UNIT II

Image Transforms

2-D FFT , Properties.

Walsh transform, Hadamard Transform,

Discrete cosine Transform,

Haar transform, Slant transform,

Hotelling transform.

Textbooks

R. C. Gonzalez, R. E. Woods, *Digital Image Processing,*

Anil K. Jain *Fundamentals of Digital Image Processing*

1-D DISCRETE COSINE TRANSFORM DCT

$$C(u) = a(u)\sum_{x=0}^{N-1} f(x)\cos\left[\frac{(2x+1)u\pi}{2N}\right]$$

$$u = 0,1,\ldots,N-1$$

$$a(u) = \begin{cases} \sqrt{\frac{1}{N}} & u = 0 \\ \sqrt{\frac{2}{N}} & u = 1,\ldots,N-1 \end{cases}$$

1-D INVERSE DISCRETE COSINE TRANSFORM
IDCT

$$f(x) = \sum_{u=0}^{N-1} a(u)C(u)\cos\left[\frac{(2x+1)u\pi}{2N}\right]$$

2-D DISCRETE COSINE TRANSFORM
DCT

$$C(u,v) = a(u)a(v)\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)\cos\left[\frac{(2x+1)u\pi}{2N}\right]\cos\left[\frac{(2y+1)v\pi}{2N}\right]$$

$$f(x,y) = \sum_{u=0}^{N-1}\sum_{v=0}^{N-1} a(u)a(v)C(u,v)\cos\left[\frac{(2x+1)u\pi}{2N}\right]\cos\left[\frac{(2y+1)v\pi}{2N}\right]$$

$$u,v = 0,1,\ldots,N-1$$

- Hotelling transform
- Eigen vector transform
- Principal component Transform
- Karhunen- Loeve Transform (KL Transform)

Hotelling Transform

$$\underline{x}_1 = \begin{bmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ x_n \end{bmatrix}, \ldots\ldots\ldots, \underline{x}_M$$

M data points

Mean:

$$\underline{m}_x = E\{\underline{x}\} \approx \frac{1}{M}\sum_{k=1}^{M} x_k$$

Covariance:

$$\underline{\underline{C}}_x = E\{(\underline{x} - \underline{m}_x)(\underline{x} - \underline{m}_x)^T\} \approx \frac{1}{M}\sum_{k=1}^{M} \underline{x}_k \underline{x}_k^T - \underline{m}_k \underline{m}_k^T$$

Hotelling Transform: $\underline{y} = \underline{\underline{A}}(\underline{x} - \underline{m}_x)$

The rows of matrix A are the eigen vectors of the covarience matrix arranged in descending order (The first row corresponds to the eigen vector corresponding to the largest eigen value of C, ...)

Example

- Consider 4 column vectors (M=4)

x1	x2	x3	x4
0	1	1	1
0	0	1	0
0	0	0	1

- Mean vector m_x = $\frac{1}{4}\begin{pmatrix}3\\1\\1\end{pmatrix}$

Covariance matrix

Cx =

$$\frac{1}{16}\begin{pmatrix} 3 & 1 & 1 \\ 1 & 3 & -1 \\ 1 & -1 & 3 \end{pmatrix}$$

- Eigen values of Cx
- 1,4,4

- Eigen vectors of Cx
- -0.5774 0.5774 0.5774
- -0.1543 -0.7715 0.6172
- 0.8018 0.2673 0.5345

- Hotelling Transform:

