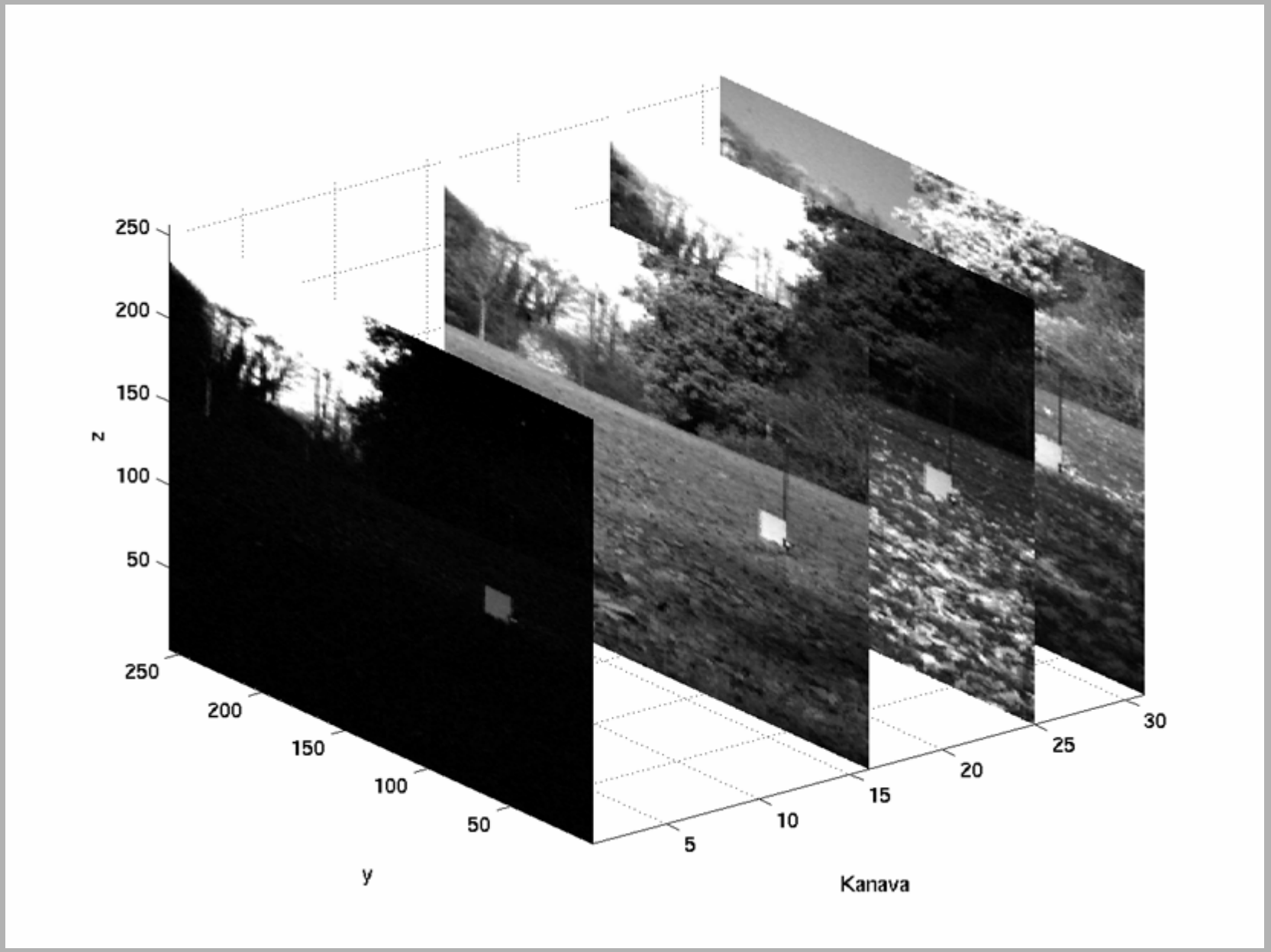
Digital Color

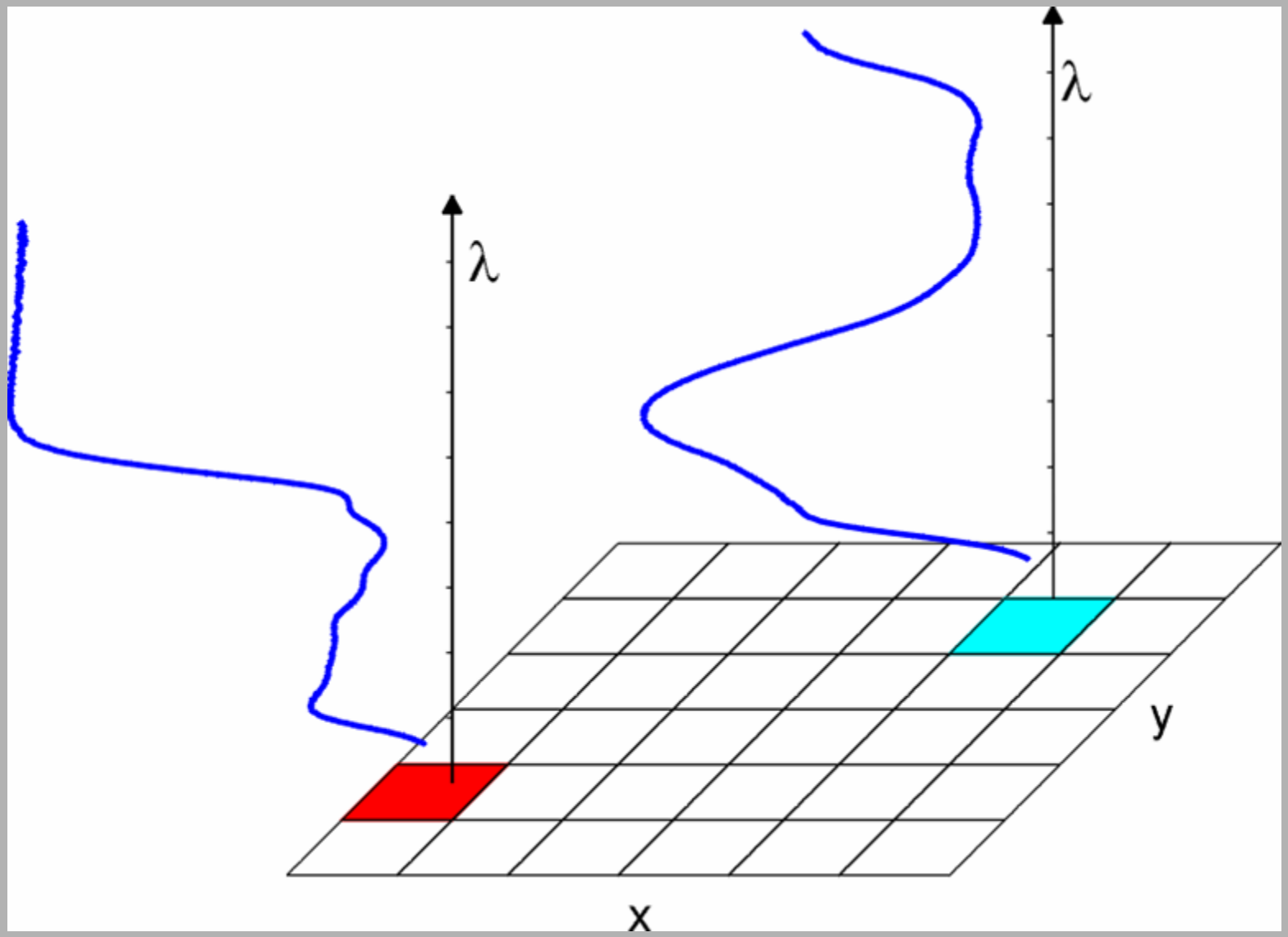
Lecture 6

Spectral image analysis

Spectral component images



Spectral Image



Algorithm PCA

1. Calculate correlation matrix for the data set
2. Calculate eigenvalues and eigenvectors for the correlation matrix
3. Select eigenvectors corresponding to the largest eigenvalues as a new basis for the data set
4. Calculate principal components between eigenvectors and the data set

