Feature Extraction

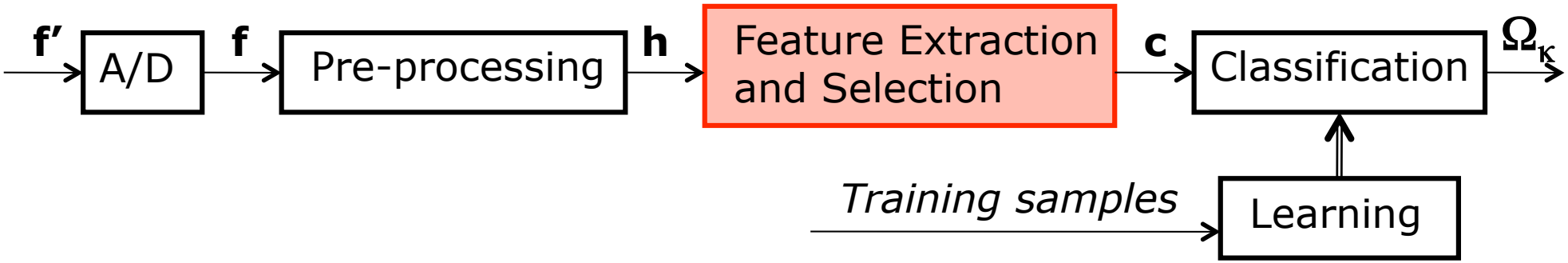
Overview and Fourier Transform

Dr. Elli Angelopoulou

Lehrstuhl für Mustererkennung (Informatik 5)

Friedrich-Alexander-Universität Erlangen-Nürnberg

Pattern Recognition Pipeline



- The input to our feature extraction/computation is the pre-processed image, $h = T\{f\}$
- In feature extraction we compute a numerical characteristic vector $\vec{c} \in R^N$, which facilitates the subsequent classification task.
- In feature selection we select the best subset of features to create a lower dimensional characteristic vector $\vec{c}\,' \in R^M, \quad M < N$.

Motivation

- Why should we use features? Why not the entire signal?

1. Classes may not be as easily separable in the original signal space.

Motivation

2. Curse of dimensionality

- It is often the case that when our signal is high-dimensional, (e.g. fused medical modalities, multispectral images, etc.) we may end up having *too much data*.
- In high dimensions our intuition and topological/ spatial comprehension fails.
- We usually get non-linear behaviors.
- The notion of a neighbor is not clearly defined. Closely related topic is the employment of a suitable distance metric.

Curse of Dimensionality – Nearest Neighbor

- In the k-Nearest Neighbor algorithm, given a new sample *f* examine its k nearest neighbors (the k existing samples that are most similar to *f*). Assign to the new sample *f* the class of the majority of the nearest neighbors.
- Works fine in low dimensions, i.e. n=2,3,4
- But in high dimensions:

a) If we are to examine all directions of the neighborhood we need to examine 2^n samples.

b) The hyper-structures force one to examine a large percentage of the data to compute the neighborhood (not uniformly spread data)

Curse of Dimensionality - continued

- Is it all so bad? Should we even bother then with high-dimensional data collection?
- There is no such thing as too much data.
- There is a bright side:

a) much of the data is irrelevant.

b) data may be correlated, so the true dimensionality of the data may be lower dimensional.

- Feature extraction allows for:

a) a compressed representation.

b) isolating the important/relevant information.

Example: Different Color Spaces

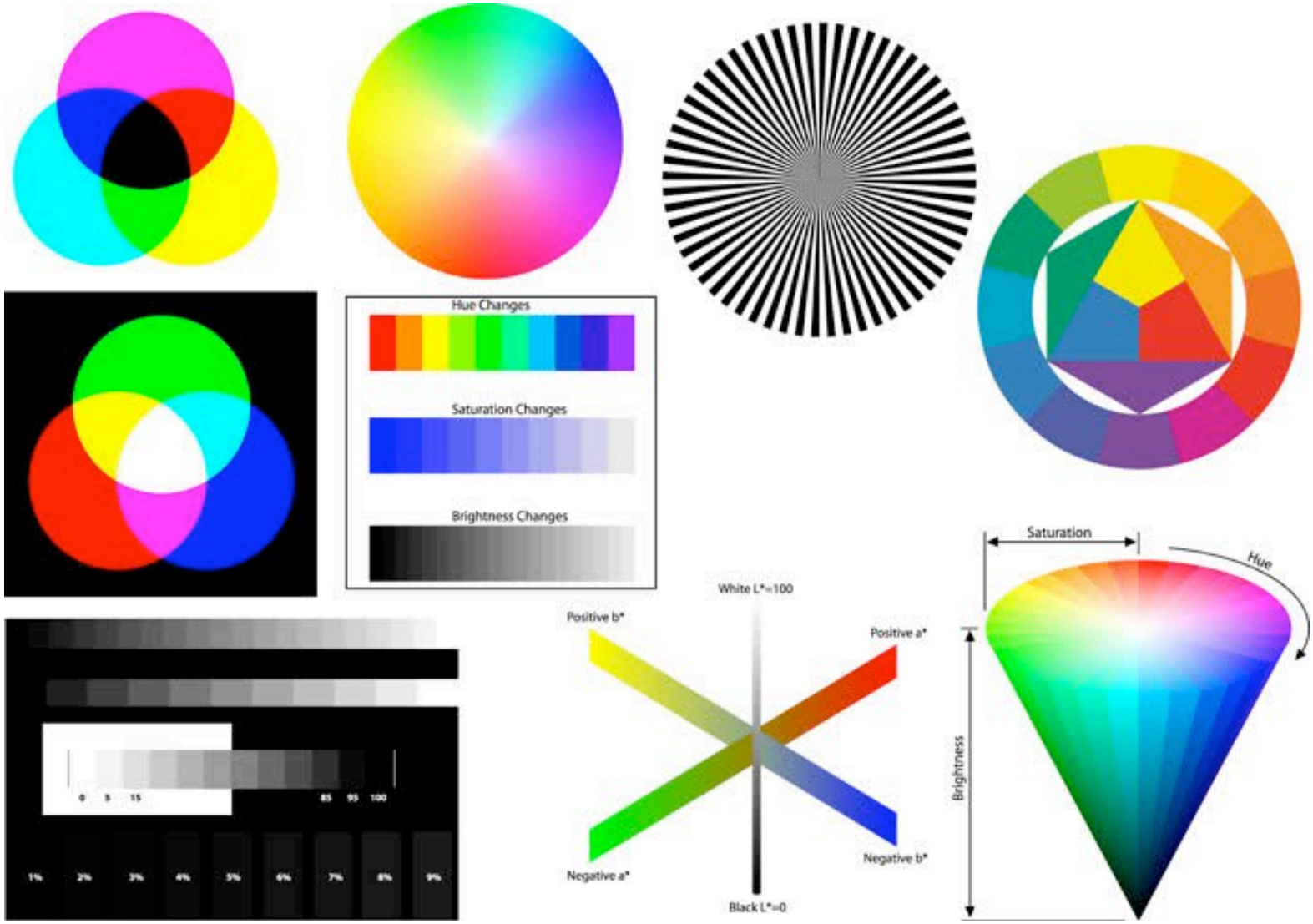


Image courtesy of http://down0day.blogspot.com/2009/06/rgb-cmyk-color-guides-with-print.html

Heuristic vs Analytic Feature Extraction

- There are two distinct methods for extracting features:

1. **Heuristic methods**: Their use is based on experience (trial and error), empirical measurements, discussion with experts etc. They are still based on mathematics.

2. **Analytic methods**: They define an objective function for the quality that the resulting features should satisfy. Features are computed by solving an optimization problem.