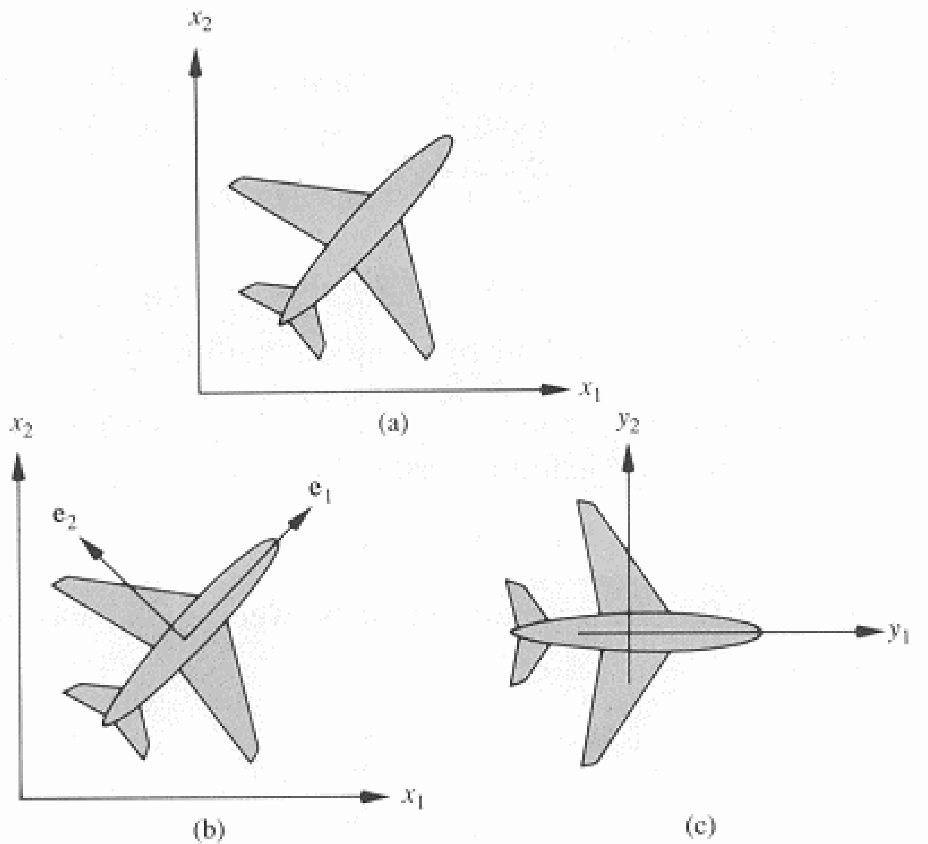
$$\underline{y} = \underline{\underline{A}}(\underline{x} - \underline{m_x})$$

A			mx	x1	x2	x3	x4
0.80	0.27	0.53	0.75	0	1	1	1
-0.15	-0.77	0.62	0.25	0	0	1	0
-0.58	0.58	0.58	0.25	0	0	0	1
				x1-mx	x2-mx	x3-mx	x4-mx
				-0.75	0.25	0.25	0.25
				-0.25	-0.25	0.75	-0.25
				-0.25	-0.25	-0.25	0.75
				y1	y2	y3	y4
				-0.80	0.00	0.27	0.53
				0.15	0.00	-0.77	0.62
				0.14	-0.43	0.14	0.14
				my			
				0			
				0			
				0			

- $C_y = A\ C_x\ A^T$

4.00	0.00	0.00
0.00	4.00	0.00
0.00	0.00	1.00

Example:

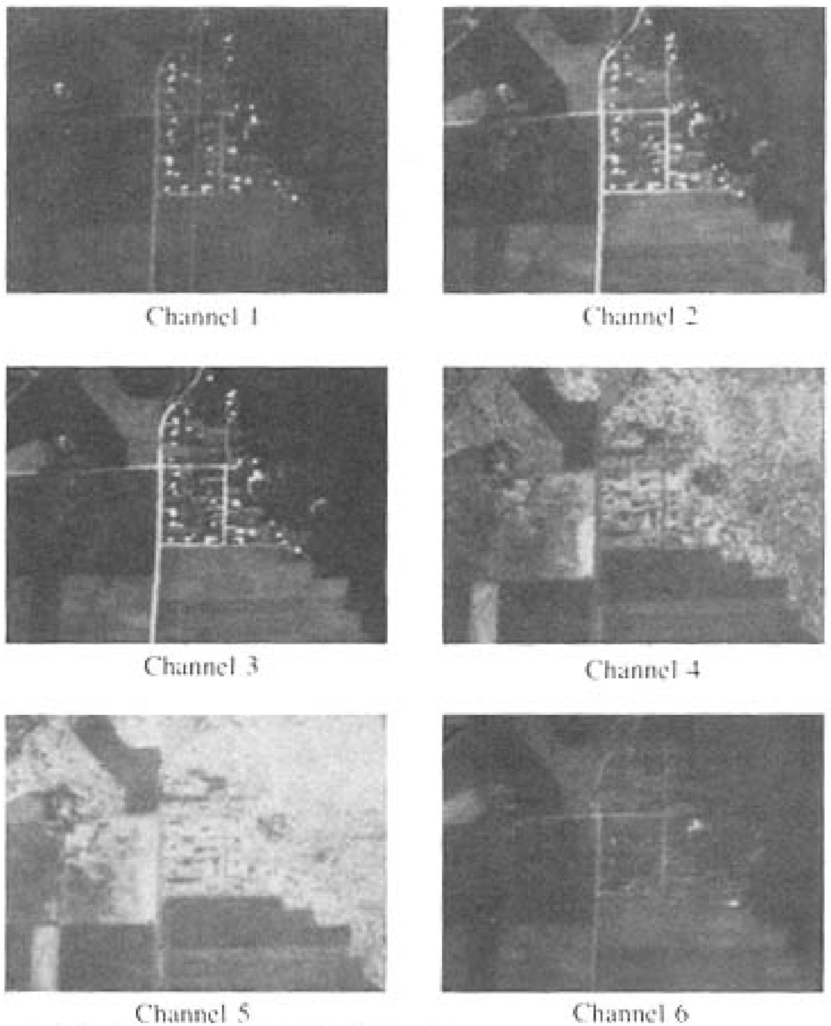


(a) A binary object; (b) its principal axes (eigenvectors);
(c) object rotated by Hotelling Transform

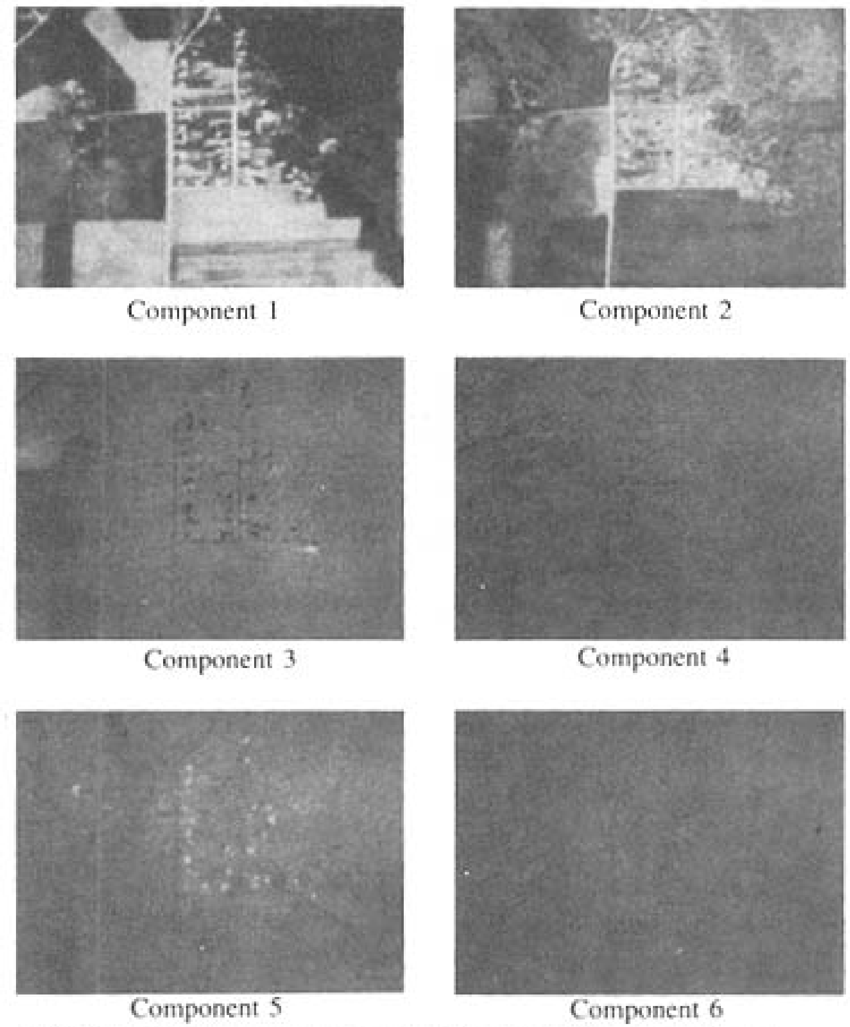
- Establish a new coordinate system whose origin is at the **centroid of the population**.
- And whose axes are in direction of eigen vectors of Cx.
- The transformation is rotation which aligns the data with the eigen vectors.
- The alignment is a mechanism which decorrelates the data.

