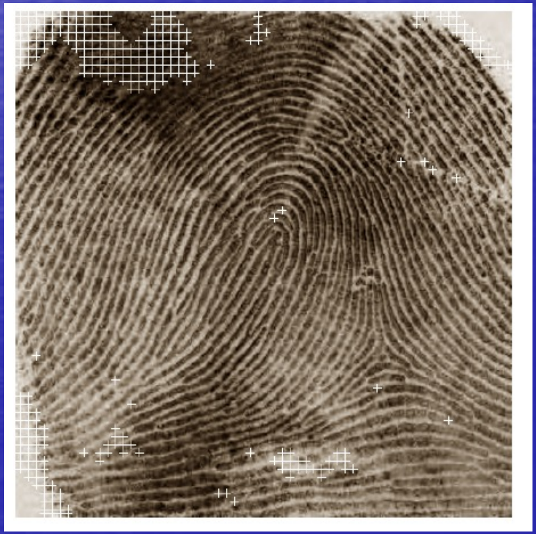
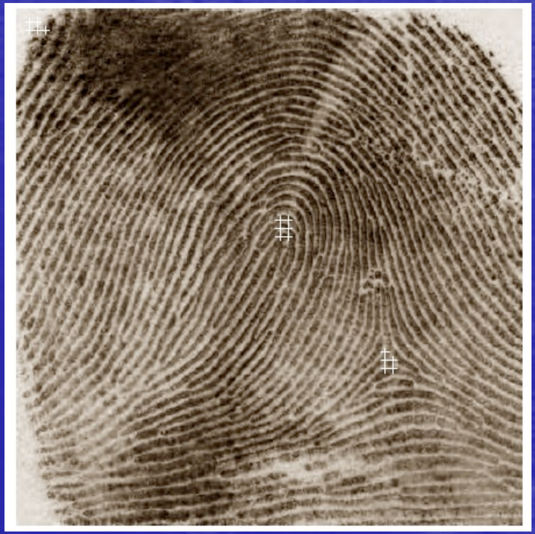
MINDTCT -- Direction Map