Heuristic Methods

- There are different heuristic feature extraction methods:

1. Projection to orthogonal bases
   - Fourier Transform
   - Walsh/Hadamard Transform
   - Haar Transform
2. Linear Predictive Coding
3. Geometric Moments
4. Feature Extraction via Filtering
5. Wavelets
6. Edges
7. Statistics-based
8. Color
9. Geometric measurements …

Orthogonal Basis

- Let $\vec{f}$ be the signal after pre-processing.
- Assume, that the vector space of $\vec{f}$ is spanned by the orthonormal vectors $\vec{\varphi}_1,\vec{\varphi}_2,\ldots,\vec{\varphi}_M$ such that:

$$\vec{\varphi}_i \bullet \vec{\varphi}_j = \begin{cases} 0 & \text{if } i \neq j \\ 1 & \text{if } i = j \end{cases}$$

- Since $\vec{\varphi}_1,\vec{\varphi}_2,\ldots,\vec{\varphi}_M$ spans the vector space of $\vec{f}$ we can rewrite $\vec{f}$ as:

$$\vec{f} = \sum_{i=1}^{M} c_i \vec{\varphi}_i = \Phi \vec{c} \quad \text{where} \quad \Phi = \left[\vec{\varphi}_1,\vec{\varphi}_2,\ldots,\vec{\varphi}_M\right] \quad \vec{c} = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_M \end{bmatrix}$$

Orthogonal Basis -continued

$$\vec{f} = \sum_{i=1}^{M} c_i \vec{\varphi}_i = \Phi \vec{c}$$

- So the coefficients c_i and the vectors $\vec{\varphi}_i$ can give us a representation of the original signal $\vec{f}$ in another form (space) where:

a) hopefully the classes will be better separated.

b) the dimensionality of the new orthonormal space will be lower than that of $\vec{f}$:

$$\dim(\vec{f}) = N$$

$$\dim(\vec{c}) = M$$

hopefully $M << N$

Information Loss

- If $M = N$, then there is no information loss
- But when $M << N$, we incur some information loss and then:

$$\hat{\vec{f}} = \Phi\vec{c} \quad \text{where} \; \hat{\vec{f}} \neq \vec{f}$$

- Goal: Minimize the information loss.
- Minimize: $\varepsilon = \left(\vec{f} - \hat{\vec{f}}\right)^T \left(\vec{f} - \hat{\vec{f}}\right)$

$$\varepsilon = \left(\vec{f} - \Phi\vec{c}\right)^T \left(\vec{f} - \Phi\vec{c}\right) = \vec{f}^T\vec{f} - \left(\Phi\vec{c}\right)^T\vec{f} - \vec{f}^T\left(\Phi\vec{c}\right) + \left(\Phi\vec{c}\right)^T\left(\Phi\vec{c}\right)$$

$$\varepsilon = \vec{f}^T\vec{f} - 2\left(\Phi\vec{c}\right)^T\vec{f} + \vec{c}^T\Phi^T\Phi\vec{c}$$

Minimize Information Loss

$$\varepsilon = \vec{f}^T \vec{f} - 2\left(\Phi\vec{c}\right)^T \vec{f} + \vec{c}^T \Phi^T \Phi\vec{c}$$

- We want to find the set of coefficients $\vec{c}$ that minimizes the information loss.
- How?
- Differentiation

$$\frac{\partial\varepsilon}{\partial\vec{c}} = -2\Phi^T \vec{f} + 2\Phi^T \Phi\vec{c} = 0 \Rightarrow$$

$$\Phi^T \vec{f} = \Phi^T \Phi\vec{c} \Rightarrow$$

$$\boxed{\vec{c} = \Phi^T \vec{f}}$$

$\vec{c}$ is the feature vector

Choice of Orthogonal Bases

- The bases should be at least orthogonal, i.e.

$$\vec{\varphi}_i \bullet \vec{\varphi}_j = 0, \quad \forall i \neq j$$

- We prefer orthonormal bases because:
  - Computing $\vec{c}$ is easier.
  - The feature vector $\vec{c}$ is often more intuitive.
- There are several different options for orthonormal basis functions:
  - Fourier series,
  - Walsh functions,
  - Haar functions,
  - Hermite functions,
  - Legendre polynomials,…

Fourier Basis Functions

- The Fourier transform describes a way of decomposing a function into a sum of orthogonal sinusoidal functions.
- The standard basis functions used for the Fourier transform are:

$$\sin(2\pi\omega x)$$
$$\cos(2\pi\omega x) \qquad \omega \in R$$

- As the frequency ω varies over the set of all real numbers we get an infinite collection of basis functions.

Fourier Basis Functions - continued

- Recall that: $e^{2\pi j\omega x} = \cos(2\pi\omega x) + j\sin(2\pi\omega x)$
- By using the exponential form of the basis functions we can represent both real and complex valued functions (signals) by their Fourier transform.
- Furthermore, any two basis functions of different frequencies are orthogonal to each other.
- For example, consider the real case of only the cosine terms:

$$\int \cos(2\pi\omega_1 x)\cos(2\pi\omega_2 x)dx = 0, \quad \forall\, \omega_1 \neq \omega_2$$

because the product of cosines is a cosine function itself.

Fourier Transform

- Given a discrete signal $\vec{f}$ we can project it onto the Fourier basis functions $e^{-2\pi j\omega x}$ and get the Fourier transform:

$$F_\omega = \sum_x f_x e^{-2\pi j\omega x}$$

- We can choose specific frequencies as follows:

$$F_\nu = \sum_{i=0}^{M-1} f_i e^{-2\pi j \frac{i\nu}{M}}$$

- Let $W_M = e^{-2\pi j \frac{1}{M}}$
- Then

$$F_\nu = \sum_{i=0}^{M-1} f_i \left(W_M\right)^{i\nu}$$

Fourier Transform as a Feature Vector

- So we have the following representation based on the Fourier transform

$$F_\nu = \sum_{i=0}^{M-1} f_i \left(W_M\right)^{i\nu}$$

- If we use: $\vec{\varphi}_\nu = \left(\left(W_M\right)^0, \left(W_M\right)^\nu, \left(W_M\right)^{2\nu}, \ldots \left(W_M\right)^{(M-1)\nu}\right)$

- We get: $F = \Phi^T \vec{f}$ $\vec{f}$

  which looks very similar to our feature vector computation of

$$\vec{c} = \Phi^T \vec{f}$$

FT as a Feature Vector - continued

- Now we have the right format, we have to check if the basis vectors are orthogonal (orthonormal).
- Since we have a complex basis, we use the conjugate transpose when we compute the inner product:

$$\vec{\varphi}_i \bullet \vec{\varphi}_j^{\,*} = \begin{cases} 0 & \text{if } i \neq j \\ 1 & \text{if } i = j \end{cases}$$

- Conclusion: We can use the FT and use the Fourier coefficients as features.

Translation Invariance

- The Fourier-derived coefficients have an additional property. Their absolute value is invariant to translation.

$$F(\omega) = \int_{-\infty}^{\infty} f(x) e^{-j\omega x} dx$$

- For a shifted signal:

$$G(\omega) = \int_{-\infty}^{\infty} f(x+k) e^{-j\omega x} dx = \int_{-\infty}^{\infty} f(u) e^{-j\omega(u-k)} du$$

$$= e^{j\omega k} \int_{-\infty}^{\infty} f(u) e^{-j\omega u} du = e^{j\omega k} F(\omega)$$

Translation Invariance - continued

- So we have $F(\omega) = \int_{-\infty}^{\infty} f(x) e^{-j\omega x} dx$
- And for a shifted signal: $G(\omega) = e^{j\omega k} F(\omega)$
- By taking the absolute value:

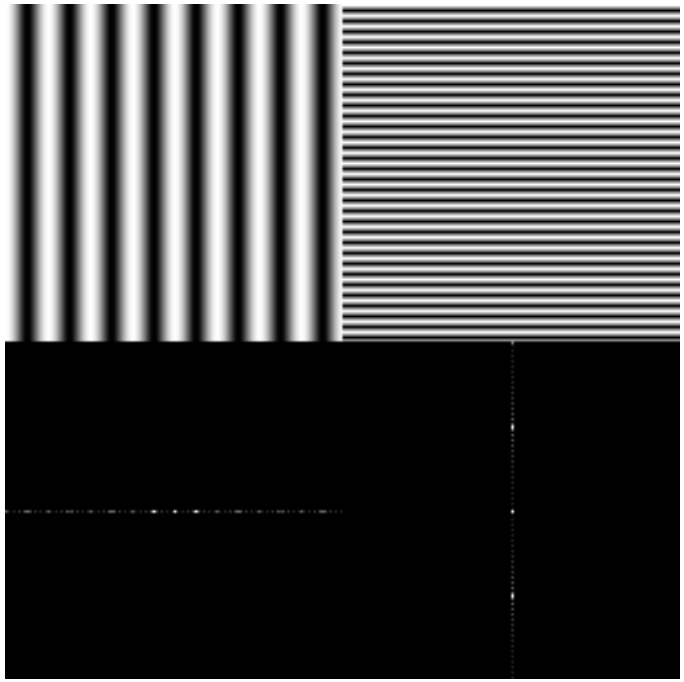
$$\left|G(\omega)\right| = \left|e^{j\omega k}\right| \left|F(\omega)\right| = \left|F(\omega)\right| \quad \text{since} \left|e^{j\omega k}\right| = 1$$

- So given a particular signal (or a particular object), independent of where the object is located, we get the same absolute value of its FT. Easier to detect!!!
- But we don't like to deal with complex numbers, so we often use:

$$c_\nu = \left|F_\nu\right|^2 = F_\nu \bullet F_\nu^*$$

FT Simple Example

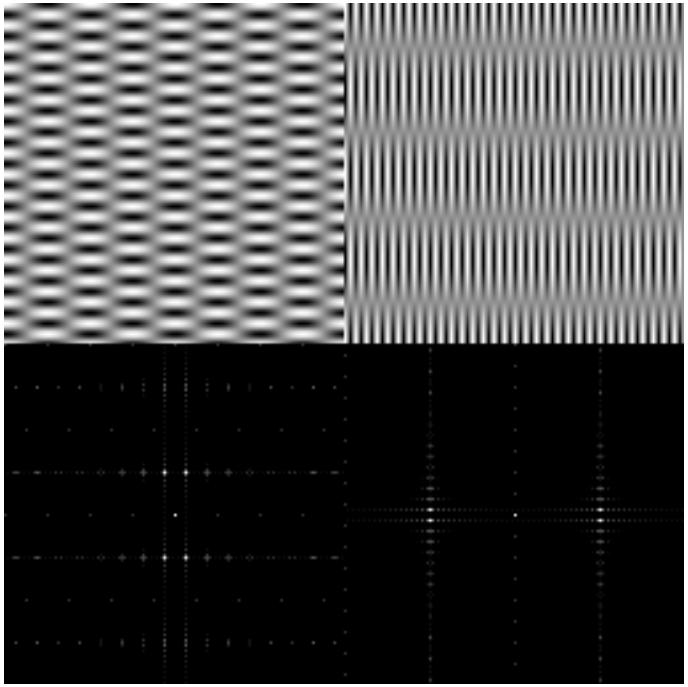
- The FT tries to represent all images as a summation of “cosine images”.
- Images that are pure cosines have very simple FTs.