- The concept of aligning a 2-D object with its principal eigen vectors plays an important role in image analysis.
- After an object has been extracted from an image, computer techniques for recognizing the image are sensitive to image rotation.
- Because the identity of an object is not known prior to recognition, the ability to align the object with its principal axes provides a reliable means for removing the effects of rotation from the image analysis process.

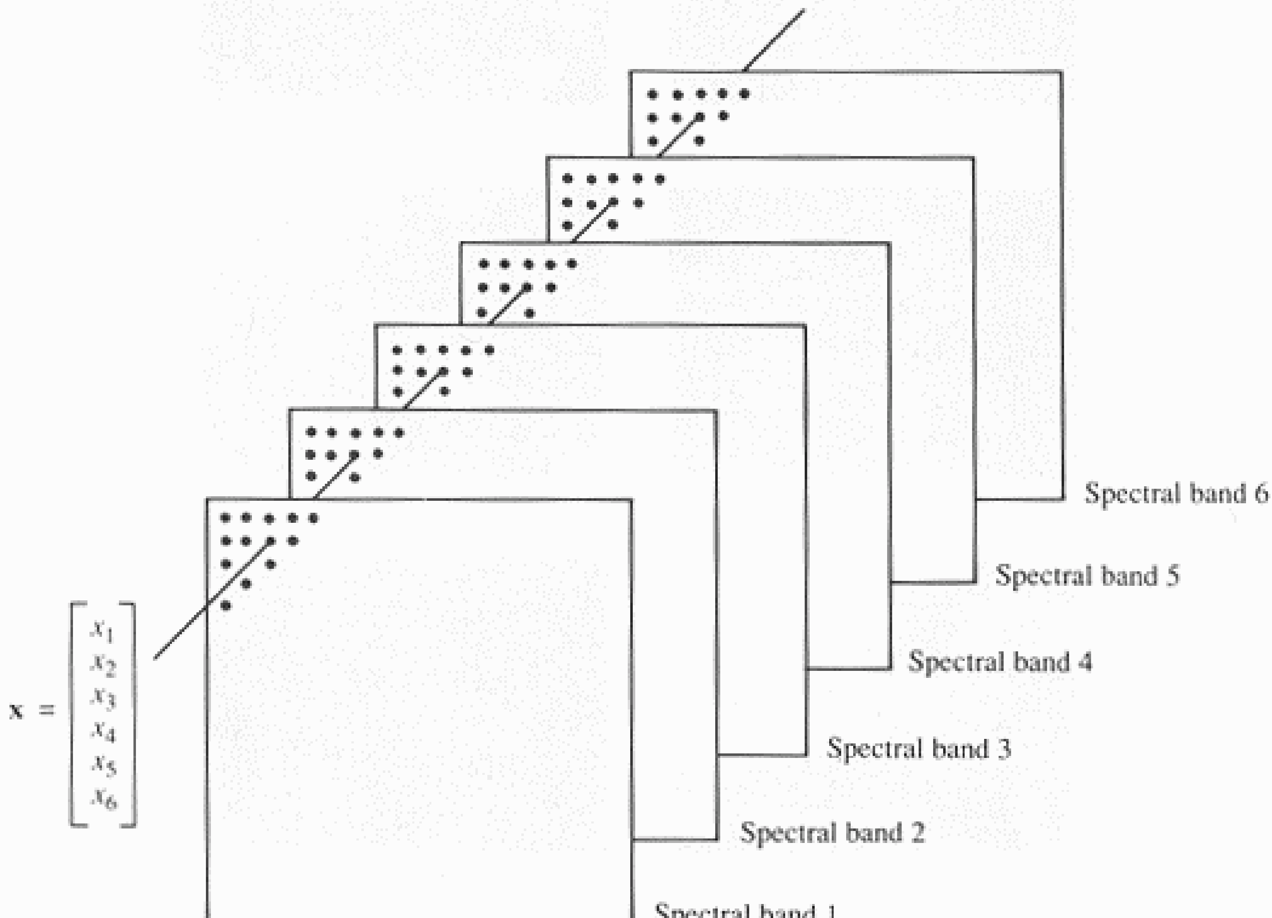
From channels to principle components



(a) A six-channel aerial image



(b) Six principle components after Hotelling Transform



Formation of a vector from corresponding pixels in six images.

- Eigen values of covarience matrix of the images shown
- $\lambda 1=3210$
- $\lambda 2=931.4$
- $\lambda 3=118.5$
- $\lambda 4=83.88$
- $\lambda 5=64.00$
- $\lambda 6=13.40$

- As the first two images account for 94% of the total variance
- Instead of storing all 6 images, only first 2 images along with mx and first 2 rows of A are stored.
- **Data compression** is by product of Hotelling transform.