- Direction Map Result

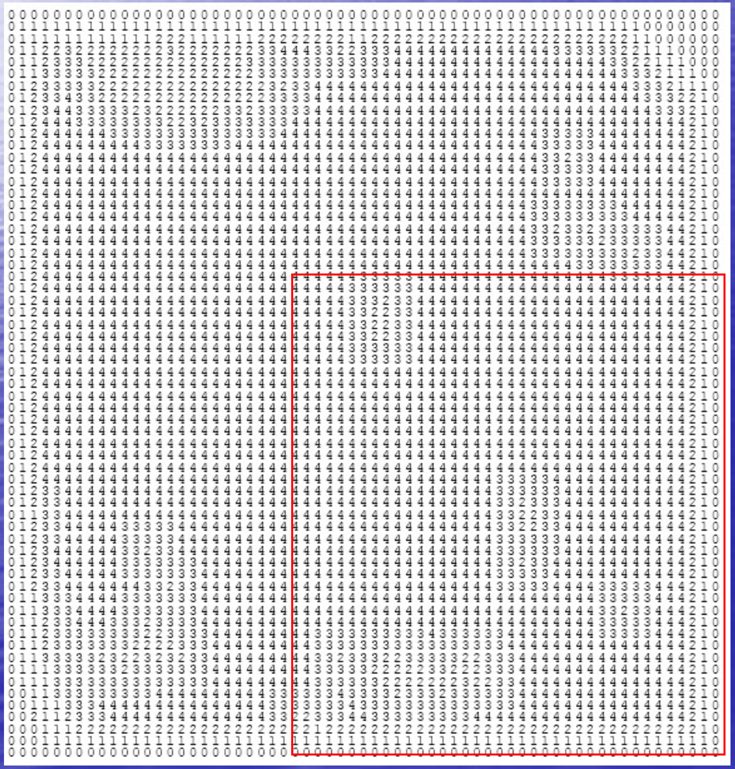
MINDTCT

- Low Contrast Map
- Low Flow Map
- High curve map

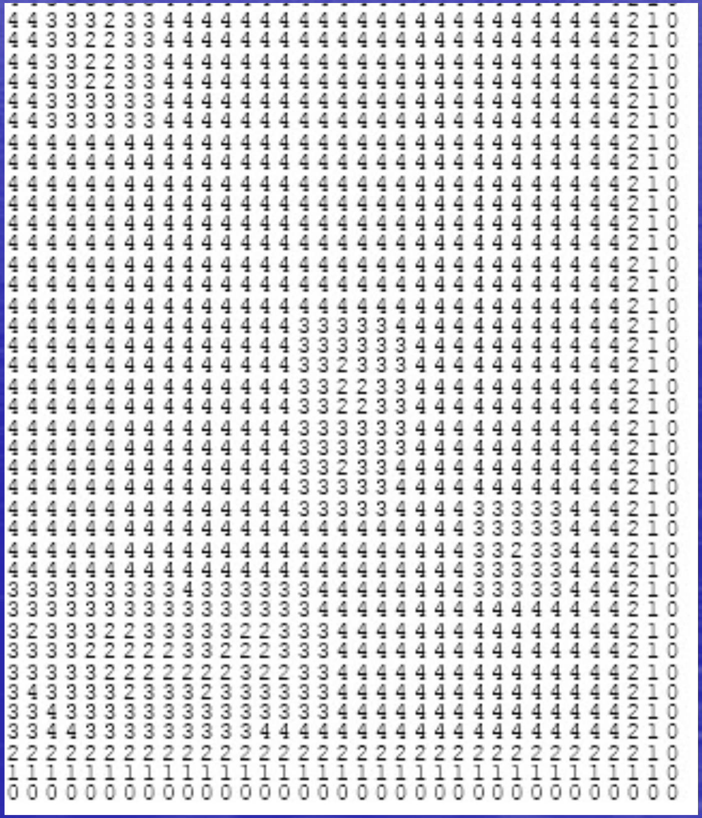




MINDTCT--Quality Map

- the low contrast map, low flow map, and the high curve map all point to different low quality regions of the image.
- The information in these maps is integrated into one general map, and contains 5 levels of quality.



```
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0
0 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 1 1 1 1 1 1 2 2 2 2 2 2 2 2 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 0 0 0 0 0 0 0
0 1 1 2 2 3 2 2 2 2 2 2 2 2 3 2 2 2 2 2 2 2 3 3 4 4 4 3 3 2 2 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 3 3 3 3 2 2 1 1 1 0 0 0 0
0 1 1 2 3 3 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 2 2 1 1 1 0 0 0
0 1 1 3 3 3 3 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 3 2 1 1 1 0 0
0 1 2 3 3 3 3 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 2 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 3 2 1 1 1 0
0 1 2 3 3 4 3 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 3 2 2 1 0
0 1 2 3 4 4 3 3 3 3 2 3 2 2 2 2 2 2 3 3 2 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 3 2 1 0
0 1 2 4 4 4 3 3 3 3 3 3 3 3 2 2 3 2 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
0 1 2 4 4 4 4 4 4 3 3 3 3 3 3 3 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 3 3 3 4 4 4 4 4 4 4 4 2 1 0
0 1 2 4 4 4 4 4 4 4 4 4 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 3 3 3 4 4 4 4 4 4 4 4 2 1 0
0 1 2 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 2 3 3 4 4 4 4 4 4 4 4 2 1 0
0 1 2 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 3 3 3 4 4 4 4 4 4 4 4 2 1 0
0 1 2 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 3 3 3 4 4 4 4 4 4 4 4 2 1 0
0 1 2 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
0 1 2 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 3 3 3 3 3 3 4 4 4 4 4 4 2 1 0
0 1 2 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 3 3 3 3 3 3 3 4 4 4 4 4 2 1 0
0 1 2 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 2 3 3 2 3 3 3 3 3 3 4 4 2 1 0
0 1 2 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 3 3 3 3 2 3 3 3 3 3 4 4 2 1 0
0 1 2 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 3 3 3 3 3 3 3 2 3 3 4 4 2 1 0
0 1 2 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 3 3 3 3 3 3 4 4 2 1 0
0 1 2 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 3 3 3 4 4 2 1 0
0 1 2 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
0 1 2 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 3 2 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
0 1 2 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 2 2 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
0 1 2 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 2 2 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
0 1 2 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 2 2 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
0 1 2 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
0 1 2 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
0 1 2 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
0 1 2 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
0 1 2 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
0 1 2 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
0 1 2 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
0 1 2 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
0 1 2 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
0 1 2 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
0 1 2 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
0 1 2 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
0 1 2 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 2 1 0
0 1 2 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 2 3 3 3 4 4 4 4 4 4 4 4 4 4 4 2 1 0
0 1 1 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 2 2 3 3 4 4 4 4 4 4 4 4 4 4 4 2 1 0
0 1 2 3 3 4 4 4 4 4 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 2 2 3 3 4 4 4 4 4 4 4 4 4 4 4 2 1 0
0 1 2 3 3 4 4 4 4 4 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 2 1 0
0 1 2 3 4 4 4 4 4 4 3 3 2 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 2 1 0
0 1 2 3 4 4 4 4 4 4 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 2 3 3 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
0 1 2 3 3 4 4 4 4 4 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
0 1 2 3 3 4 4 4 4 4 4 4 3 3 2 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 3 3 3 4 4 4 4 3 3 3 3 3 4 4 4 2 1 0
0 1 1 3 3 3 4 4 4 4 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 3 3 3 4 4 4 2 1 0
0 1 1 3 3 3 4 4 4 4 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 2 3 3 4 4 4 2 1 0
0 1 1 2 3 3 4 4 4 3 3 3 2 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 3 3 3 4 4 4 2 1 0
0 1 1 2 3 3 3 3 3 3 3 2 2 3 3 3 3 4 4 4 4 4 4 4 4 4 3 3 3 3 3 3 3 3 3 3 4 3 3 3 3 3 3 4 4 4 4 4 4 4 4 3 3 3 3 3 4 4 4 2 1 0
0 1 1 2 3 3 3 3 3 3 3 2 2 2 2 3 3 3 4 4 4 4 4 4 4 4 4 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
0 1 1 3 3 3 3 3 2 3 2 2 3 2 3 2 3 3 4 4 4 4 4 4 4 4 4 3 3 2 3 3 3 2 2 3 3 3 3 3 2 2 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
0 1 1 1 3 3 3 3 3 3 2 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 3 3 3 3 3 2 2 2 2 2 3 3 2 2 2 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
0 1 1 1 3 3 3 3 3 3 3 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 3 3 3 3 3 3 2 2 2 2 2 2 2 3 2 2 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
0 0 1 1 3 3 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 3 3 3 3 3 3 4 3 3 3 3 2 3 3 3 2 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
0 0 1 1 1 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 3 3 3 3 3 4 3 3 3 3 3 3 3 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
0 0 2 1 1 2 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 2 2 3 3 3 4 4 3 3 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
0 0 0 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 0
0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```



