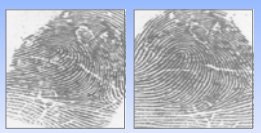
Finger Print Recognition using Minutiae

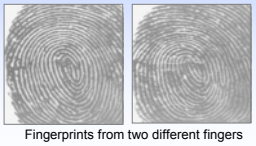
Biometrics
CSE 190-a
Lecture 10

Fingerprint Matching

- Find the similarity between two fingerprints



Fingerprints from the same finger



Fingerprints from two different fingers

Fingerprint Sensors

- Optical, capacitive, ultrasound, pressure, thermal, electric field

Fingerprint Classification

- Assign fingerprints into one of pre-specified types

Plain Arch | Tented Arch | Right Loop | Left Loop

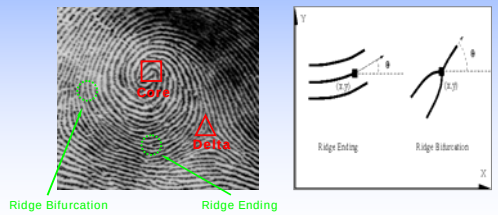
Accidental | Pocket Whorl | Plain Whorl | Double Loop

Terminology

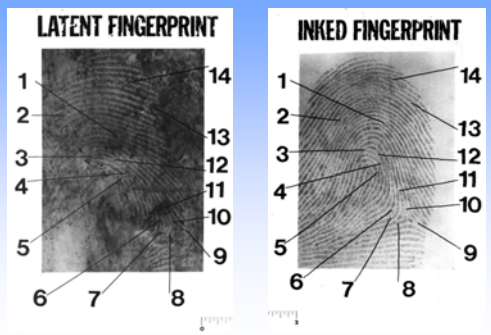
- Fingerprint – Impression of a finger
- Minutiae – Ridge bifurcations,endings and many other features (52 types listed, 7 are usually used by human experts and two by automated systems)
- Core – uppermost point on the innermost ridge
- Delta – separating point between pattern area and non-pattern area

Fingerprint Representation

- Local ridge characteristics (minutiae): ridge ending and ridge bifurcation
- Singular points: Discontinuity in ridge orientation



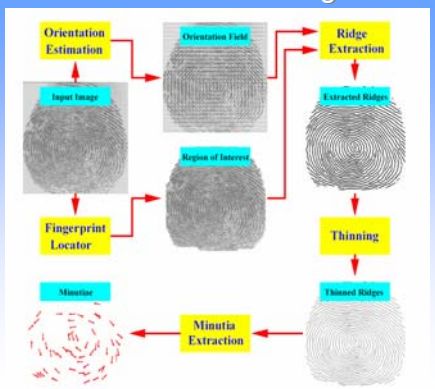
Minutiae-based Representation



Steps in Minutiae Extraction

- Orientation field estimation
- Fingerprint area location
- Ridge extraction
- Thinning
- Minutia extraction

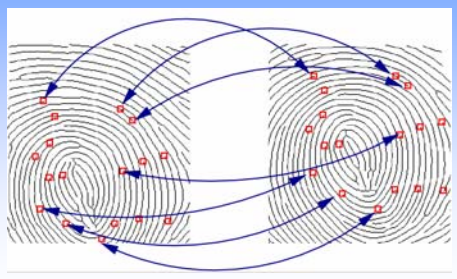
Minutiae Extraction Algorithm



Minutiae Type Detection

- A ridge pixel is a ridge ending, if the number of ridge pixels in the 8-neighborhood is 1
- A ridge pixel is a ridge bifurcation, if the number of ridge pixels in the 8-neighborhood is greater than or equal to 3
- A ridge pixel is a intermediate ridge pixel, if the number of ridge pixels in the 8-neighborhood is 2
- [x, y, θ, associated ridge] are stored for each minutia