Camera model

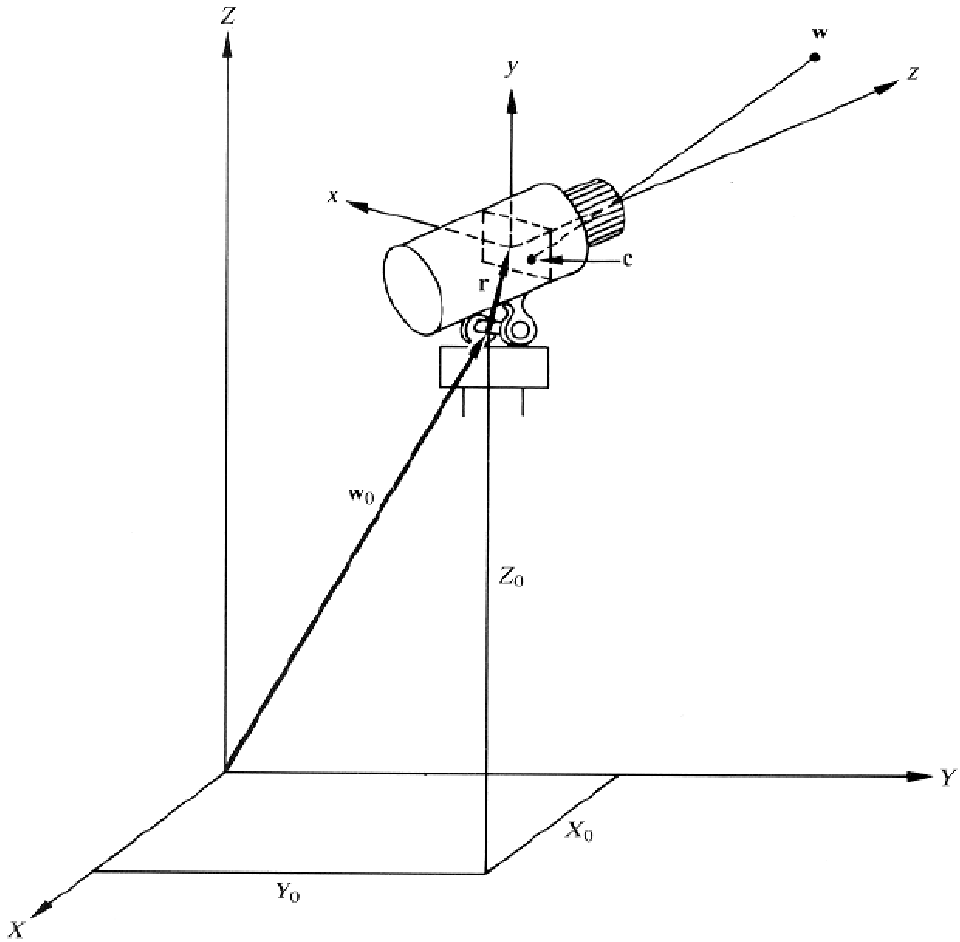


Figure 2.18 *Imaging geometry with two coordinate systems. (From Fu, Gonzalez, and Lee [1987].)*

Suppose that, initially, the camera was in *normal position*, in the sense that the gimbal center and origin of the image plane were at the origin of the world coordinate system, and all axes were aligned. The geometric arrangement of Fig. 2.18 may then be achieved in several ways. Let us assume the following sequence of steps: (1) displacement of the gimbal center from the origin, (2) pan of the x axis, (3) tilt of the z axis, and (4) displacement of the image plane with respect to the gimbal center.

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 0 & -X_0 \\ 0 & 1 & 0 & -Y_0 \\ 0 & 0 & 1 & -Z_0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

$$\mathbf{R} = \begin{bmatrix} \cos\theta & \sin\theta & 0 & 0 \\ -\sin\theta\cos\alpha & \cos\theta\cos\alpha & \sin\alpha & 0 \\ \sin\theta\sin\alpha & -\cos\theta\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{C} = \begin{bmatrix} 1 & 0 & 0 & -r_1 \\ 0 & 1 & 0 & -r_2 \\ 0 & 0 & 1 & -r_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{c}_h = \mathbf{PCRGw}_h$$

$$x = \lambda \frac{(X - X_0)\cos\theta + (Y - Y_0)\sin\theta - r_1}{-(X - X_0)\sin\theta\sin\alpha + (Y - Y_0)\cos\theta\sin\alpha - (Z - Z_0)\cos\alpha + r_3 + \lambda} \tag{2.5-42}$$

and