```
4 4 3 3 3 2 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
4 4 3 3 2 2 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
4 4 3 3 2 2 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
4 4 3 3 2 2 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
4 4 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
4 4 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 2 1 0
4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 2 3 3 3 4 4 4 4 4 4 4 4 4 4 4 2 1 0
4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 2 2 3 3 4 4 4 4 4 4 4 4 4 4 4 2 1 0
4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 2 2 3 3 4 4 4 4 4 4 4 4 4 4 4 2 1 0
4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 2 1 0
4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 2 1 0
4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 2 3 3 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 3 3 3 4 4 4 4 3 3 3 3 3 4 4 4 2 1 0
4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 3 3 3 4 4 4 2 1 0
4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 2 3 3 4 4 4 2 1 0
4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 3 3 3 3 3 4 4 4 2 1 0
3 3 3 3 3 3 3 3 3 4 3 3 3 3 3 3 4 4 4 4 4 4 4 4 3 3 3 3 3 4 4 4 2 1 0
3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
3 2 3 3 3 2 2 3 3 3 3 3 2 2 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
3 3 3 3 2 2 2 2 2 3 3 2 2 2 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
3 3 3 3 3 2 2 2 2 2 2 2 3 2 2 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
3 4 3 3 3 3 2 3 3 3 2 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
3 3 4 3 3 3 3 3 3 3 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
3 3 4 4 3 3 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 1 0
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 0
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

