



Universität Hamburg

DER FORSCHUNG | DER LEHRE | DER BILDUNG

MIN-Fakultät

Fachbereich Informatik

Arbeitsbereich SAV/BV (KOGS)

Image Processing 1 (IP1)

Bildverarbeitung 1

Lecture 8 – Image Compression 1

Winter Semester 2014/15

Slides: Prof. Bernd Neumann

Slightly revised by: Dr. Benjamin Seppke & Prof. Siegfried Stiehl

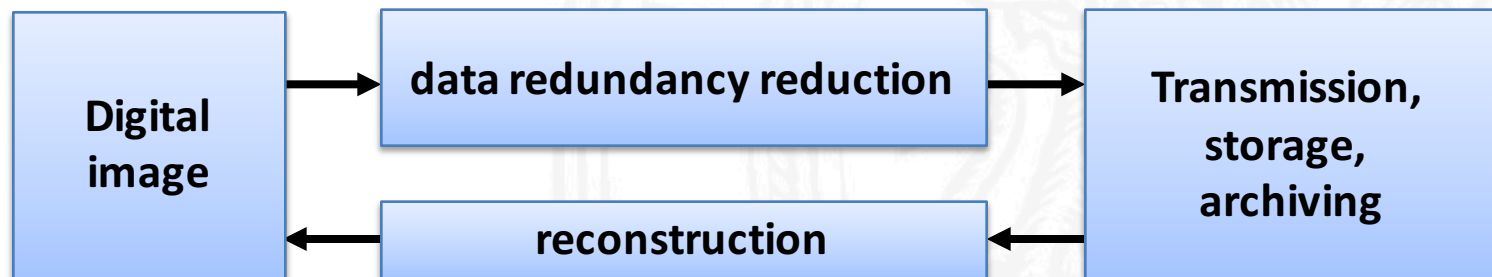
Image Data Compression

Data compression allows to save storage space as well as transmission capacity for digital/discrete data. Therefore, the data will be rearranged in a new structure with needs less storage than the former one.

Image data compression is important for:

- Image archives e.g. large satellite images
- Image transmission e.g. over the internet
- Multimedia applications e.g. desktop editing

Image data compression exploits redundancy for more efficient coding:



Different Kinds of Compression

Lossless compression:

- Original data may be reconstructed **exactly** from the compressed data.
- Examples:
 - File compression, e.g. ZIP, GZ etc.
 - Image formats: TIFF and PNG

Lossy compression:

- Original data may be reconstructed **approximately** from the compressed data.
- Examples:
 - JPEG image files
 - DVDs and Blu-Rays
 - MP3 audio files

Run Length Coding

Images with repeating greyvalues along rows (or columns) can be compressed by storing "runs" of identical greyvalues in the format:



For B/W images (e.g. fax data) another run length code is used:



	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0																
1																
2																
3																

Run length code:

(0 3 5 9 9)

(1 1 7 9 9)

(3 4 4 6 6 8 8 10 10 12 14)

Probabilistic Data Compression

A discrete image encodes information redundantly if

1. the greyvalues of individual pixels are not equally probable
2. the greyvalues of neighbouring pixels are correlated

Information Theory provides limits for minimal encoding of probabilistic information sources.

Redundancy of the encoding of individual pixels with G greylevels each:

$$r = b - H \qquad b = \lceil \log_2 G \rceil = \text{number of bits used for each pixel}$$

$$H = \sum_{g=0}^{G-1} P(g) \log_2 \frac{1}{P(g)} \qquad H = \text{entropy of pixel source}$$

= mean number of bits required to encode information of this source

The entropy of a pixel source with equally probable greyvalues is equal to the number of bits required for coding.

Huffman Coding I

The Huffman coding scheme provides a variable-length code with minimal average code-word length, i.e. least possible redundancy, for a discrete message source.
(Here messages are greyvalues)

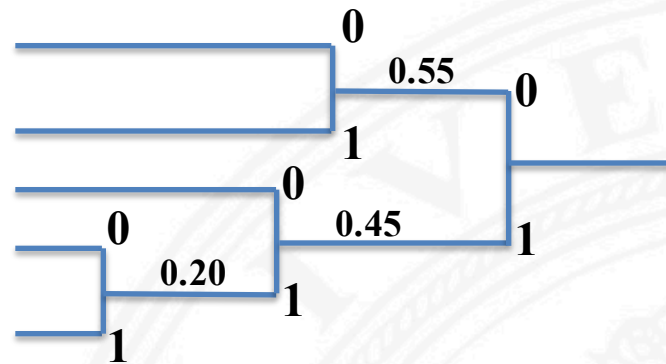
Algorithm:

1. Sort messages along increasing probabilities such that $g^{(1)}$ and $g^{(2)}$ are the least probable messages
2. Assign 1 to code word of $g^{(1)}$ and 0 to the code word of $g^{(2)}$.
3. Merge $g^{(1)}$ und $g^{(2)}$ to one new message by adding their probabilities.
4. Repeat steps 1 – 4 until a single message is left.