$$y = \lambda \frac{-(X - X_0)\sin\theta\cos\alpha + (Y - Y_0)\cos\theta\cos\alpha + (Z - Z_0)\sin\alpha - r_2}{-(X - X_0)\sin\theta\sin\alpha + (Y - Y_0)\cos\theta\sin\alpha - (Z - Z_0)\cos\alpha + r_3 + \lambda} \tag{2.5-43}$$

Summary of Image Transforms

DFT/unitary DFT	Fast transform, most useful in digital signal processing, convolution, digital filtering, analysis of circulant and Toeplitz systems. Requires complex arithmetic. Has very good energy compaction for images.
Cosine	Fast transform, requires real operations, near optimal substitute for the KL transform of highly correlated images. Useful in designing transform coders and Wiener filters for images. Has excellent energy compaction for images.

Hadamard	Faster than sinusoidal transforms, since no multiplications are required; useful in digital hardware implementations of image processing algorithms. Easy to simulate but difficult to analyze. Applications in image data compression, filtering, and design of codes. Has good energy compaction for images.
Haar	Very fast transform. Useful in feature extracton, image coding, and image analysis problems. Energy compaction is fair.
Slant	Fast transform. Has "image-like basis"; useful in image coding. Has very good energy compaction for images.
Karhunen-Loeve	Is optimal in many ways; has no fast algorithm; useful in performance evaluation and for finding performance bounds. Useful for small size vectors e.g., color multispectral or other feature vectors. Has the best energy compaction in the mean square sense over an ensemble.