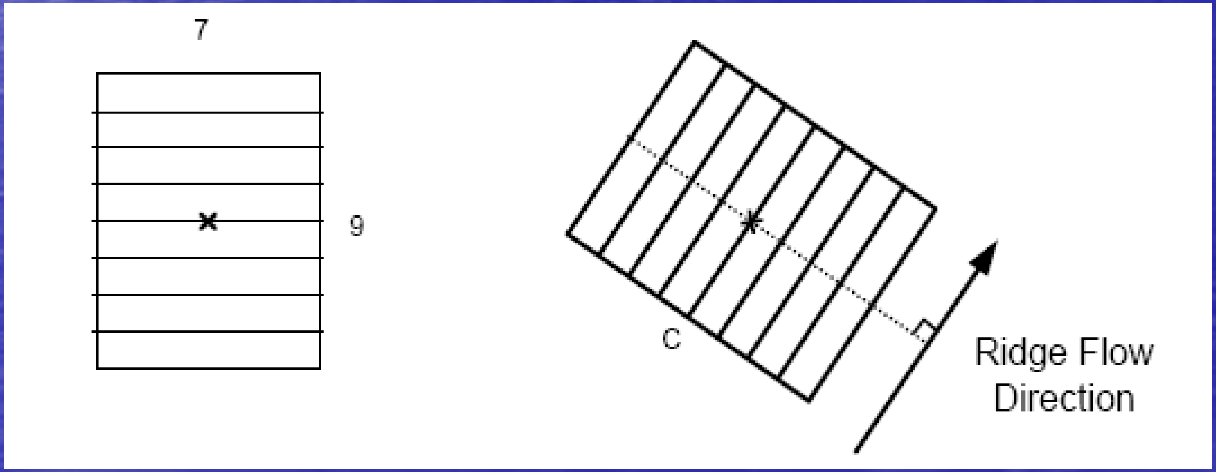
Portion of Quality Map

MINDTCT -- Binarize Image

- The minutiae detection algorithm in this system is designed to operate on a bi-level (or binary) image where black pixels represent ridges and white pixels represent valleys in a finger's friction skin.
- To create this binary image, every pixel in the grayscale input image must be analyzed to determine if it should be assigned a black or white pixel.
- This process is referred to as image *binarization*.

MINDTCT -- Binarize Image

- A pixel is assigned a binary value based on the ridge flow direction associated with the block the pixel is within.
- With the pixel of interest in the center, the grid is rotated so that its rows are parallel to the local ridge flow direction

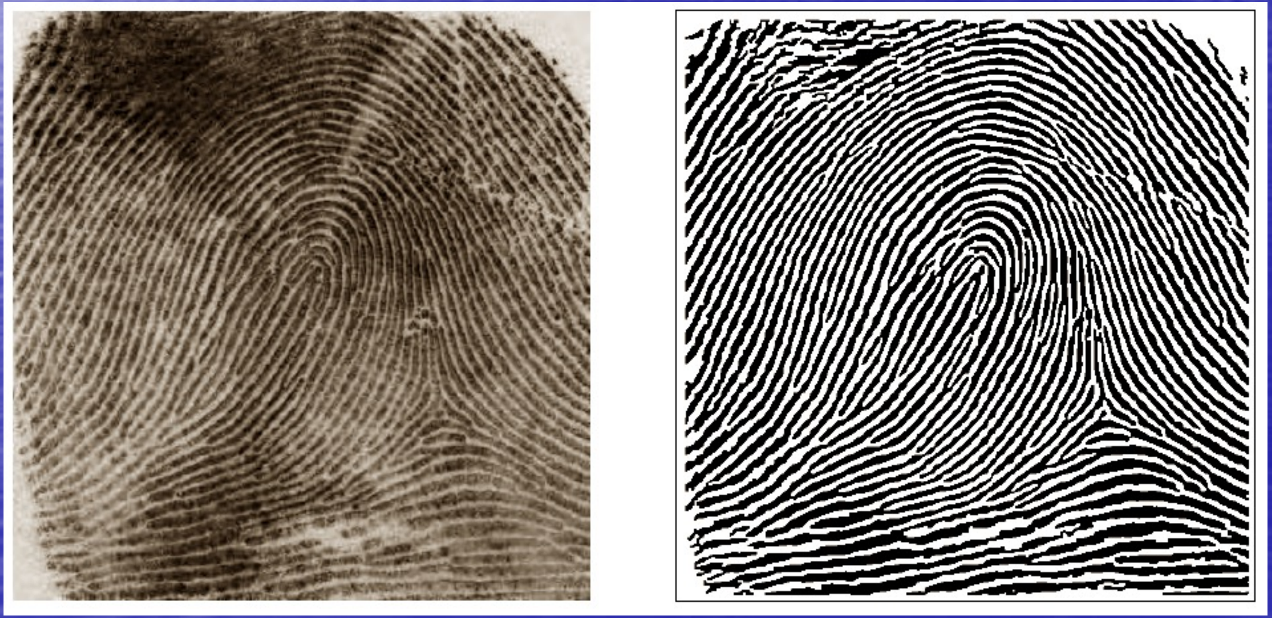


MINDTCT -- Binarize Image

- Grayscale pixel intensities are accumulated along each rotated row in the grid, forming a vector of row sums.
- The binary value to be assigned to the center pixel is determined by multiplying the center row sum by the number of rows in the grid and comparing this value to the accumulated grayscale intensities within the entire grid.
- If the multiplied center row sum is less than the grid's total intensity, then the center pixel is set to black; otherwise, it is set to white.