- Applications: compression, pattern recognition etc. Frequently used for spectral image data

Spectral Images & Principal Component Analysis (PCA)

spectrum

$$\mathbf{s}(\lambda) = (s(\lambda_1)\ s(\lambda_2)\ \ldots\ s(\lambda_n))^T$$

2-D spectral image

$$\mathbf{S} = \begin{pmatrix} s_1(\lambda_1) & \cdots & s_m(\lambda_1) \\ \vdots & \ddots & \vdots \\ s_1(\lambda_n) & \cdots & s_m(\lambda_n) \end{pmatrix}$$

1. calculating correlation matrix and eigenvectors, selecting base vectors

$$\mathbf{R} = \frac{1}{m}\mathbf{S}\mathbf{S}^T \qquad \mathbf{R}\Phi = \sigma\Phi$$

$$\mathbf{B} = \begin{pmatrix} b_1(\lambda_1) & \cdots & b_1(\lambda_n) \\ \vdots & \ddots & \vdots \\ b_m(\lambda_1) & \cdots & b_m(\lambda_n) \end{pmatrix}$$

Matrix B needs to be transposed

2. calculating inner product images

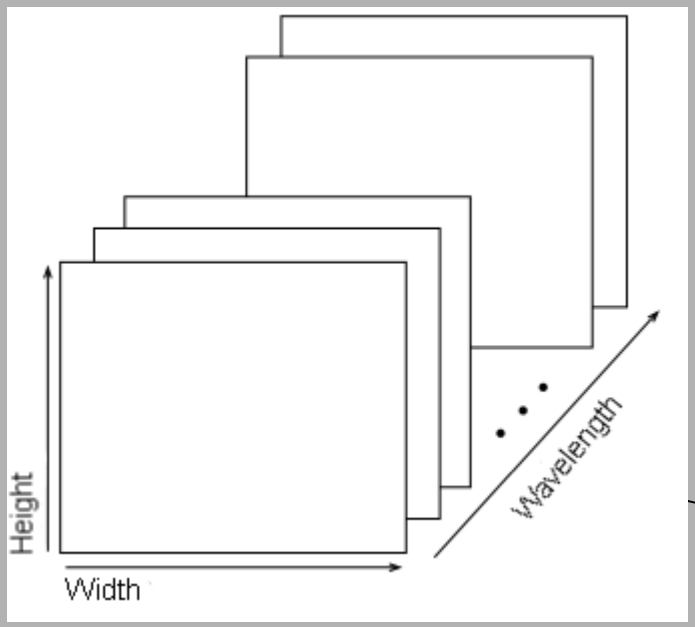
$$\mathbf{P} = \mathbf{B}^T\mathbf{S}$$

3. reconstructing spectral image

$$\tilde{\mathbf{S}} = \mathbf{B}\mathbf{B}^T\mathbf{S} = \mathbf{B}\mathbf{P}$$

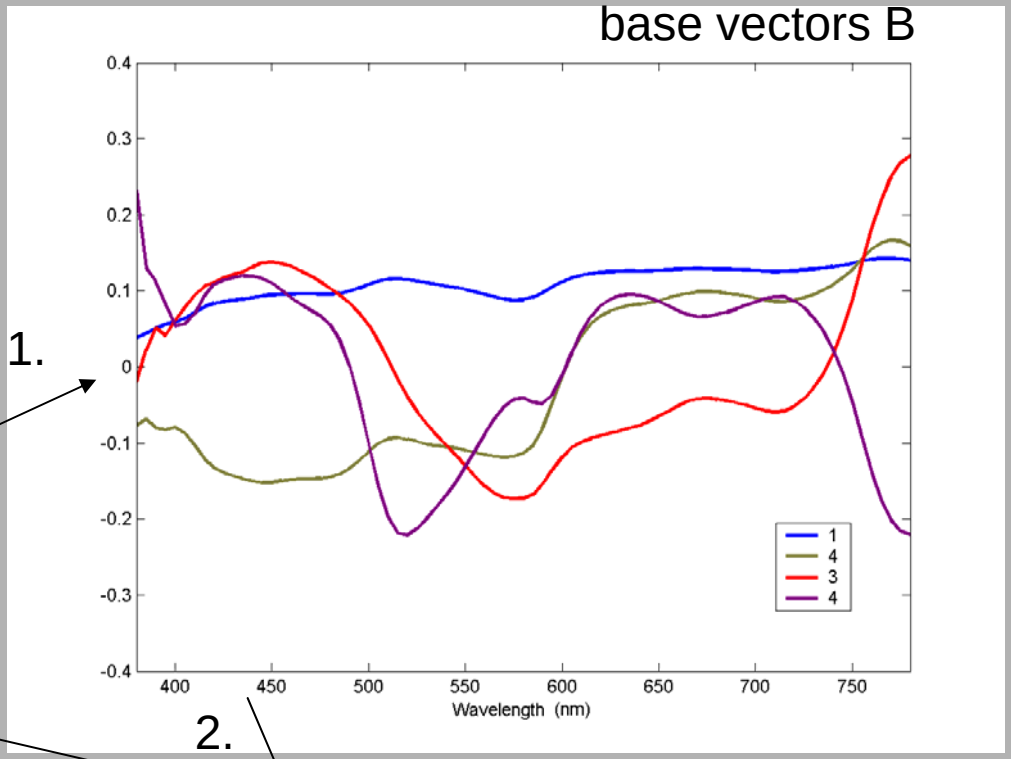
Illustration of PCA

original spectral image S



1.

base vectors B



2.