Minutiae Correspondences



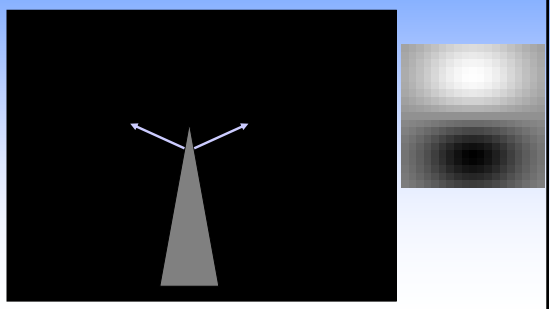
Minutiae Matching

- Point pattern matching problem
- Let $P = \{(x_1^P, y_1^P, \theta_1^P), \ldots, (x_M^P, y_M^P, \theta_M^P)\}$
  be the set of M minutiae in the template image
- Let $Q = \{(x_1^Q, y_1^Q, \theta_1^Q), \ldots, (x_N^Q, y_N^Q, \theta_N^Q)\}$
  be the set of N minutiae in the input image
- Find the number of corresponding minutia pairs between P and Q and compare it against a threshold

Stages of Minutiae-based Verification

- Extract Minutiae using corner detection
- Characterize (label) Minutiae
- Transformations between fingerprint images
- RANSAC

Corner Detection



Finding Corners

Intuition:

- Right at corner, gradient is ill-defined.
- Near corner, gradient has two different values.

What is image filtering?

- Modify the pixels in an image based on some function of a local neighborhood of the pixels.

(From Bill Freeman)

Convolution

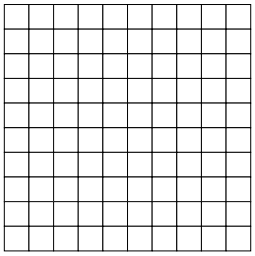


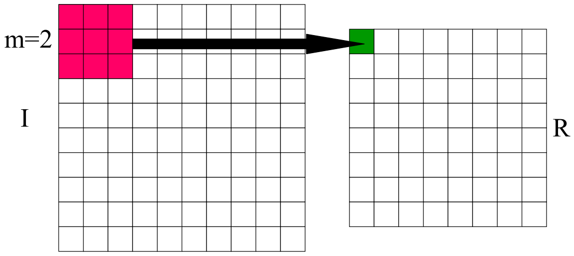
Image (I)

*

Kernel (K)

Note: Typically Kernel is relatively small in vision applications.