MINDTCT -- Binarize Image

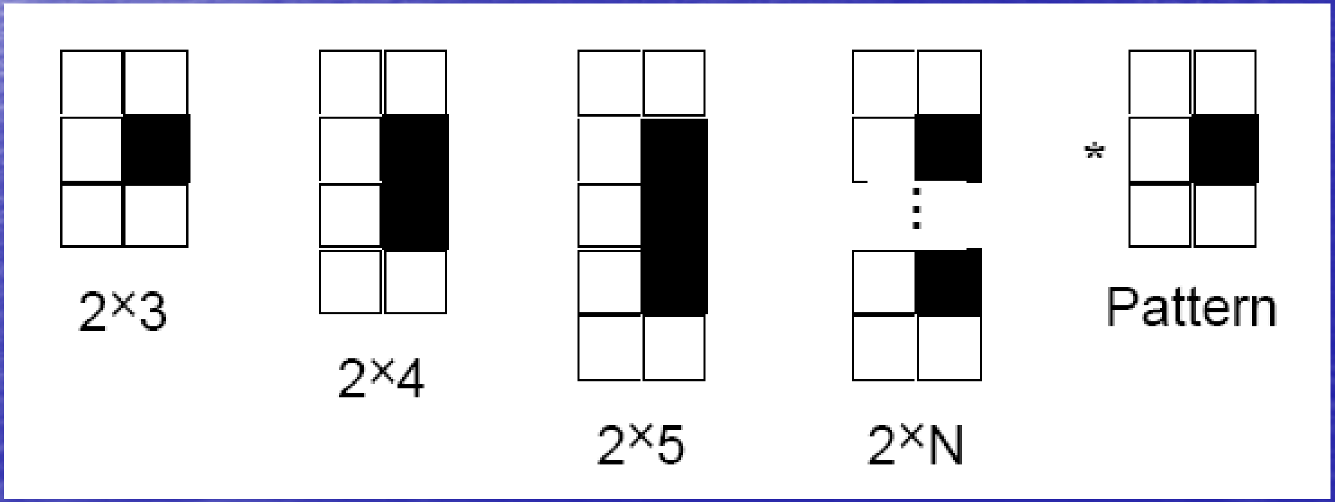
- The binarization results need to be *robust* in terms of effectively dealing with varying degrees of image quality and *reliable* in terms of rendering ridge and valley structures accurately.