inner product images P

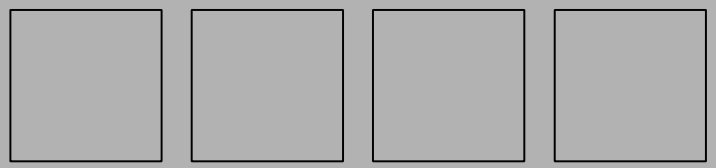
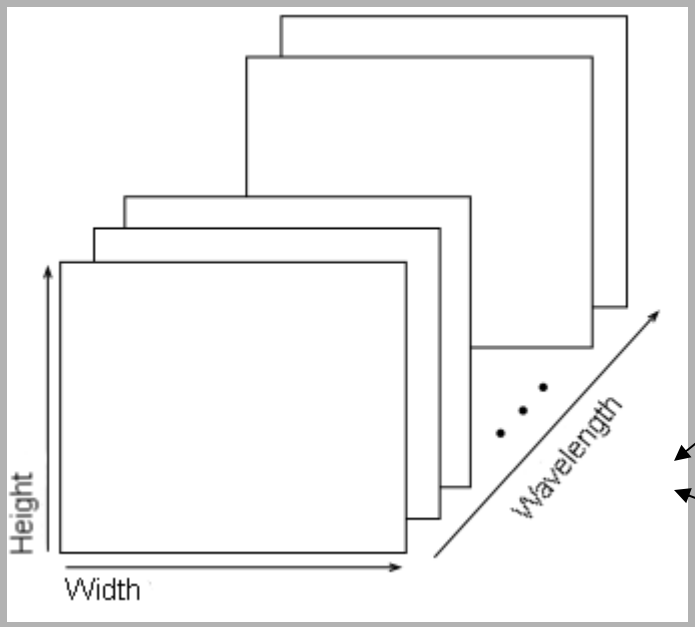
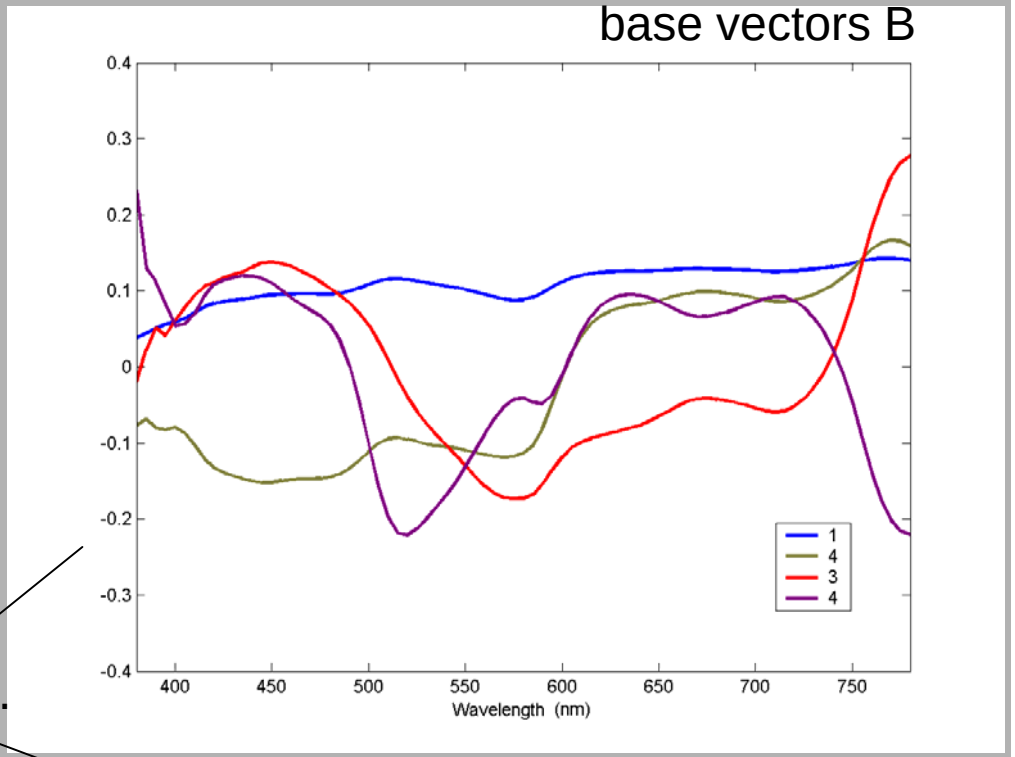


Illustration of PCA

reconstructed spectral image $\tilde{S}$

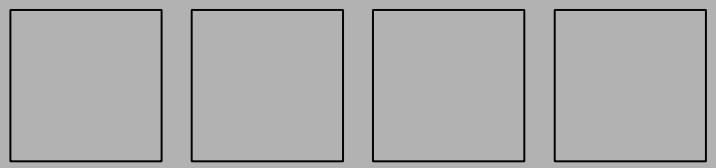


base vectors B

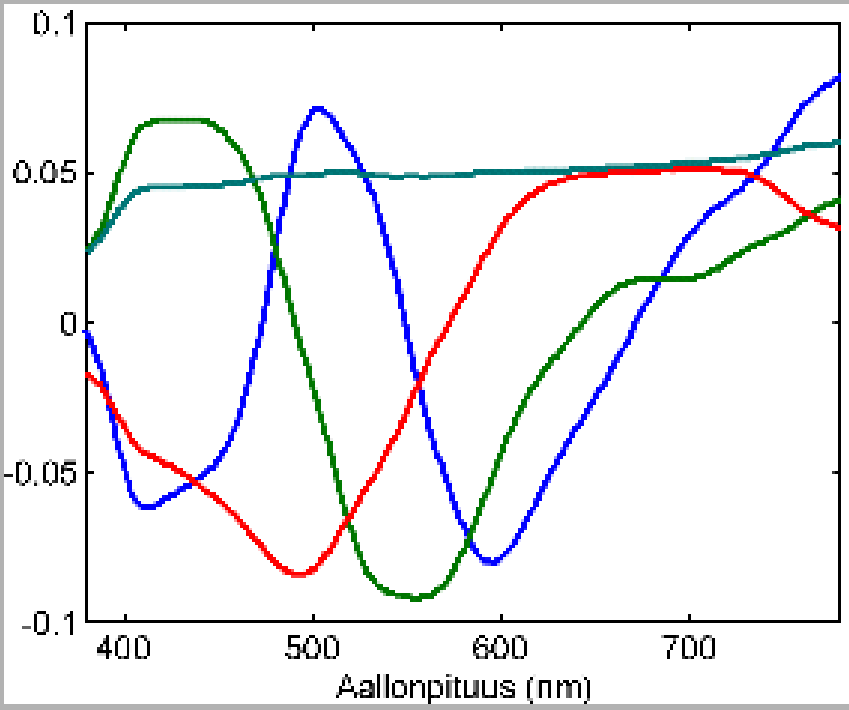


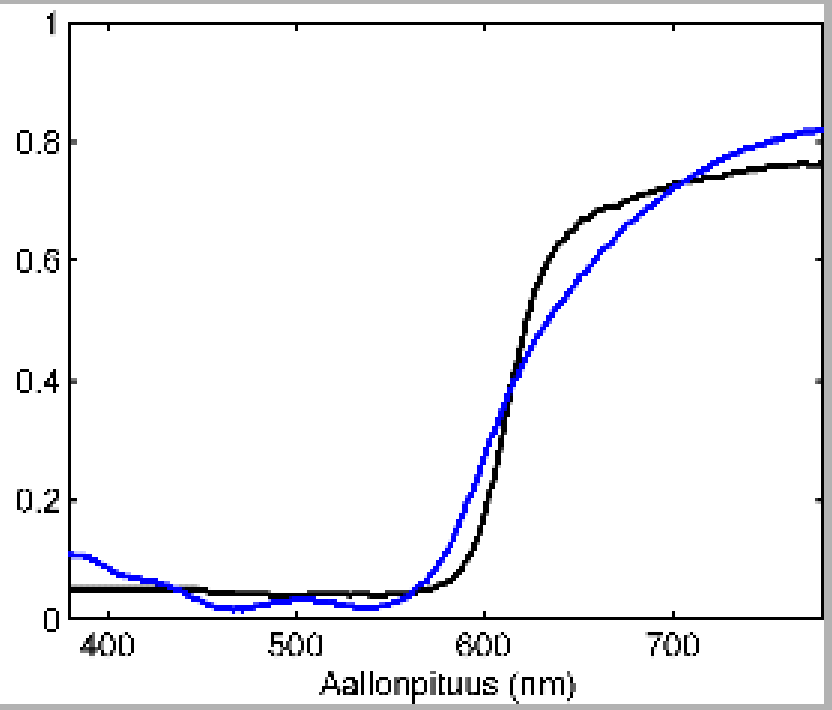
3.