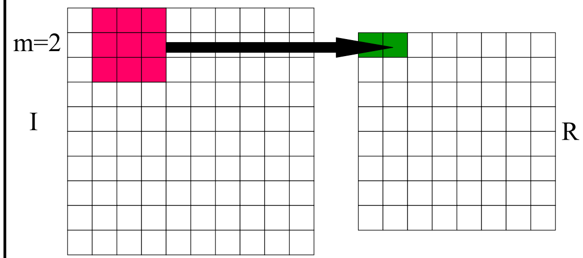
Convolution: R= K*I



Kernel size
is m+1 by m+1

$$R(i,j)=\sum_{h=-m/2}^{m/2}\sum_{k=-m/2}^{m/2}K(h,k)I(i-h,j-k)$$

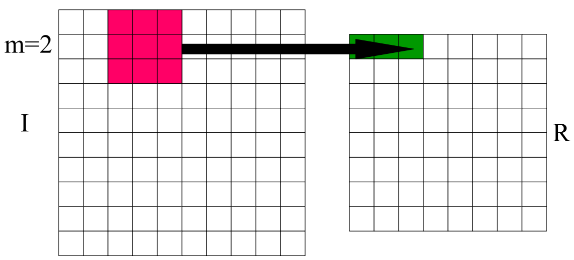
Convolution: R= K*I



Kernel size
is m+1 by m+1

$$R(i,j)=\sum_{h=-m/2}^{m/2}\sum_{k=-m/2}^{m/2}K(h,k)I(i-h,j-k)$$

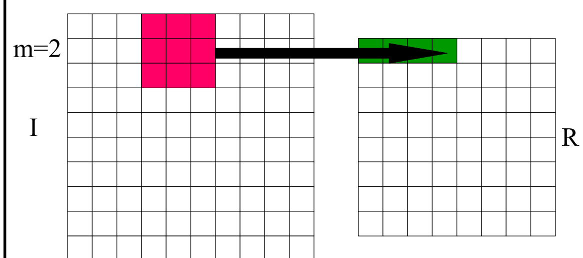
Convolution: R= K*I



Kernel size
is m+1 by m+1

$$R(i,j)=\sum_{h=-m/2}^{m/2}\sum_{k=-m/2}^{m/2}K(h,k)I(i-h,j-k)$$

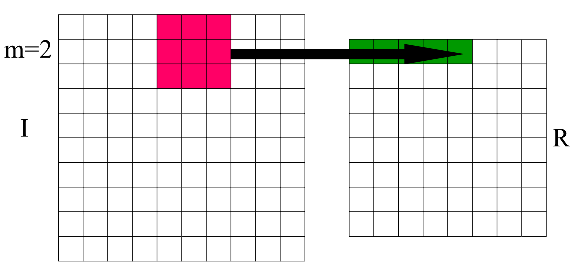
Convolution: R= K*I



Kernel size
is m+1 by m+1

$$R(i,j)=\sum_{h=-m/2}^{m/2}\sum_{k=-m/2}^{m/2}K(h,k)I(i-h,j-k)$$

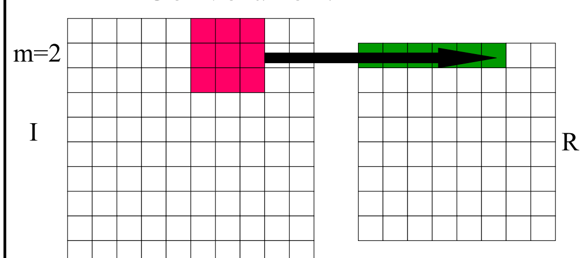
Convolution: R= K*I



Kernel size
is m+1 by m+1

$$R(i,j)=\sum_{h=-m/2}^{m/2}\sum_{k=-m/2}^{m/2}K(h,k)I(i-h,j-k)$$

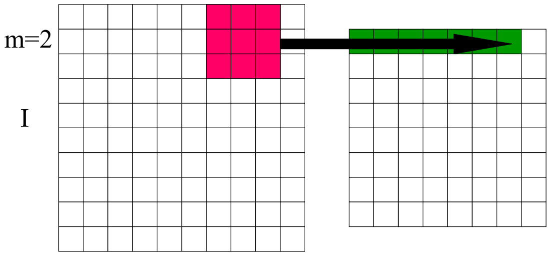
Convolution: R= K*I



Kernel size
is m+1 by m+1

$$R(i,j)=\sum_{h=-m/2}^{m/2}\sum_{k=-m/2}^{m/2}K(h,k)I(i-h,j-k)$$

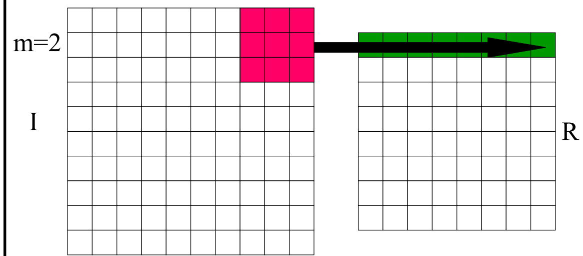
Convolution: R= K*I



Kernel size
is m+1 by m+1

$$R(i,j)=\sum_{h=-m/2}^{m/2}\sum_{k=-m/2}^{m/2}K(h,k)I(i-h,j-k)$$

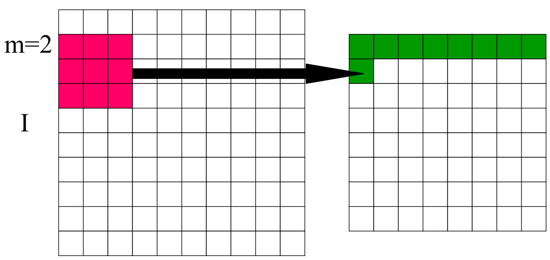
Convolution: R= K*I



Kernel size
is m+1 by m+1

$$R(i,j)=\sum_{h=-m/2}^{m/2}\sum_{k=-m/2}^{m/2}K(h,k)I(i-h,j-k)$$

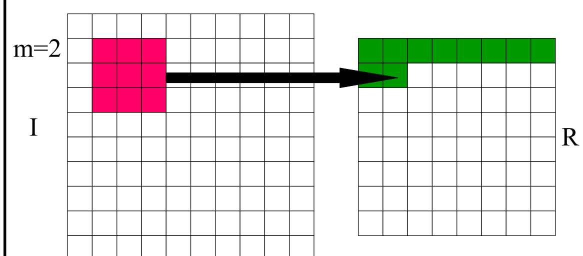
Convolution: R= K*I



Kernel size
is m+1 by m+1

$$R(i,j)=\sum_{h=-m/2}^{m/2}\sum_{k=-m/2}^{m/2}K(h,k)I(i-h,j-k)$$

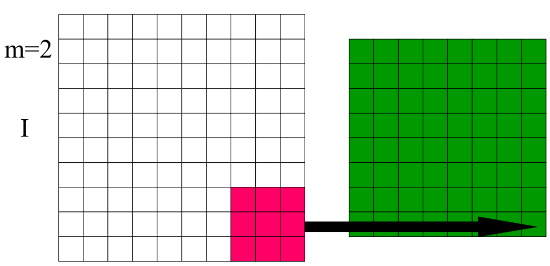
Convolution: R= K*I



Kernel size
is m+1 by m+1

$$R(i,j)=\sum_{h=-m/2}^{m/2}\sum_{k=-m/2}^{m/2}K(h,k)I(i-h,j-k)$$

Convolution: R= K*I



Kernel size
is m+1 by m+1

$$R(i,j)=\sum_{h=-m/2}^{m/2}\sum_{k=-m/2}^{m/2}K(h,k)I(i-h,j-k)$$

Image Noise

Gaussian Noise:
sigma=1

Gaussian Noise:
sigma=16

Average Filter

- Mask with positive entries, that sum 1.
- Replaces each pixel with an average of its neighborhood.