Binarization results

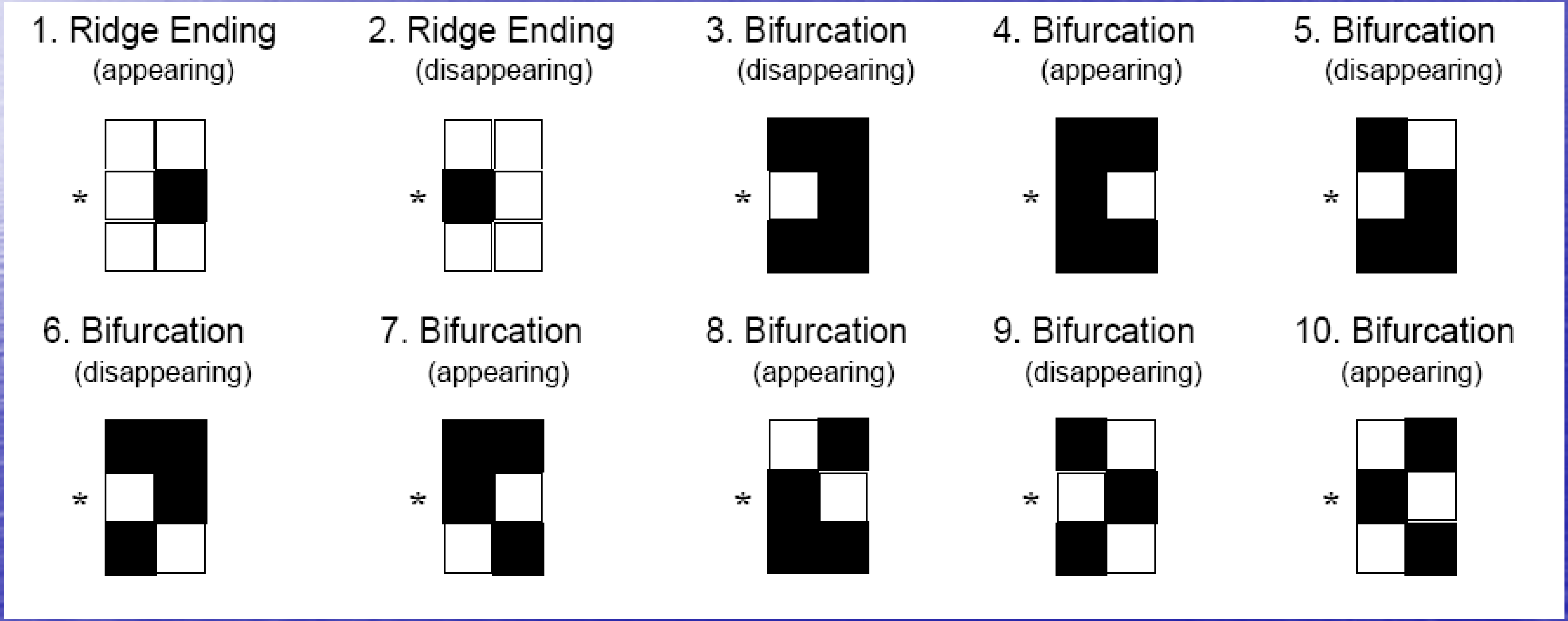
MINDTCT--Detect Minutiae

- This step methodically scans the binary image of a fingerprint, identifying localized pixel patterns that indicate the ending or splitting of a ridge.
- This pattern may represent the end of a black ridge protruding into the pattern from the right.



Pixel pattern used to detect ridge endings

MINDTCT -- Pixel patterns used to detect minutiae


1. Ridge Ending (appearing)
2. Ridge Ending (disappearing)
3. Bifurcation (disappearing)
4. Bifurcation (appearing)
5. Bifurcation (disappearing)
6. Bifurcation (disappearing)
7. Bifurcation (appearing)
8. Bifurcation (appearing)
9. Bifurcation (disappearing)
10. Bifurcation (appearing)
*

Remove False Minutiae

- These steps include removing islands, lakes, holes, minutiae in regions of poor image quality, side minutiae, hooks, overlaps, minutiae that are too wide, and minutiae that are too narrow (pores).

MINDTCT -- Output Minutiae File

- The direction map is stored in <oroot>.dm;
- The low contrast map is stored in <oroot>.lcm;
- The low flow map is stored in <oroot>.lfm;
- The high curve map is stored in <oroot>.hcm; and
- The quality map is stored in <oroot>.qm.
- The maps are represented by a grid of numbers, each corresponding to a block in the fingerprint image.
- The resulting minutiae can be accessed in the text file <oroot>.min containing a formatted listing of attributes associated with each detected minutiae in the fingerprint image.
- For all input types the detected minutiae are also written to a text file <oroot>.xyt that is formatted for use with the bozorth3 matcher.
- This file has one space delimited line per minutiae containing its x and y coordinate, direction angle theta, and the minutiae quality.

Image Quality (NFIQ)

Image Quality (NFIQ)

- The NFIQ algorithm is an implementation of the NIST "Fingerprint Image Quality" algorithm.
- It takes an input image that is in ANSI/NIST or NIST IHEAD format or compressed using WSQ, baseline JPEG, or lossless JPEG.
- NFIQ outputs the image quality value
- for the image (where 1 is highest quality and 5 is lowest quality).

Image Quality (NFIQ)

- Neural networks offer a very powerful and very general framework for representing non-linear mappings from several input variables to several output variables, where the form of the mapping is governed by a number of adjustable parameters (*weights*).
- The process of determining the values for these weights based on the data set is called *training* and the data set of examples is generally referred to as a *training set*.

How to perform training

- Fing2pat gets the list of gray-scale fingerprint images in the training set along with their class labels as input and computes patterns for MLP training and writes them to a binary file.
- Each pattern consists of a feature vector, along with a class vector.
- The user can compute the global mean and standard deviation statistics using znormdat or use the set provided in nfiq/znorm.dat.
- These global statistics can be applied to new pattern files using nzormpat.
- The user needs to write a spec file, setting parameters of the training runs that MLP is to perform.
- The spec file used in the training of NIST Fingerprint Image Quality can be found in the file nfiq/spec.