inner product images P



Reconstruction of a spectrum

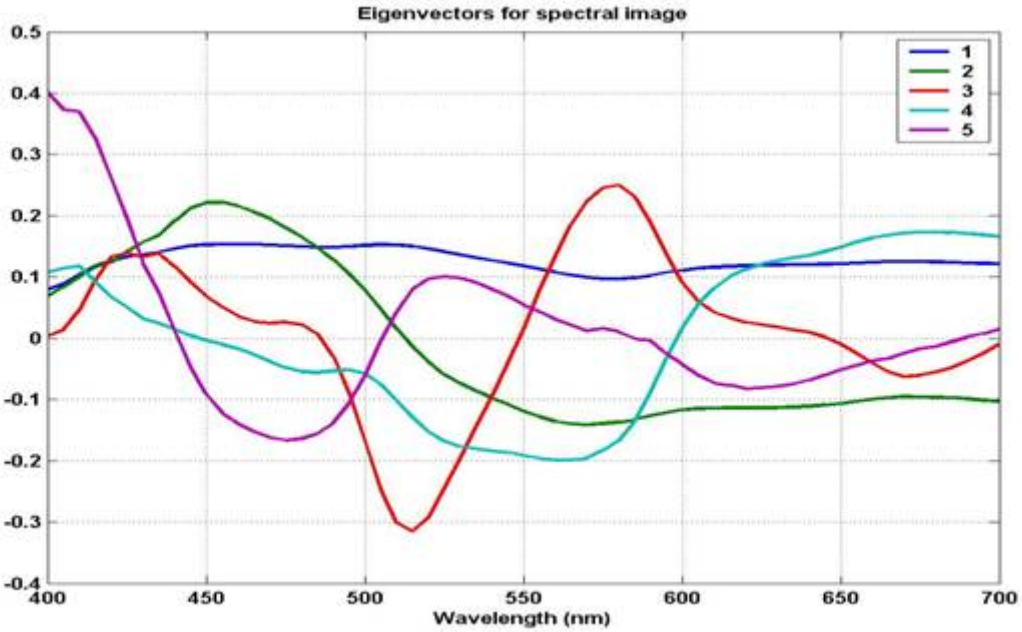




Component images of a spectral image

Spectral image as RGB-image

PCA



Inner-product images between the spectral image and eigenvectors:

1

2

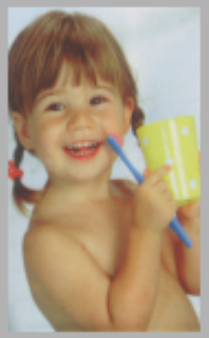
3

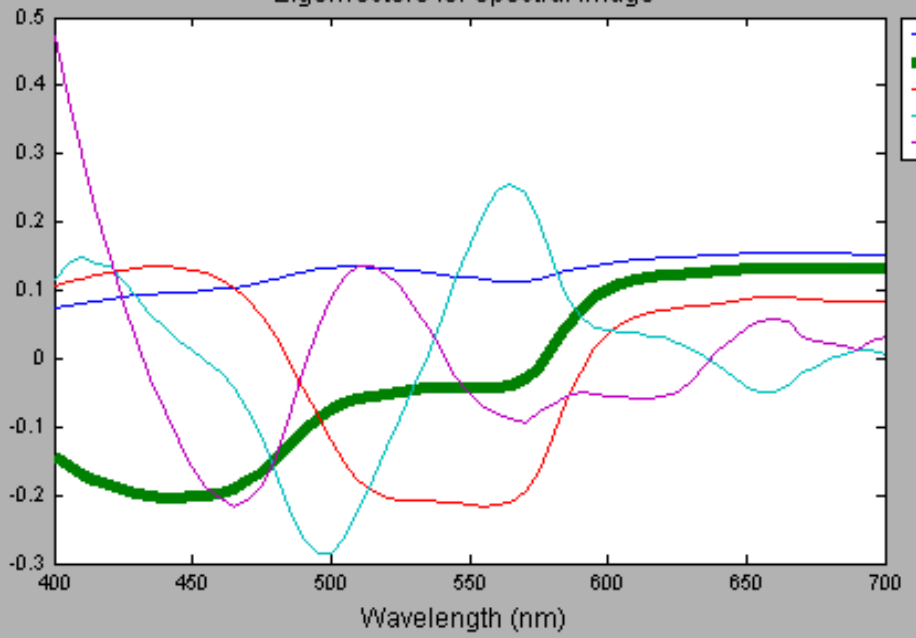
4

5

Principal Component Analysis

File Help





Load image

Wavelength
400:5:700

Change wavelength

Number of eigenvectors: 5

Update view

Select images for reconstruction

Reset selections

Fidelity 99.9937 %

Inner product images between spectral image and eigenvectors:

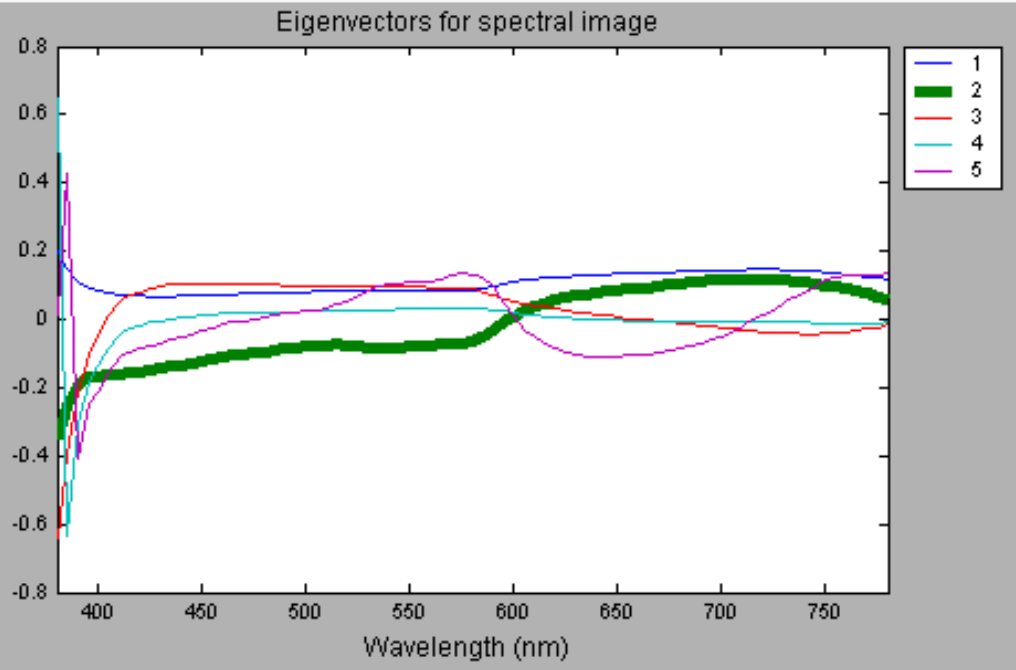
Images selected for reconstruction: All

1 2 3 4 5

Inner product images | Compression errors

Principal Component Analysis

File Help



Load image

Wavelength
380:5:780

Change wavelength

Number of eigenvectors: 5

Update view

Select images for reconstruction

Reset selections

Fidelity 99.8986 %

Inner product images between spectral image and eigenvectors:

Images selected for reconstruction: All

1 2 3 4 5

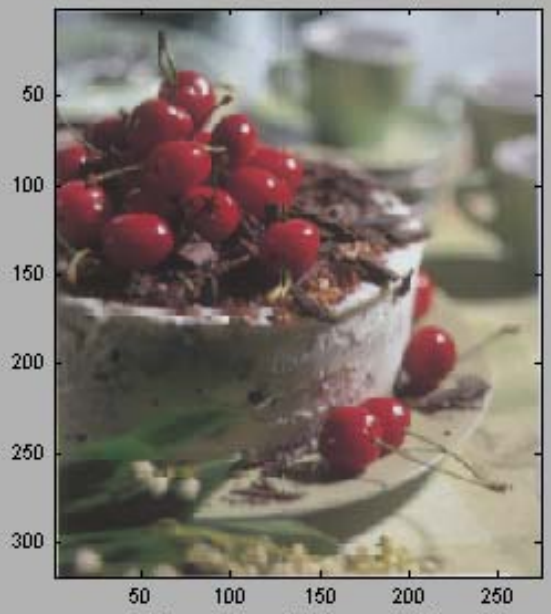
Inner product images

Compression errors

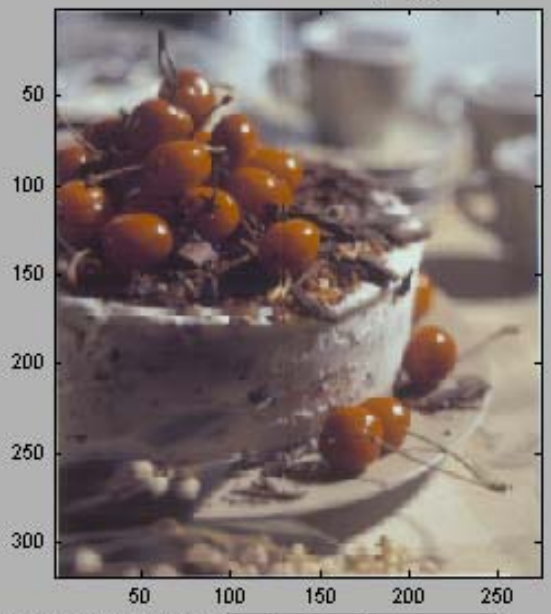
Principal Component Analysis

File Help

Original image (1)

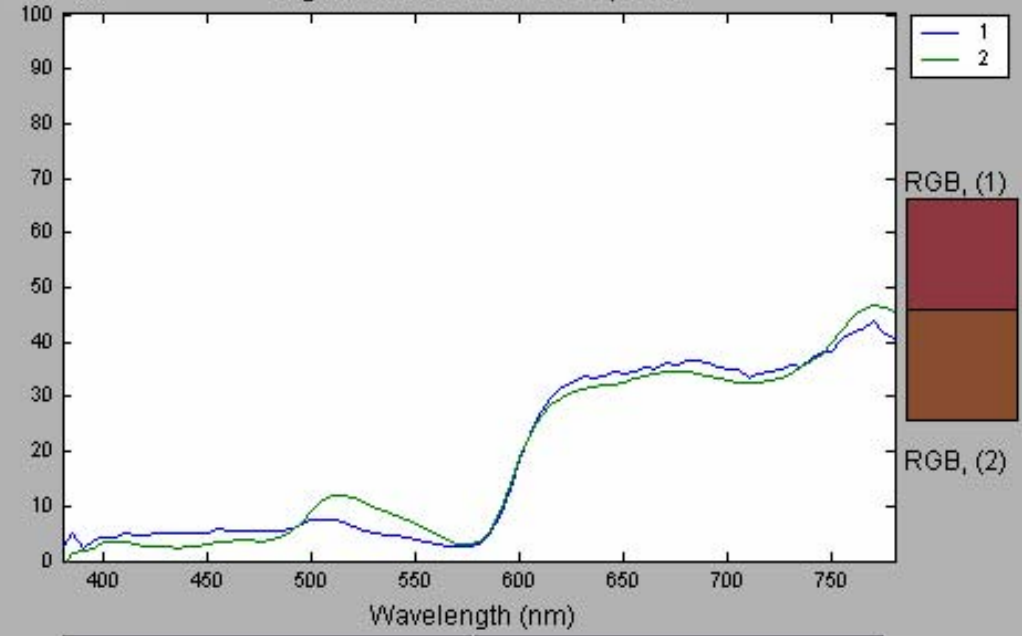


Reconstructed image (2)



Inner product images | Compression errors

Original and reconstructed spectra



Compression Rate	26.98	pixel 97,111 :	
Average errors:		Errors of selected spectrum:	
∆ E	7.54	∆ E	21.5
∆ E S-CIELAB	6.35	∆ E S-CIELAB	5.34
Peak Signal-to-Noise Ratio	34.3	Peak Signal-to-Noise Ratio	32.42
Mean Square Error	0.00037	Mean Square Error	0.00057

Zoom image | Original size | Actual pixels

Select light source:
- A
- C
- D50
- D65

Select system:
- 1931
- 1964

Spectrum type:
- Reflectance
- Relative radiance

Select other source

Spectral image compression

Original

PCA 1D

PCA 3D

PCA 4D

Spectral image databases

$\Rightarrow$ fast methods for searching images will be needed in the future

Experimental data

76 real-world spectral images

Algorithm of the SOM

- Let L be a discrete lattice of units x_i, where i is the index of the unit.
- Each unit x_i is assigned a weight vector w_i.
- A point s from the input space S is mapped to a *best matching unit* x_j for which