- The dot at the center represents the (0,0) frequency term or average value of the image.
- Images usually have a large average value (like 128) and lots of low frequency information so FT images usually have a bright blob of components near the center.
- Notice that each FT image just has a single component (besides the average (0,0) value), represented by 2 bright spots symmetrically placed about the center of the FT image.

FT Simple Example 2

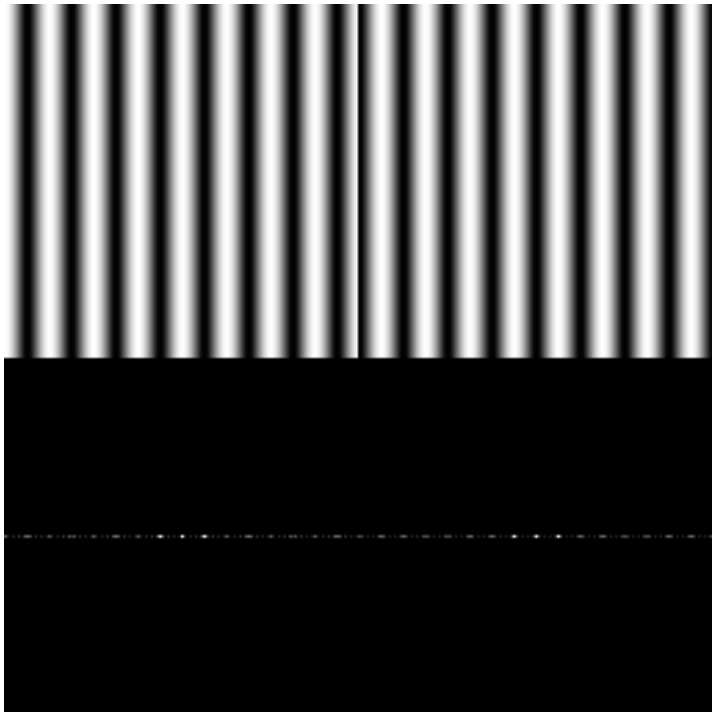
- The following images are of 2D cosines with both horizontal and vertical components.
- The left one has 4 cycles horizontally and 16 cycles vertically.
- The right one has 32 cycles horizontally and 2 cycles vertically.



- For all REAL images, the FT is symmetrical about the origin: the 1st and 3rd quadrants are the same and the 2nd and 4th quadrants are the same.

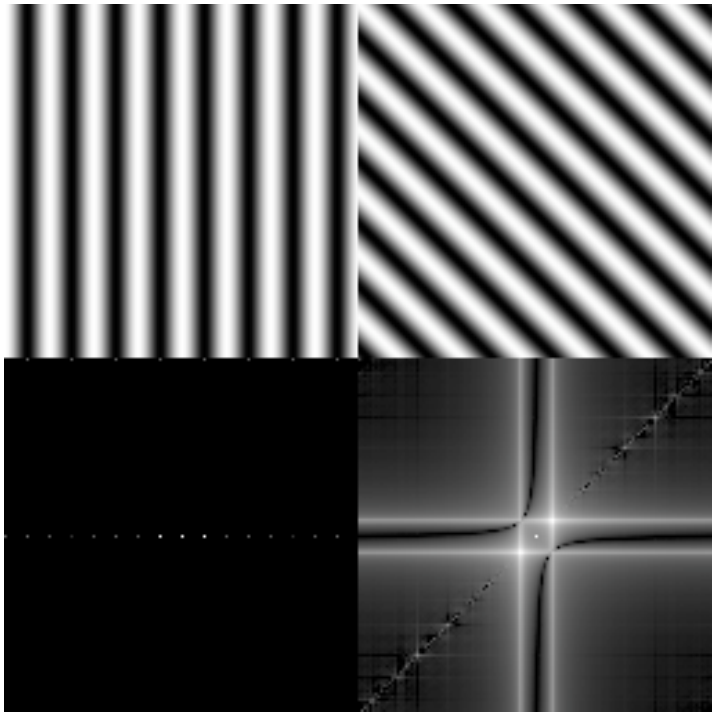
FT Shift Invariance Example

- Almost all the FT images show just the magnitude of the FT.



- Notice that we have the same sinusoidal just shifted. The FT is the same.

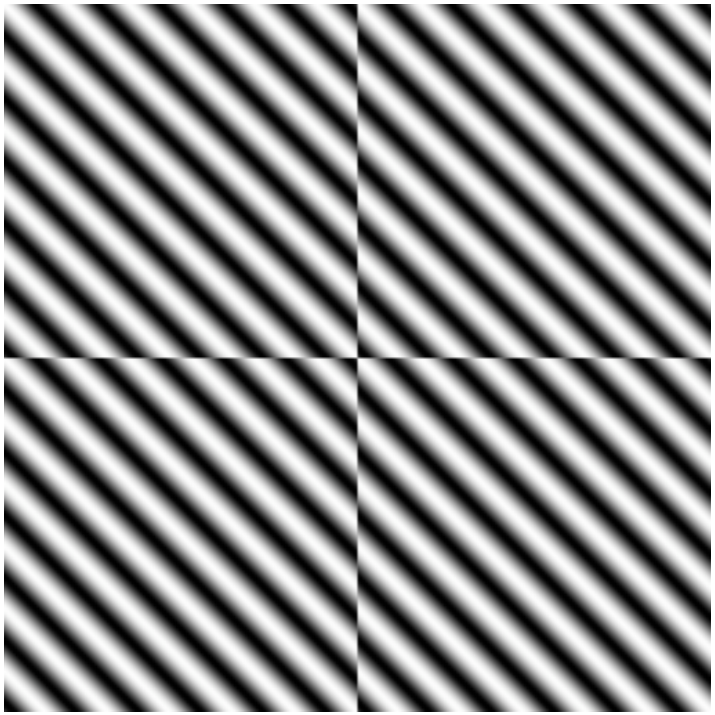
FT Rotation Example



- The rotated cosine has a complicated FT with strong diagonal components and a strong "plus sign" shaped horizontal and vertical components.

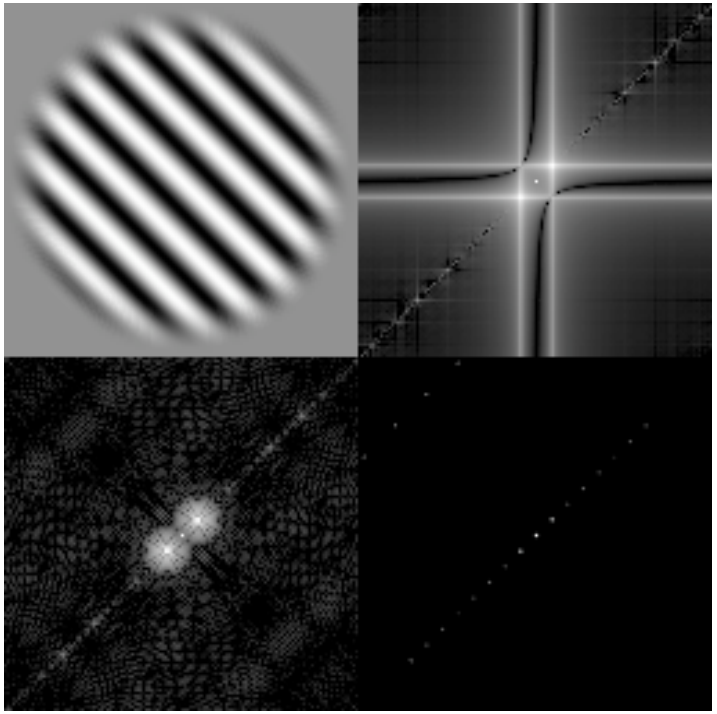
FT Rotation Example -continued

- Why the complex effect?
- The FT always treats an image as if it were part of a periodically replicated array of identical images extending horizontally and vertically to infinity.
- If we extend this rotated cosine image, we get strong edge effects.