- If all weights are equal, it is called a BOX filter.

F

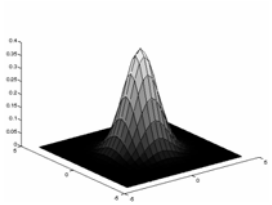
$$\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

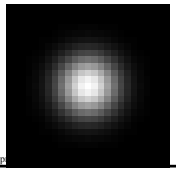
(Camps)

Smoothing by Averaging

Kernel: ■

An Isotropic Gaussian





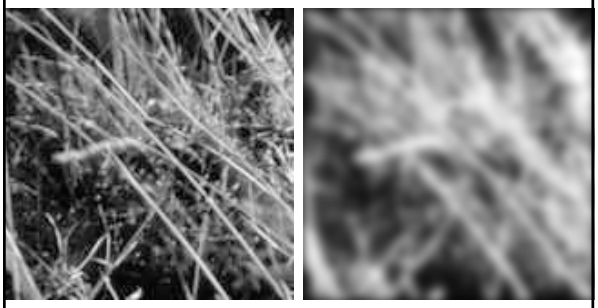
- The picture shows a smoothing kernel proportional to

$$\exp\left(-\left(\frac{x^2+y^2}{2\sigma^2}\right)\right)$$

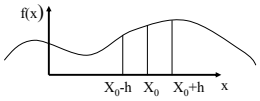
(which is a reasonable model of a circularly symmetric fuzzy blob)

Smoothing with a Gaussian

Kernel: ■



Numerical Derivatives



Take Taylor series expansion of f(x) about x_0

$f(x) = f(x_0)+f'(x_0)(x-x_0) + \frac{1}{2} f''(x_0)(x-x_0)^2 + \ldots$

Consider Samples taken at increments of h and first two terms, we have

$f(x_0+h) = f(x_0)+f'(x_0)h+ \frac{1}{2} f''(x_0)h^2$

$f(x_0-h) = f(x_0)-f'(x_0)h+ \frac{1}{2} f''(x_0)h^2$

Subtracting and adding $f(x_0+h)$ and $f(x_0-h)$ respectively yields

$$f'(x_0) = \frac{f(x_0+h) - f(x_0-h)}{2h}$$

$$f''(x_0) = \frac{-f(x_0+h) + 2f(x_0) - f(x_0-h)}{2h^2}$$

On numerical derivatives

Convolve with

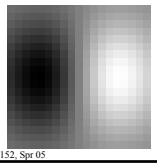
First Derivative: [-1 0 1]

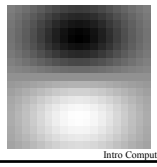
Second Derivative: [-1 2 -1]

First Derivative in Y Direction:$[-1\ 0\ 1]^T$

Smoothing and Differentiation

- Need two derivatives, in x and y direction.
- Filter with Gaussian and then compute gradient, or
- Use a derivative of Gaussian filter
  - because differentiation is convolution, and convolution is associative





Formula for Finding Corners

Let $I_x = \frac{\partial I}{\partial x}, and\ I_y = \frac{\partial I}{\partial y}$

Sum over a small region, the hypothetical corner

Gradient with respect to x, times gradient with respect to y

$$C = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix}$$

Matrix is symmetric

WHY THIS?