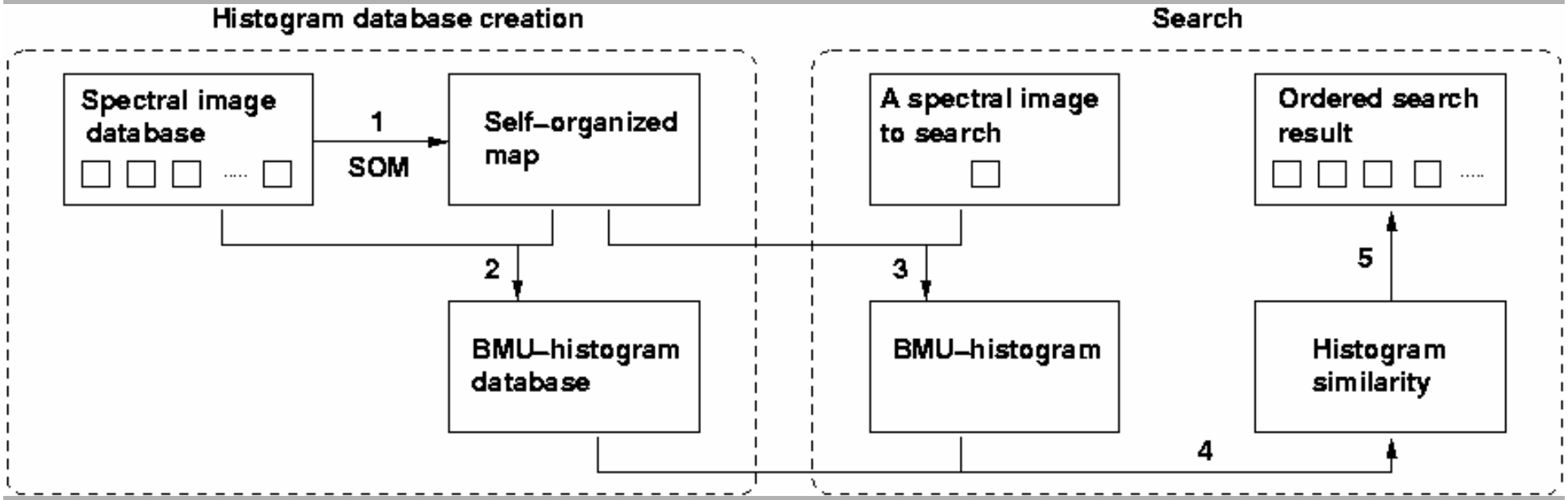
$$\|s - w_j\| = min_i \ \|s - w_i\|$$

- A self-organizing procedure:
  1. Draw a sample s from S and find the best matching unit x_j.
  2. Adjust the weights w_i of all units x_i in the neighborhood of x_j by

$$w_i^{new} = w_i^{old} + l\left(s - w_i^{old}\right),$$

     where l is the learning parameter.
  3. Repeat points 1. and 2. l times. l is the number of learning steps.