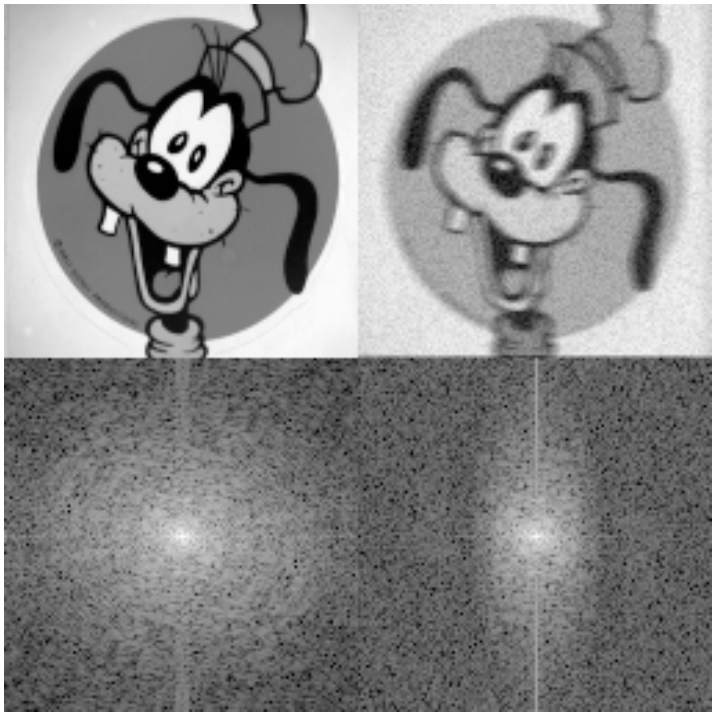
FT Rotation Example - windowing

- Solution: Windowing



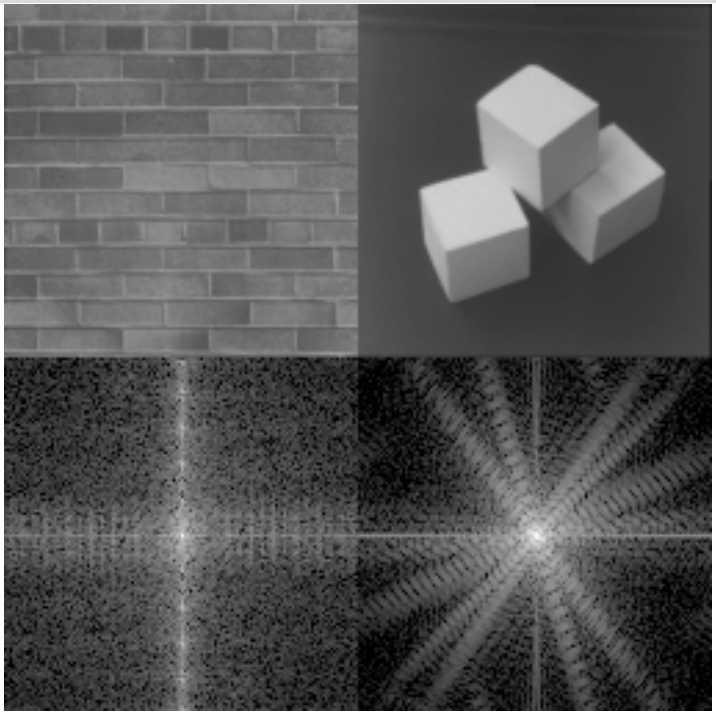
- The windowed image is shown in the upper left.
- Its FT is shown in the lower left.
- The non-windowed FT is shown in the upper right.
- The actual, true FT of a rotated cosine is shown in the lower right.

More FT Examples



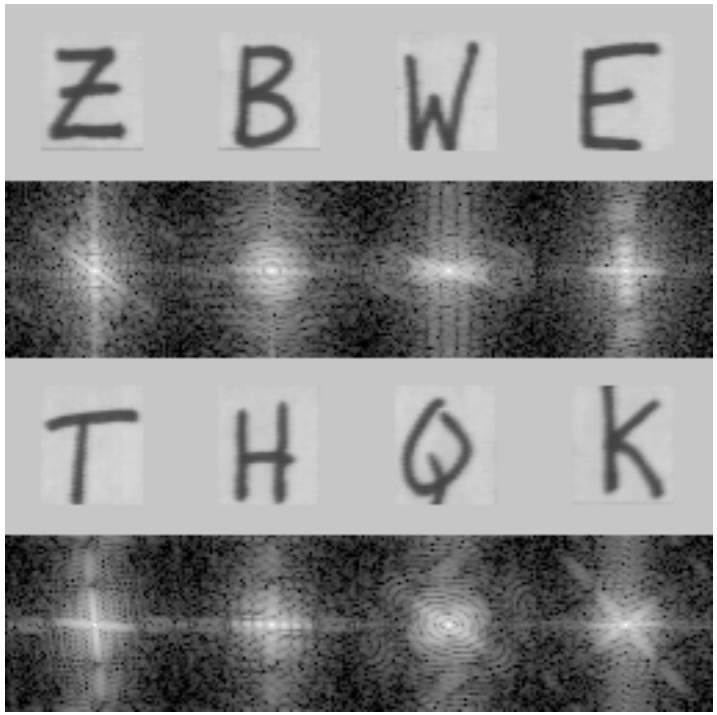
- Both suffer from edge effects as evidenced by the strong vertical line through the center.
- The degraded goofy has significantly attenuated high frequencies in the horizontal direction. This is due to the fact that the degraded image was formed by smoothing only in the horizontal direction.

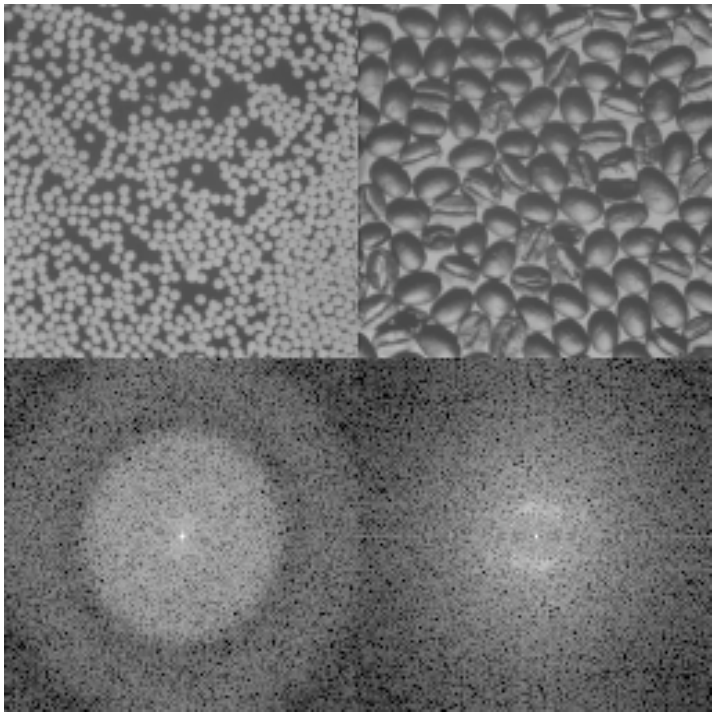
More FT Examples 2



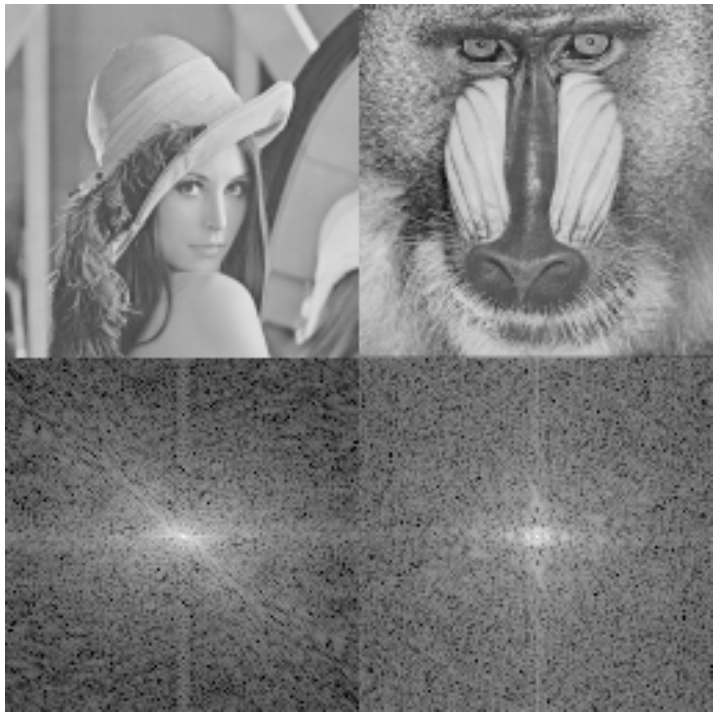
- Note the strong periodic component, especially in the vertical direction for the bricks image.
- In the blocks image there is a bright line going to high frequencies perpendicular to the strong edges in the image. Anytime an image has a sharp edge the gray values must change very rapidly. It takes lots of high frequency power to follow such an edge so there is usually such a line in its magnitude spectrum.