Minutiae Matching (BOZORTH3)

Minutiae Matching (BOZORTH3)

- The BOZORTH3 matcher uses only the location (x,y) and orientation (theta) of the minutia points to match the fingerprints.
- The matcher builds separate tables for the fingerprints being matched that define distance and orientation between minutia in each fingerprint.
- These two tables are then compared for compatibility and a new table is constructed that stores information showing the inter-fingerprint compatibility.
- The inter-finger compatibility table is used to create a match score by looking at the size and number of compatible minutia clusters.

Minutiae Matching (BOZORTH3)

- Two key things are important to note regarding this fingerprint matcher:

1. Minutia features are exclusively used and limited to location (x,y) and orientation 't', represented as {x,y,t}.
2. The algorithm is designed to be rotation and translation invariant.

Minutiae Matching (BOZORTH3)

- The algorithm is comprised of three major steps:

1. Construct Intra-Fingerprint Minutia Comparison Tables
   - One table for the probe fingerprint and
   - One table for each gallery fingerprint to be matched against.
2. Construct an Inter-Fingerprint Compatibility Table
   - Compare a probe print's minutia table to a gallery print's minutia table and construct a new inter-fingerprint compatibility table.
3. Traverse the Inter-Fingerprint Compatibility Table

   a. Traverse and link table entries into clusters

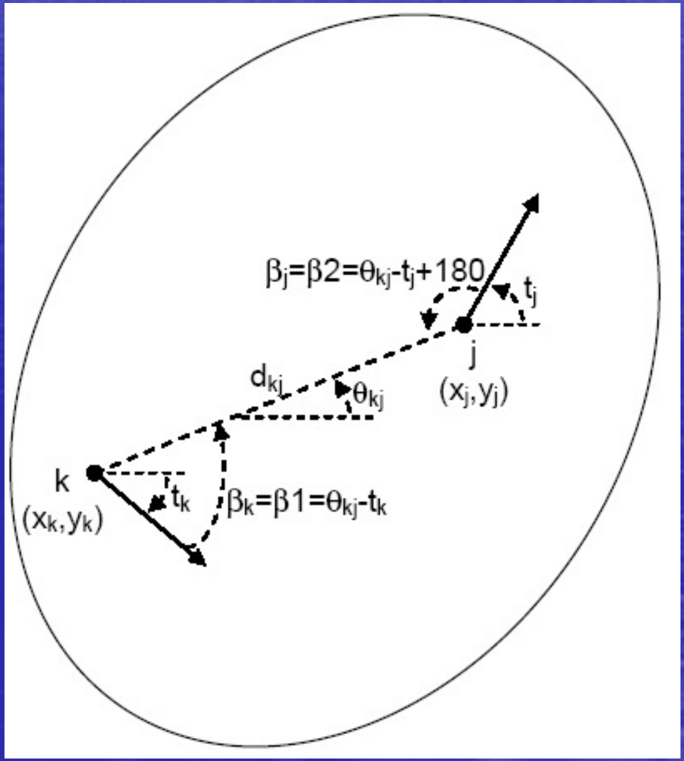
   b. Combine compatible clusters and accumulate a match score.

Minutiae Matching (BOZORTH3)

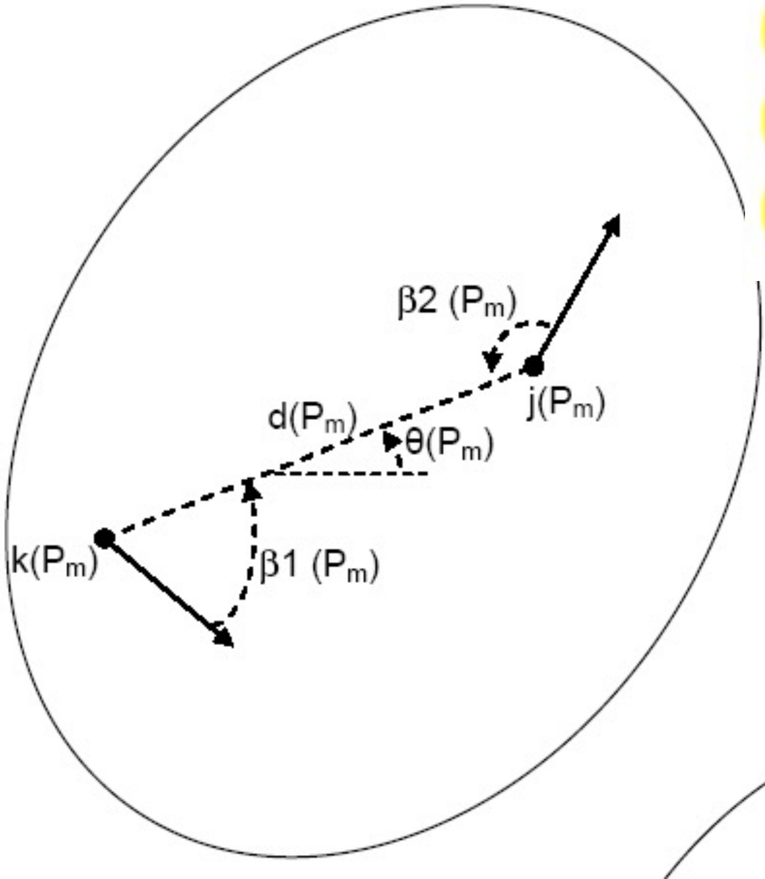
- Construct Intra-Fingerprint Minutia Comparison Tables
  - Compute relative measurements from each minutia in a fingerprint to all other minutia in the same fingerprint.
  - These relative measurements are stored in a minutia comparison table and are what provide the algorithm's rotation and translation invariance.