First, consider case where:

$$C = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$

This means all gradients in neighborhood are:

(k,0) or (0, c) or (0, 0) (or off-diagonals cancel).

What is region like if:

1. $\lambda_1 = 0$?
2. $\lambda_2 = 0$?
3. $\lambda_1 = 0$ and $\lambda_2 = 0$?
4. $\lambda_1 > 0$ and $\lambda_2 > 0$?

General Case:

From Linear Algebra, it follows that

$$C = R^{-1} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R$$

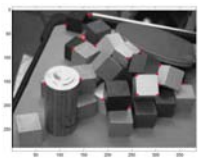
since C is symmetric. So every case is like the one on the last slide.

So, to detect corners

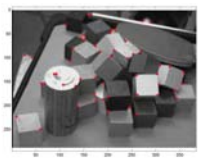
- Filter image.
- Compute the gradient everywhere.
- We construct C in a window of some size.
- Use linear algebra to find λ_1 and λ_2.
- If λ_1 and λ_2 are both big, we have a corner.
  1. Let $e(u,v) = \min(\lambda_1(u,v), \lambda_2(u,v))$
  2. (u,v) is a corner local maximum of e(u,v) and $e(u,v) > \tau$

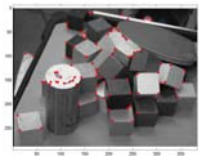
Corner Detection Sample Results

Threshold=25,000



Threshold=10,000



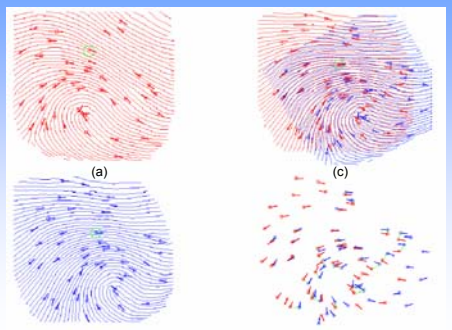


Threshold=5,000

Labeling Minutiae

- The corners found by above procedure include Minutiae, but also other "spurious" points.
- Label each corner as:
  - Core
  - Delta
  - Ridge bifurcations
  - Ridge endings
  - Other
- Build 5-class classifier given training data
- Is it built over rotations/scales? Other?

Minutiae Matching Result



Allowable transformations

- Translation
- Rigid Transformations
- Affine Transformations
- Warps (e.g., thin plate splines)

Alignment
Rigid transformation, including orientation

- Let $(x^d, y^d, \theta^d)^T$ be the reference minutia (based on which the transformation parameters are estimated). The N input minutiae are transformed as follows:

$$\begin{pmatrix} x_i^A \\ y_i^A \\ \theta_i^A \end{pmatrix} = \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta\theta \end{pmatrix} + \begin{pmatrix} \cos\Delta\theta & \sin\Delta\theta & 0 \\ \sin\Delta\theta & -\cos\Delta\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_i - x^d \\ y_i - y^d \\ \theta_i - \theta^d \end{pmatrix}$$

RANSAC

Slides shamelessly taken from Frank Dellaert and Marc Pollefeys and modified

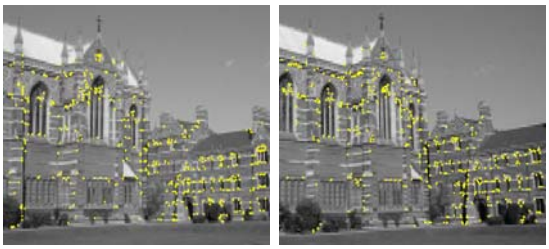
Motivation

- Estimating motion models
- Typically: points in two images
- Candidates:
  - Translation
  - Affine
  - Homography