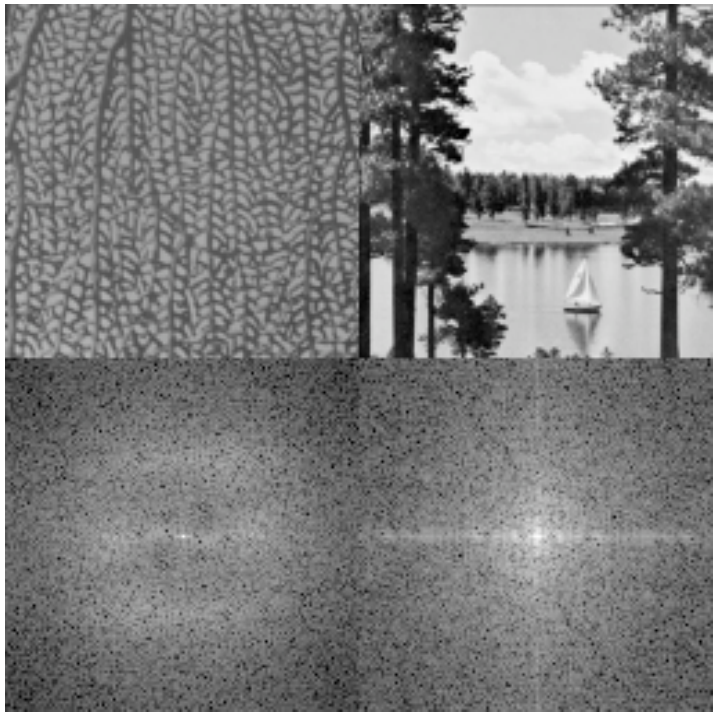
Even More FT Examples





Yet More FT Examples





A Pattern Recognition Example

- The following slides show a palmprint recognition example.
- The input signal is a greyscale image of a palm.
- After pre-processing, the data is mapped to a feature space based on the Fourier Transform.
- The example and all the presented images are from the paper:

  Wenxin Li, David Zhang, and Zhuoqun Xu. "Palmprint Identification by Fourier Transform." *International Journal of Pattern Recognition and Artificial Intelligence*, Vol. 16, No. 4, 2002, pp. 417-432.

PR Pipeline for Palmprint Recognition

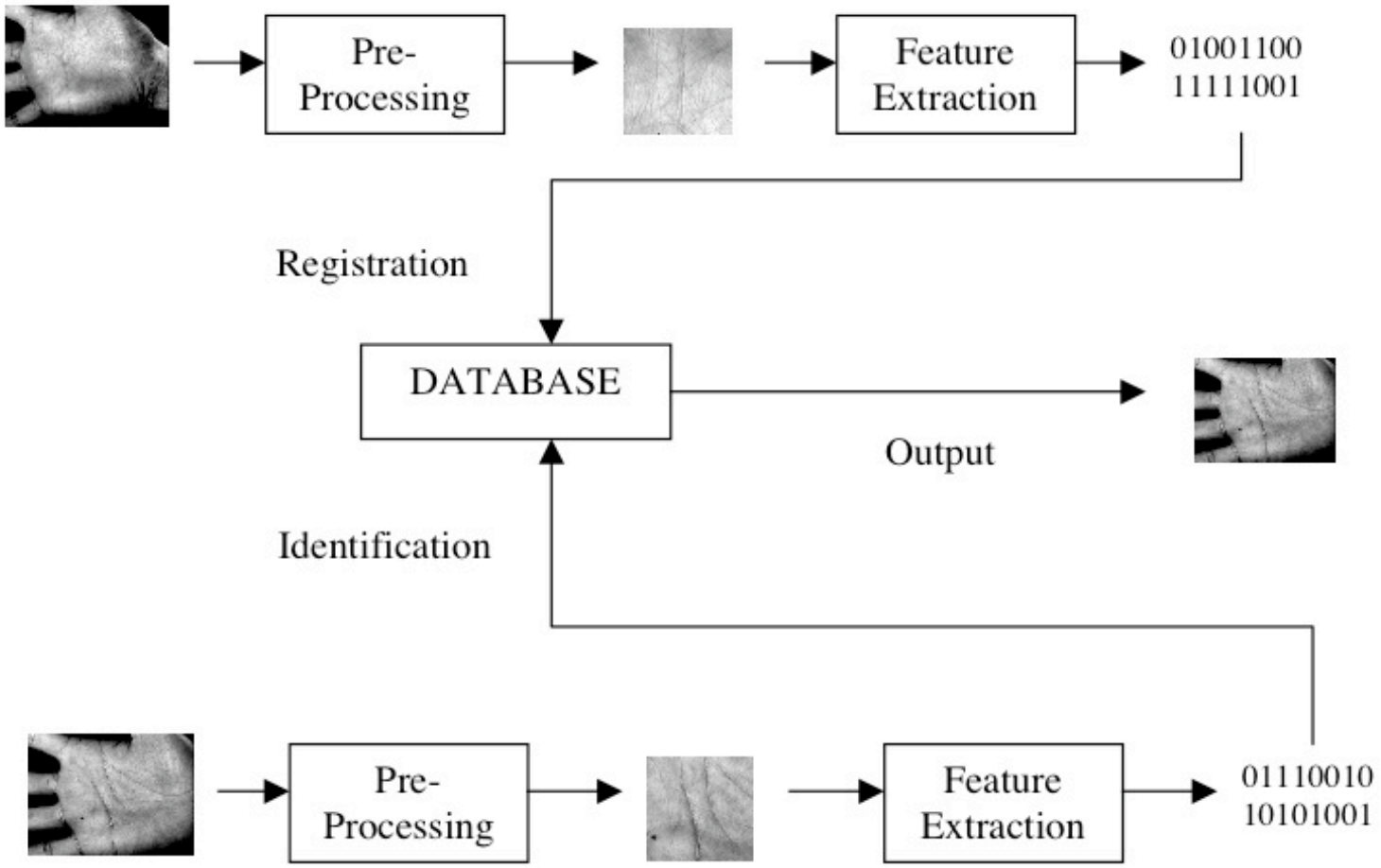
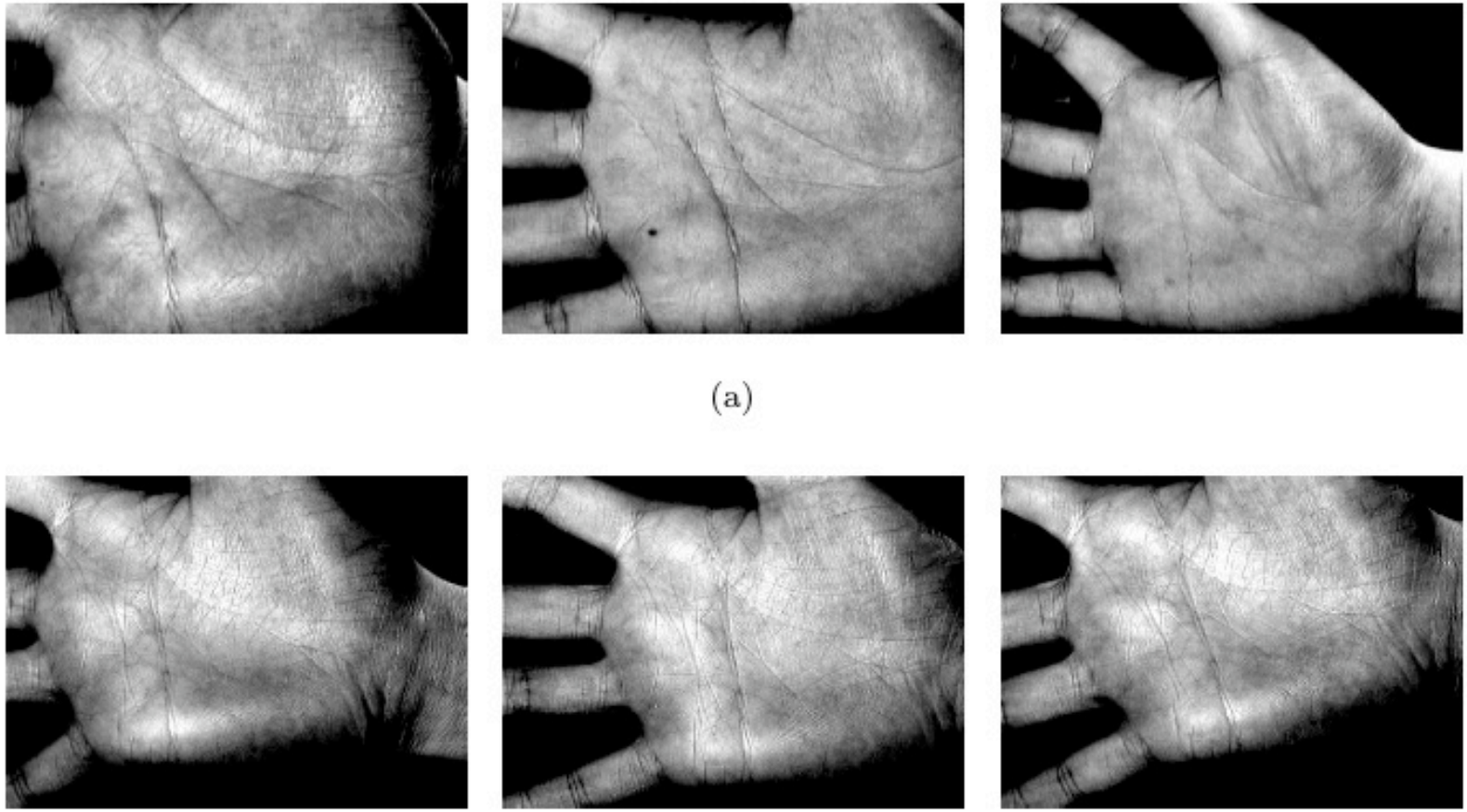


Fig. 1. The general process of palmprint identification.