Construct Intra-Fingerprint Minutia Comparison Tables

- the distance d_{kj} is computed between the two minutia locations.
- the angle of each minutia's orientation and the intervening line between both minutiae.
- Each entry consists of $\{d_{kj}, β1, β2, k, j, θkj\}$ where $β1 = \min(βk,βj)$ and $β2 = \max(βk,βj)$



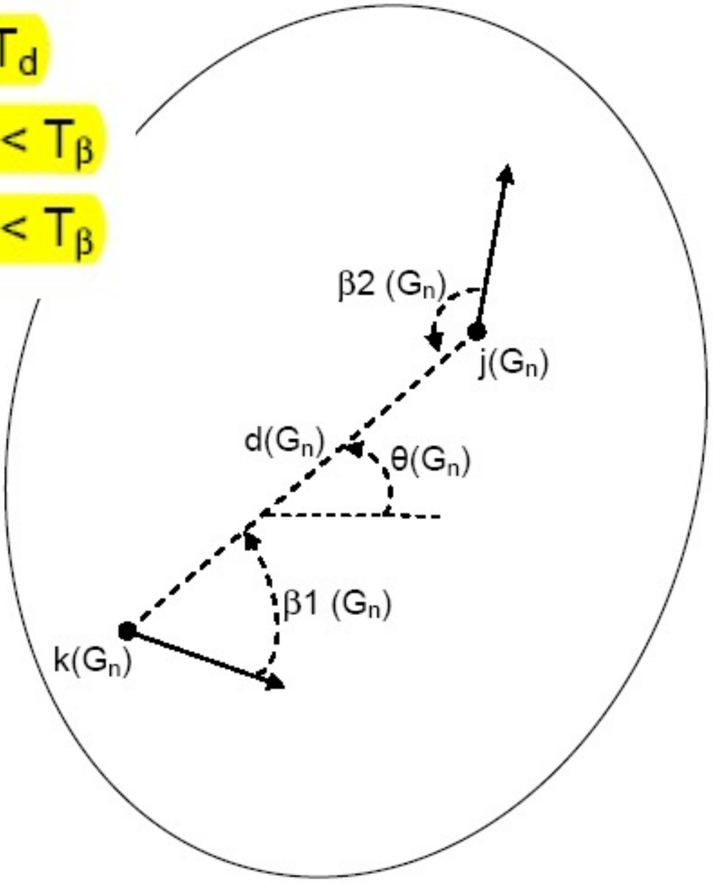
- The following three tests are conducted to determine if table entries Pm and Gn are “compatible.”
- The first test checks to see if the corresponding distances are within a specified tolerance Td.
- The last two tests check to see if the relative minutia angles are within a specified tolerance Tβ. Δd () and Δβ () are “delta” or difference functions.



$\Delta_d(d(P_m),d(G_n)) < T_d$

$\Delta_\beta(\beta 1(P_m),\beta 1(G_n)) < T_\beta$

$\Delta_\beta(\beta 2(P_m),\beta 2(G_n)) < T_\beta$



Traverse the Inter-Fingerprint Compatibility Table

- At this point in the process, we have constructed a compatibility table which consists of a list of compatibility association between two pairs of potentially corresponding minutiae.
- These associations represent single links in a <u>compatibility graph</u>.
- To determine how well the two fingerprints match each other, a simple goal would be to traverse the compatibility graph finding the longest path of linked compatibility associations.
- The match score would then be the length of the longest path.