Searching Technique



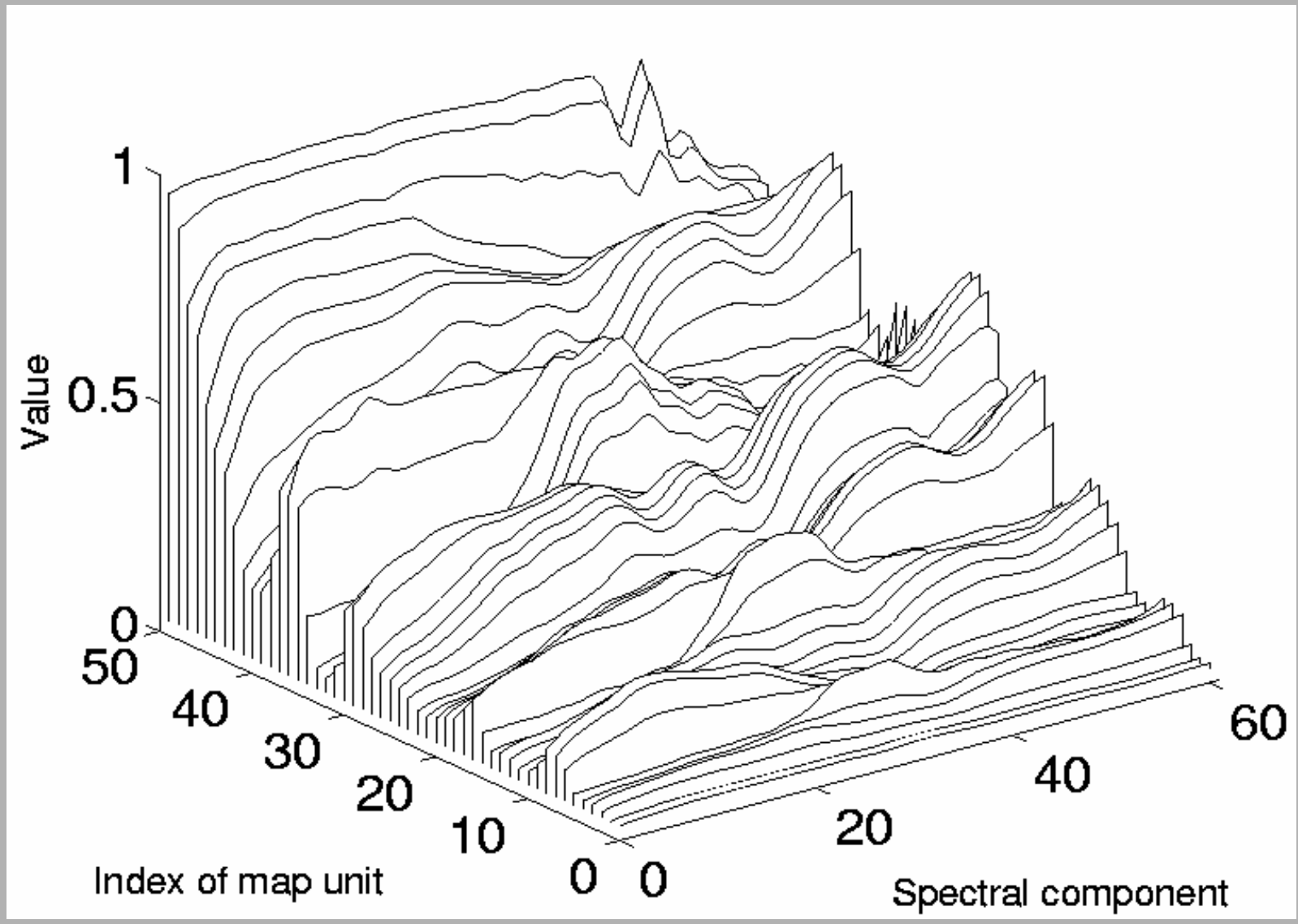
Histogram database creation

1. Select spectra randomly, train a SOM
2. Calculate BMU-images and BMU-histograms

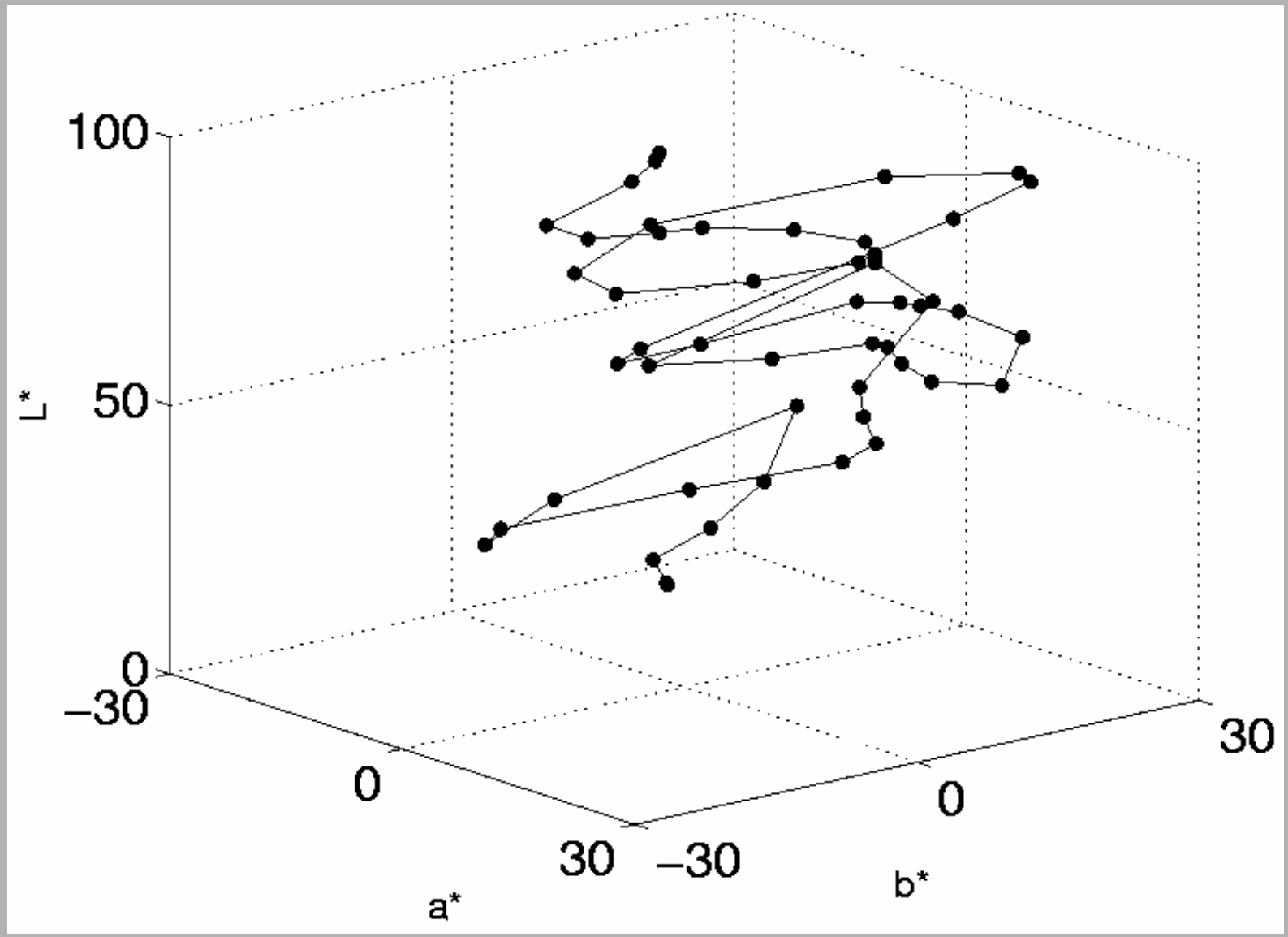
Search

3. Select a wanted image, calculate BMU-image and BMU-histogram
4. Calculate histogram similarities
5. Order the search results based on the similarities