Huffman Coding II

Example:

Message	Probability
$g^{(5)}$	0.30
$g^{(4)}$	0.25
$g^{(3)}$	0.25
$g^{(2)}$	0.10
$g^{(1)}$	0.10



Resultierende Codierungen:

Message	Probability	Coding
$g^{(5)}$	0.30	00
$g^{(4)}$	0.25	01
$g^{(3)}$	0.25	10
$g^{(2)}$	0.10	110
$g^{(1)}$	0.10	111

Entropy: $H = 2.185$



Mean code word length: 2.2

Statistical Dependence

An image may be modelled as a set of statistically dependent random variables with a multivariate distribution $p(\vec{x}) = p(x_1, x_2, \dots, x_N)$

Often the exact distribution is unknown and only correlations can be (approximately) determined.

Correlation of two variables:

$$E[x_i x_j] = c_{ij}$$

$$E[\vec{x} \vec{x}^T] = \begin{bmatrix} c_{11} & c_{12} & c_{13} & \cdots \\ c_{21} & c_{22} & c_{23} & \cdots \\ c_{31} & c_{32} & c_{33} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

Correlation matrix

Covariance of two variables:

$$E[(x_i - \mu_i)(x_j - \mu_j)] = v_{ij} \quad \text{with } \mu_k = \text{mean of } x_k$$

$$E[(\vec{x} - \vec{\mu})(\vec{x} - \vec{\mu})^T] = \begin{bmatrix} v_{11} & v_{12} & v_{13} & \cdots \\ v_{21} & v_{22} & v_{23} & \cdots \\ v_{31} & v_{32} & v_{33} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

Covariance matrix

Attention: Uncorrelated variables need not be statistically independent:

$$E[x_i x_j] = 0 \not\Rightarrow p(x_i, x_j) = p(x_i) \cdot p(x_j)$$

But: For Gaussian random variables, uncorrelatedness implies statistical independence.

Karhunen-Loève Transform

(also known as Hotelling Transform or Principal Components Analysis)

Determine uncorrelated variables \vec{y} from correlated variables \vec{x} by a linear transformation.

$$y = A(\vec{x} - \vec{\mu})$$

$$E[\vec{y} \vec{y}^T] = A E[(\vec{x} - \vec{\mu})(\vec{x} - \vec{\mu})^T] A^T = A V A^T = D \quad \mathbf{D \text{ is a diagonal matrix}}$$

- An orthonormal matrix A which diagonalizes the real symmetric covariance matrix V always exists.
- A is the matrix of eigenvectors of V , D is the matrix of corresponding eigenvalues.

Reconstruction of \vec{x} from \vec{y} using: $\vec{x} = A^T \vec{y} + \vec{\mu}$

Note: If \vec{x} is viewed as a point in n-dimensional Euclidean space, then A defines a rotated coordinate.

Compression and Reconstruction using the Karhunen-Loève Transform

Let us assume that the Eigenvalues λ_i and the corresponding Eigenvectors of A are sorted descending: $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_N$

$$D = \begin{bmatrix} \lambda_1 & 0 & 0 & \dots \\ 0 & \lambda_2 & 0 & \dots \\ 0 & 0 & \lambda_3 & \dots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

Eigenvectors \vec{a} and Eigenvalues λ are defined by $V \vec{a} = \lambda \vec{a}$ and may be estimated by solving:

$$\det(V - \lambda I) = 0.$$

There exist fast solution methods for the Eigenvalues of real symmetric matrices.

\vec{x} may be transformed into a K -dimensional vector \vec{y}_K , with $K < N$ using a transformation matrix A_K , which contains only the first K Eigenvectors of A , which correspond to the K largest Eigenvalues:

$$\vec{y}_K = A_K (\vec{x} - \vec{\mu})$$

The approximate reconstruction \vec{x}' minimizes the – mean square error (MSE) of a representation with K dimensions:

$$\vec{x}' = A_K^T \vec{y}_K + \vec{\mu}$$

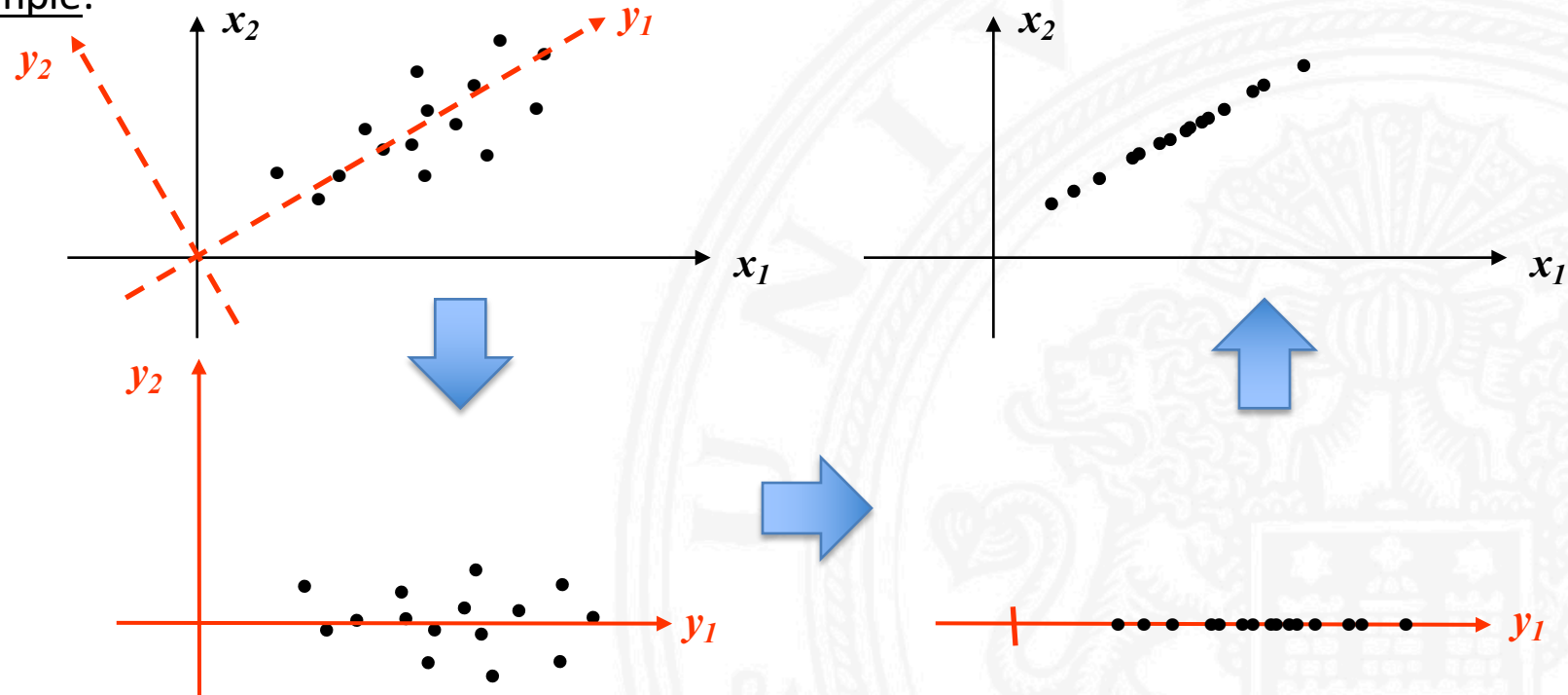
Thus, \vec{y}_K is representing a lossy data compression .

Illustration of Minimum-loss Dimension Reduction

Using the Karhunen-Loève transform, data compression is achieved by

- changing (rotating) the coordinate system
- omitting the least informative dimension(s) in the new coordinate system

Example:



Numerical Example for Karhunen-Loève Compression

Given: $N = 3$
 $\vec{x}^T = (x_1 \ x_2 \ x_3)$
 $\vec{m} = 0$

$$V = \begin{pmatrix} 2 & -0.866 & -0.5 \\ -0.866 & 2 & 0 \\ -0.5 & 0 & 2 \end{pmatrix}$$

Eigenvalues and eigenvectors

$$\det(V - \lambda I) = 0 \Rightarrow \lambda_1 = 3, \lambda_2 = 2, \lambda_3 = 1$$

$$V = \begin{pmatrix} 0.707 & 0 & 0.707 \\ -0.612 & 0.5 & 0.621 \\ -0.354 & -0.866 & 0.354 \end{pmatrix}$$

Compression into $K=2$ dimensions

$$\vec{y}_2 = A_2 \vec{x} = \begin{pmatrix} 0.707 & -0.612 & -0.354 \\ 0 & 0.5 & -0.866 \end{pmatrix} \vec{x}$$

Reconstruction from compressed values

$$\vec{x}' = A_2^T \vec{y} = \begin{pmatrix} 0.707 & 0 \\ -0.612 & 0.5 \\ -0.354 & -0.866 \end{pmatrix} \vec{y}$$

Note the discrepancies between the original and the approximated values:

$$x_1' = 0,5 x_1 - 0,43 x_2 - 0,25 x_3$$

$$x_2' = -0,085 x_1 - 0,625 x_2 + 0,39 x_3$$

$$x_3' = 0,273 x_1 + 0,39 x_2 + 0,25 x_3$$