Mosaicking: Homography

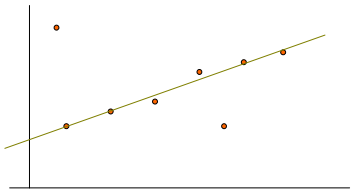
www.cs.cmu.edu/~dellaert/mosaicking

Fundamental Matrix



Simpler Example

- Fitting a straight line



- Inliers
- Outliers

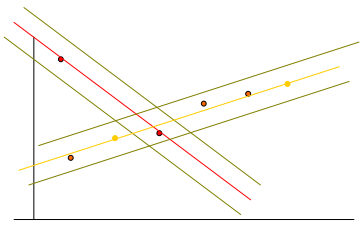
Discard Outliers

- No point with d>t
- RANSAC:
  - RANdom SAmple Consensus
  - Fischler & Bolles 1981
  - Copes with a large proportion of outliers

Main Idea

- Select 2 points at random
- Fit a line
- "Support" = number of inliers
- Line with most inliers wins

Why will this work ?



Best Line has most support

- More support -> better fit

RANSAC

Objective

Robust fit of model to data set S which contains outliers

Algorithm

(i) Randomly select a sample of *s* data points from S and instantiate the model from this subset.

(ii) Determine the set of data points S_i which are within a distance threshold *t* of the model. The set S_i is the **consensus set** of samples and defines the inliers of S.

(iii) If the subset of S_i is greater than some threshold *T*, re-estimate the model using all the points in S_i and terminate

(iv) If the size of S_i is less than *T*, select a new subset and repeat the above.

(v) After *N* trials the largest consensus set S_i is selected, and the model is re-estimated using all the points in the subset S_i

Distance threshold

Choose *t* so probability for inlier is α (e.g. 0.95)

- Often empirically
- Zero-mean Gaussian noise σ then $d_\perp^2$ follows χ_m^2 distribution with m=codimension of model

(dimension+codimension=dimension space)

Codimension	Model	t^2
1	l,F	$3.84\sigma^2$
2	H,P	$5.99\sigma^2$
3	T	$7.81\sigma^2$

How many samples?

Choose *N* so that, with probability *p*, at least one random sample is free from outliers. e.g. *p*=0.99

$$\left(1-(1-e)^s\right)^N = 1-p$$

$$N = \log(1-p)/\log\left(1-(1-e)^s\right)$$

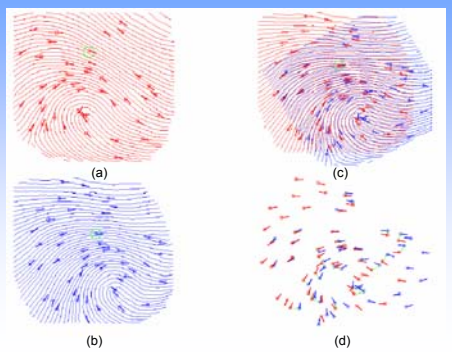
	proportion of outliers *e*						
s	5%	10%	20%	25%	30%	40%	50%
2	2	3	5	6	7	11	17
3	3	4	7	9	11	19	35
4	3	5	9	13	17	34	72
5	4	6	12	17	26	57	146
6	4	7	16	24	37	97	293
7	4	8	20	33	54	163	588
8	5	9	26	44	78	272	1177

Acceptable consensus set?

- Typically, terminate when inlier ratio reaches expected ratio of inliers

$$T = (1-e)n$$

Minutiae Matching Result



Minutiae Extraction Failure

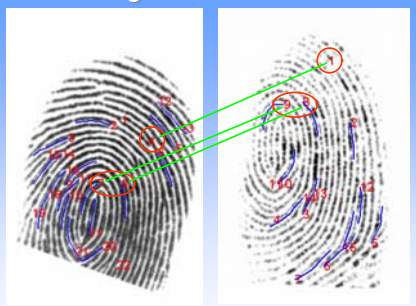
A

B

True Minutiae Matches: A1→B3, A18→B9, A19→B7
A1, B9 and B7 were detected, but the associated ridges were not detected because they are close to the boundary

Alignment Failure

A



B

True Minutiae Matches: A7→B9, A8→B8, A4→B1
A7→B9 and A8→B8 pairs have ridge points; however, there exists a false alignment that results in more than three matches

Matching Failure

A

B

No. of matching minutiae identified by the matcher = 10
No. of minutiae in A = 38; No. of minutiae in B = 34
Spurious minutiae and large deformation leads to small score