The Need for Normalization



(a)

(b)

Fig. 2. Palmprint samples with some rotation, shift and different sizes. (a) Palmprint samples with different sizes. (b) Palmprint samples from the same palm with some rotation and shift.

Palmprint Alignment

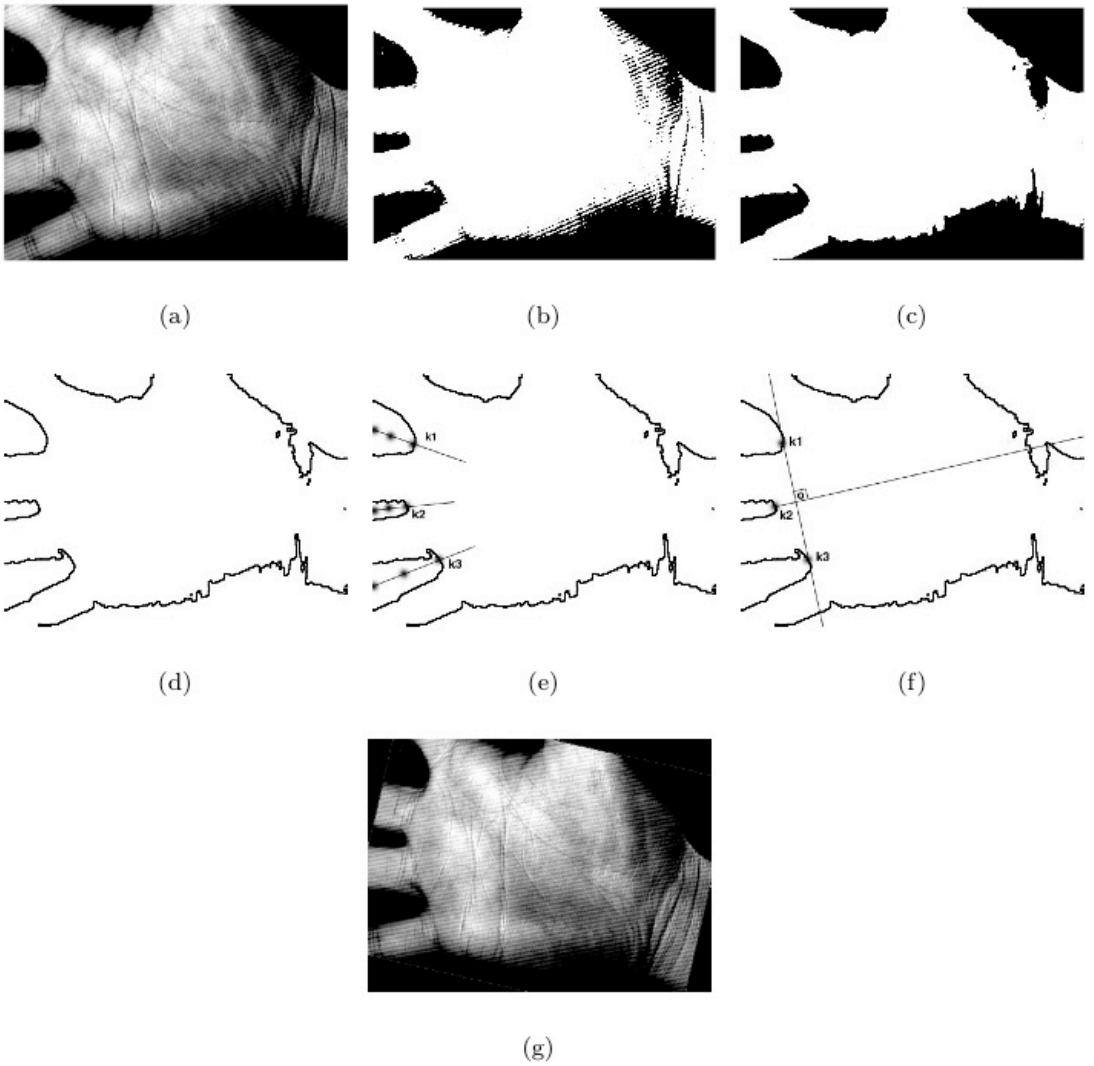


Fig. 3. The process of palmprint alignment. (a) Original image. (b) Binary image. (c) Smoothed binary image. (d) Boundary tracing. (e) Key points determination. (f) Coordination system. (g) Rotated image.

Examples of Palmprint Alignment

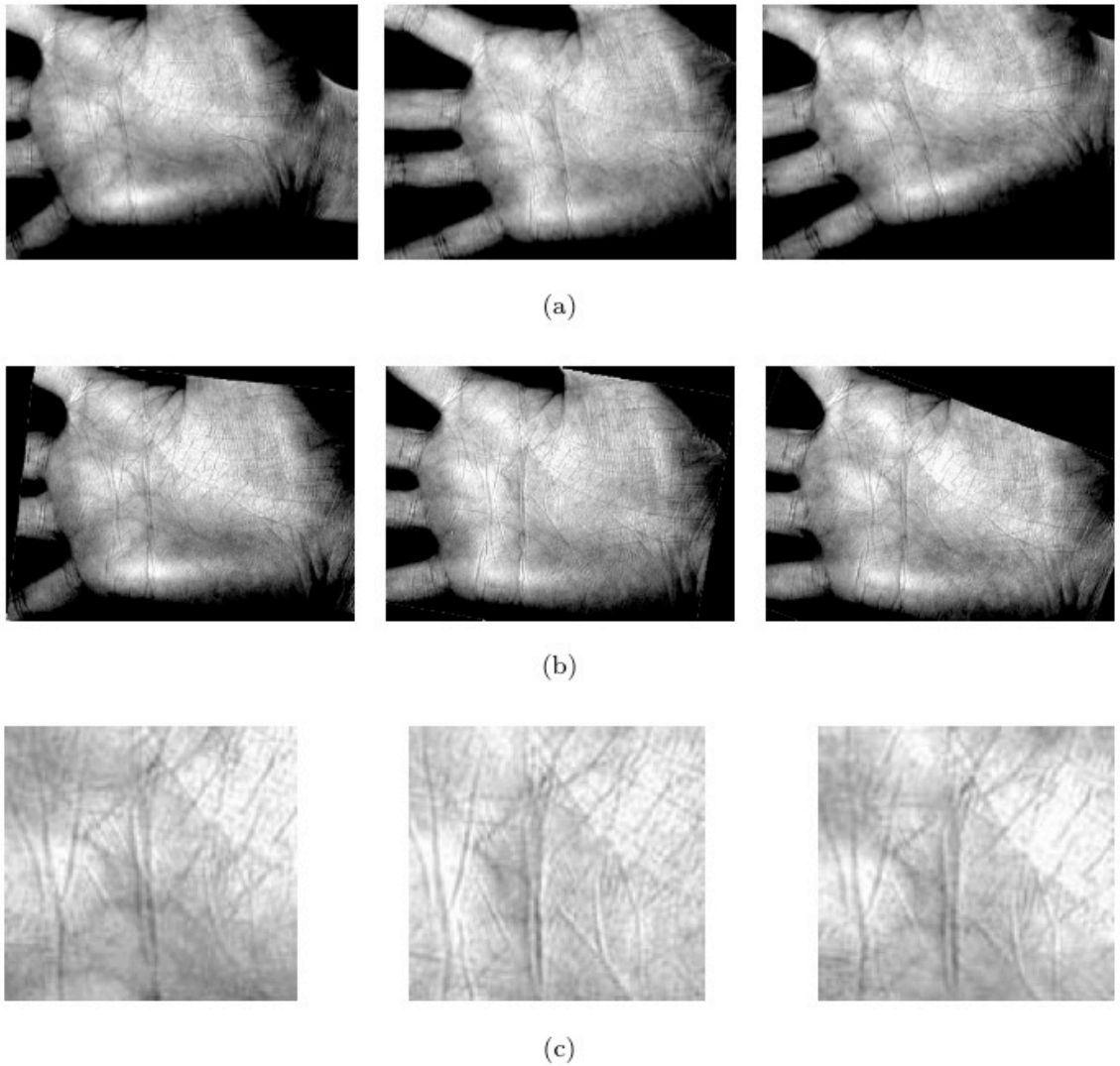
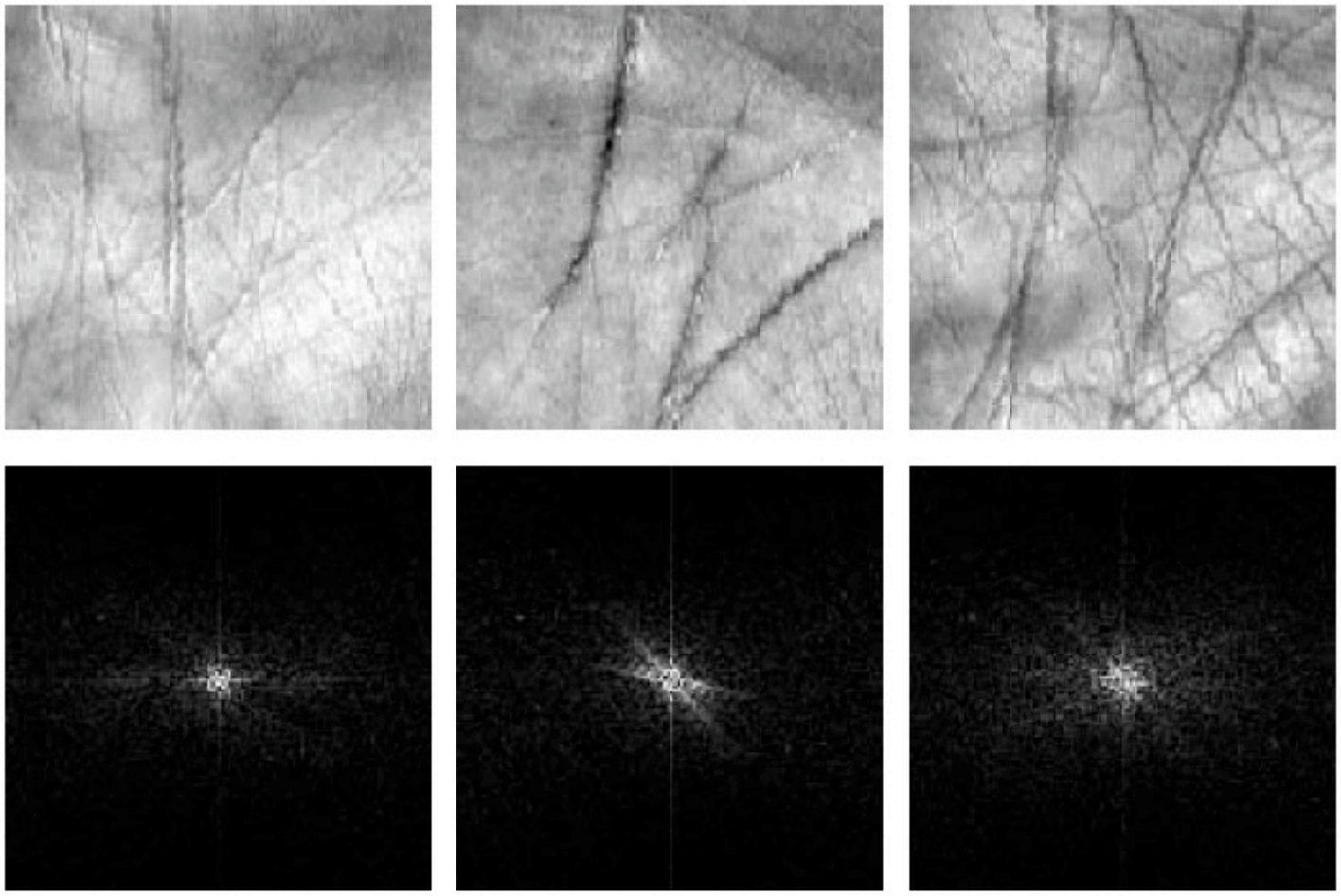


Fig. 4. Contrast to palmprints before and after alignment and their extracted subimages. (a) Three samples from the same palm with different directions and locations. (b) Alignment results. (c) The extracted subimages.

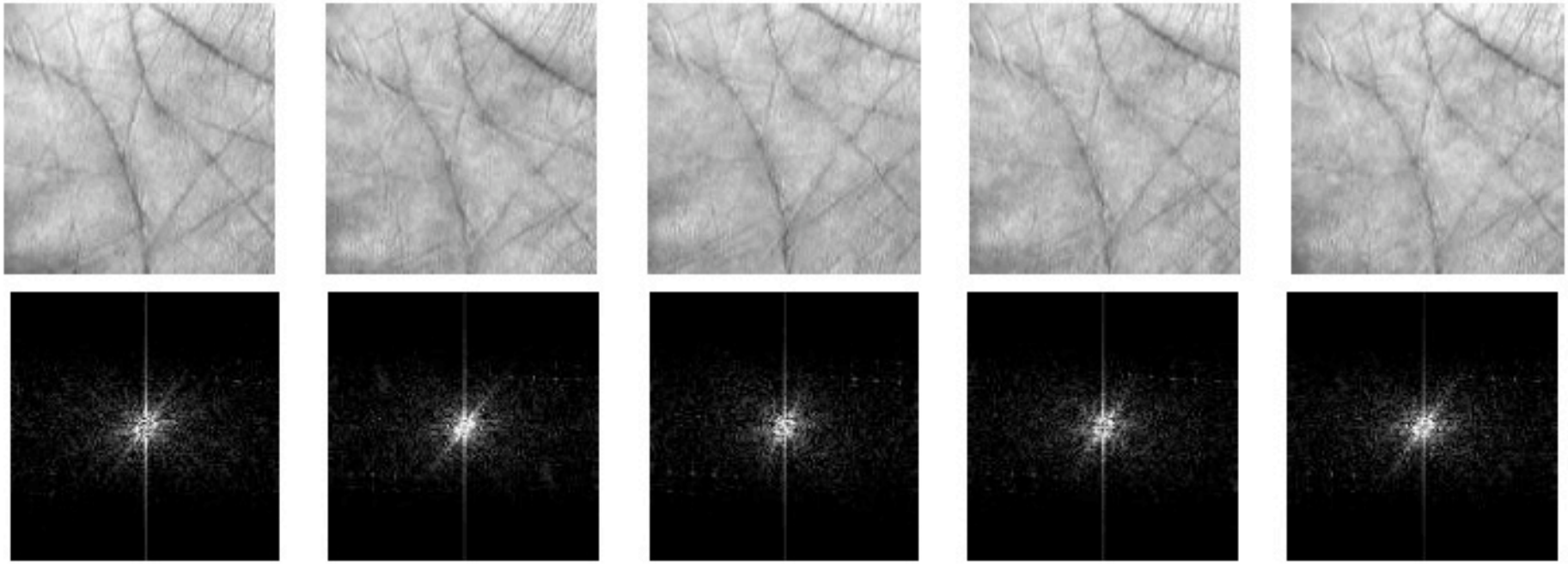
Fourier Transform of Palmprint Images



(a) (b) (c)

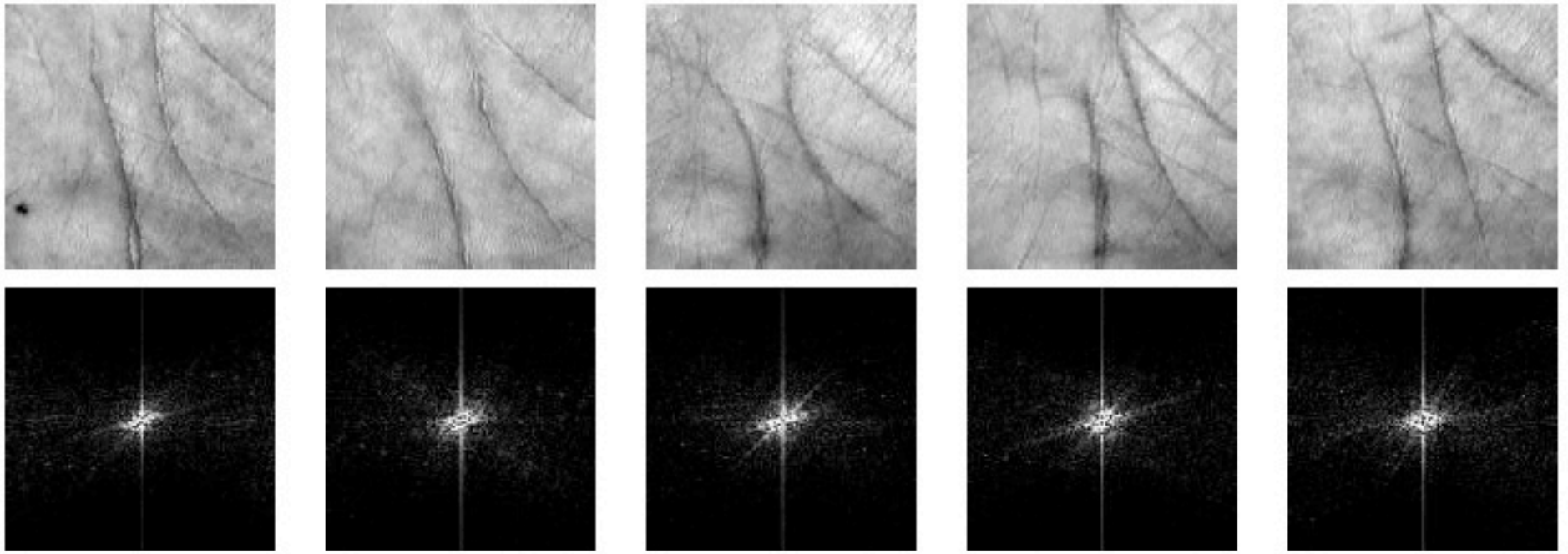
Fig. 6. Different palmprints and their corresponding frequency domain images. (a) Palmprint without strong creases. (b) Palmprint with two clear and strong creases. (c) Palmprint full of strong creases.

Examples of the Same Palm



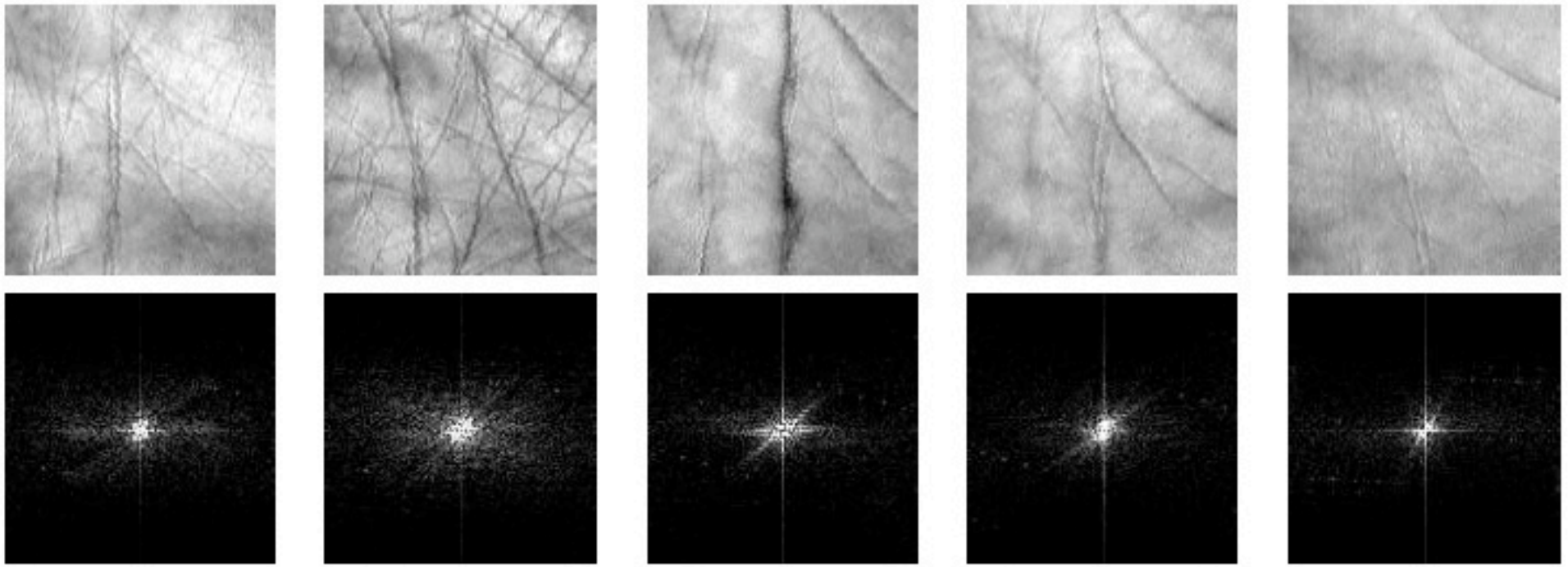
A-1 A-2 A-3 A-4 A-5

Examples of Similar Palms



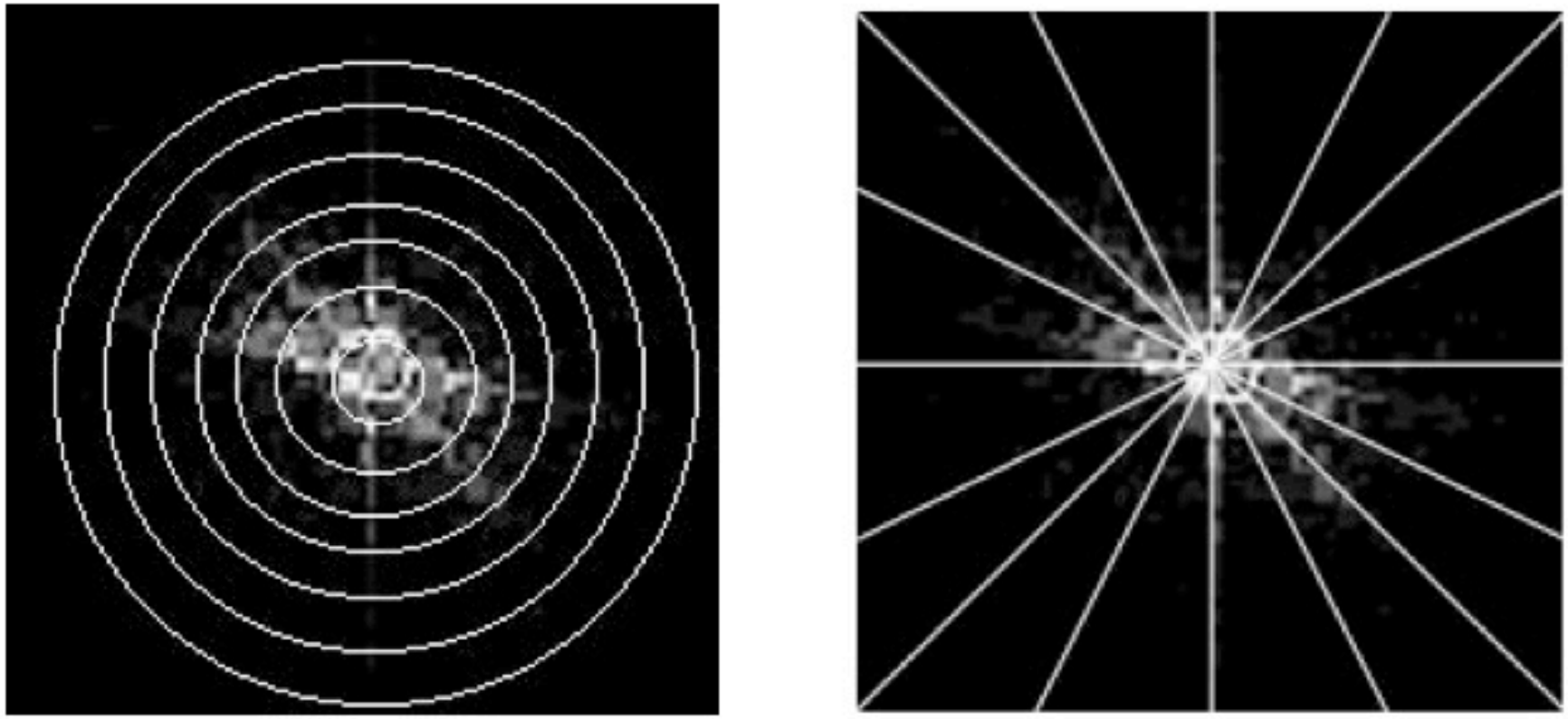
(b)

Examples of Distinct Palms



(c)

Analysis of the Images in FT Space



(a) (b)

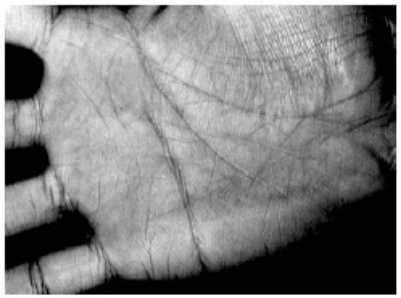
Fig. 8. Palmprint feature representation: (a) R feature and (b) θ feature.

Closest Matches (Nearest Neighbor)

(a)

(b)

(c)



(d)

(e)

(f)

(a) Input palmprint that needs to be identified.
(b)-(f) Candidate matches ranked from closer to more dissimilar one.

Palmprint Classification Results

Number of Subjects	500 (6 palmprints per subject)
Training data	500 palmprints (1 per subject)
Test data	2500 palmprints
Correctly identified	2387 palmprints
Recognition rate	95.48%

Average computation time per palmprint is 2 sec.

Sources

1. FT images are courtesy of John M. Brayer http://www.cs.unm.edu/~brayer/vision/fourier.html