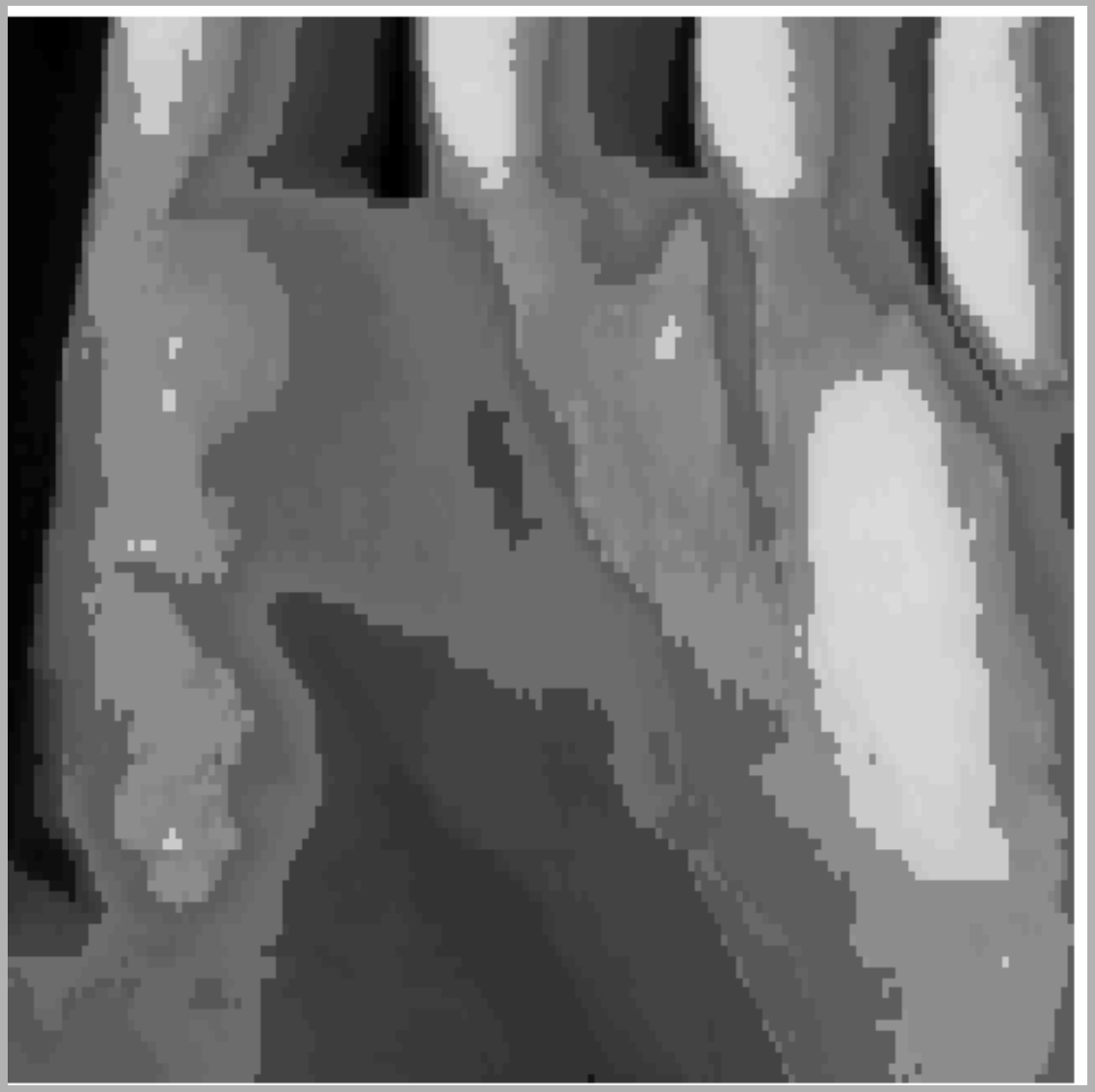
Self Organizing Map



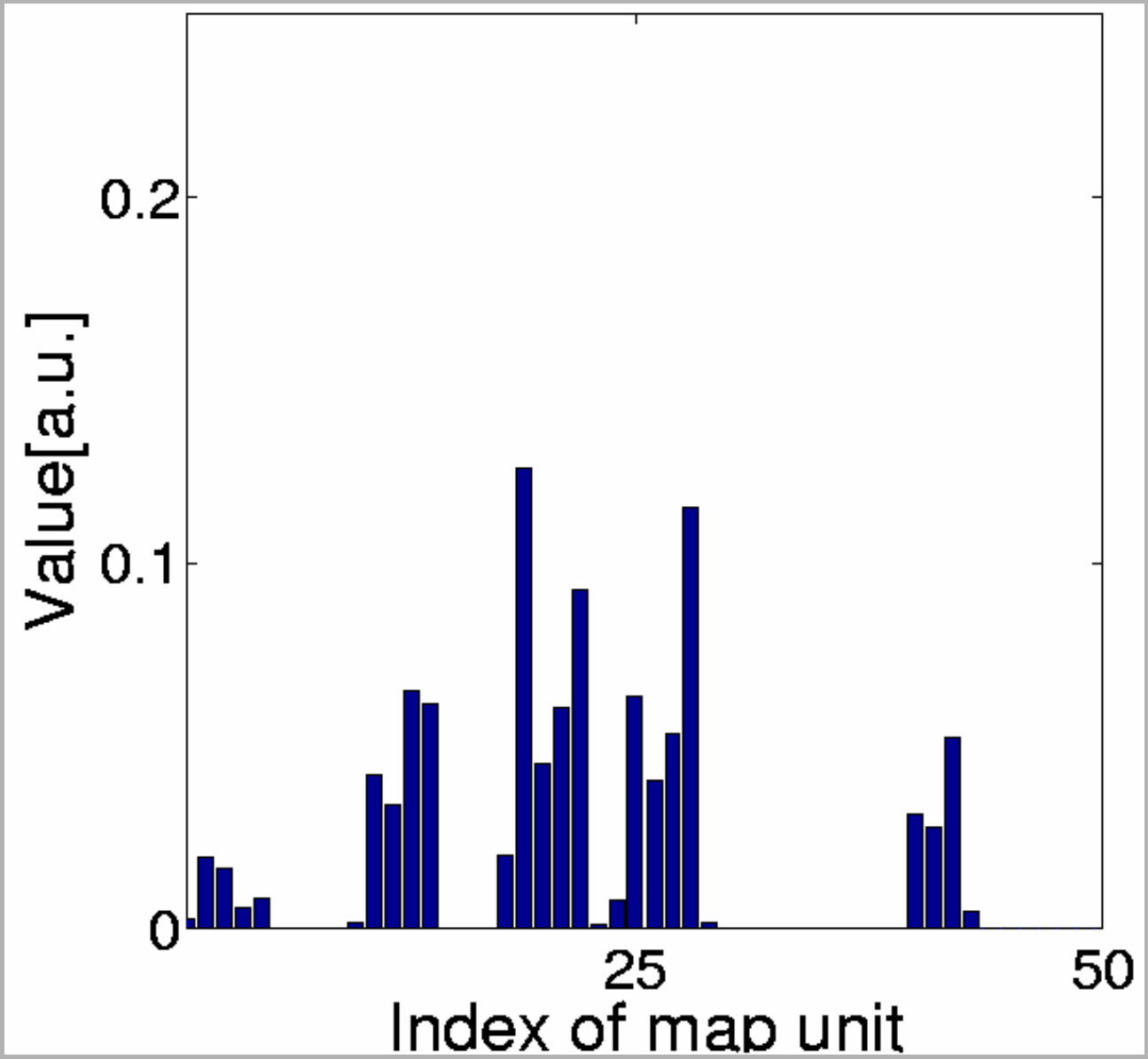
SOM-units in CIELAB



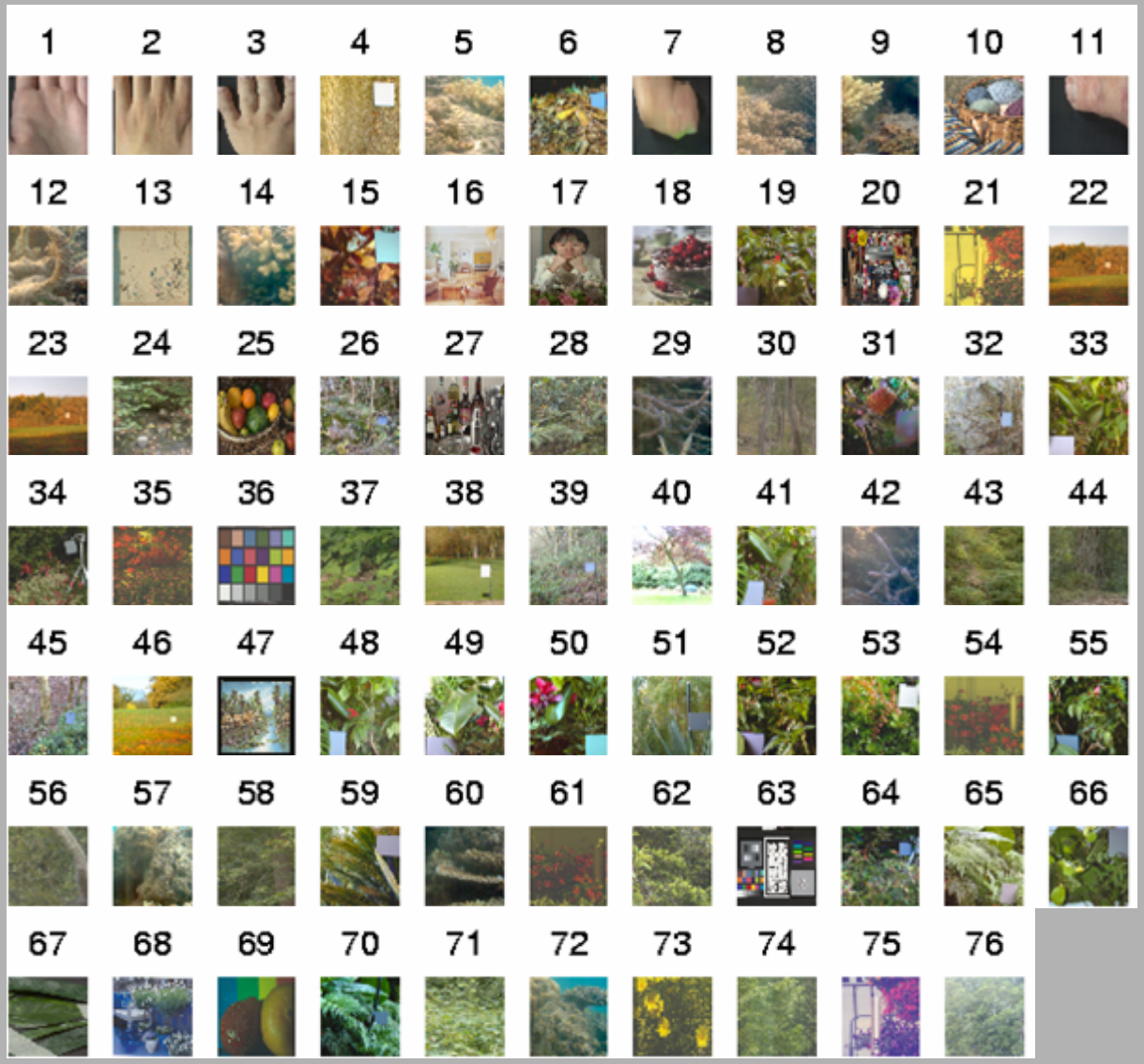
Example of BMU-image



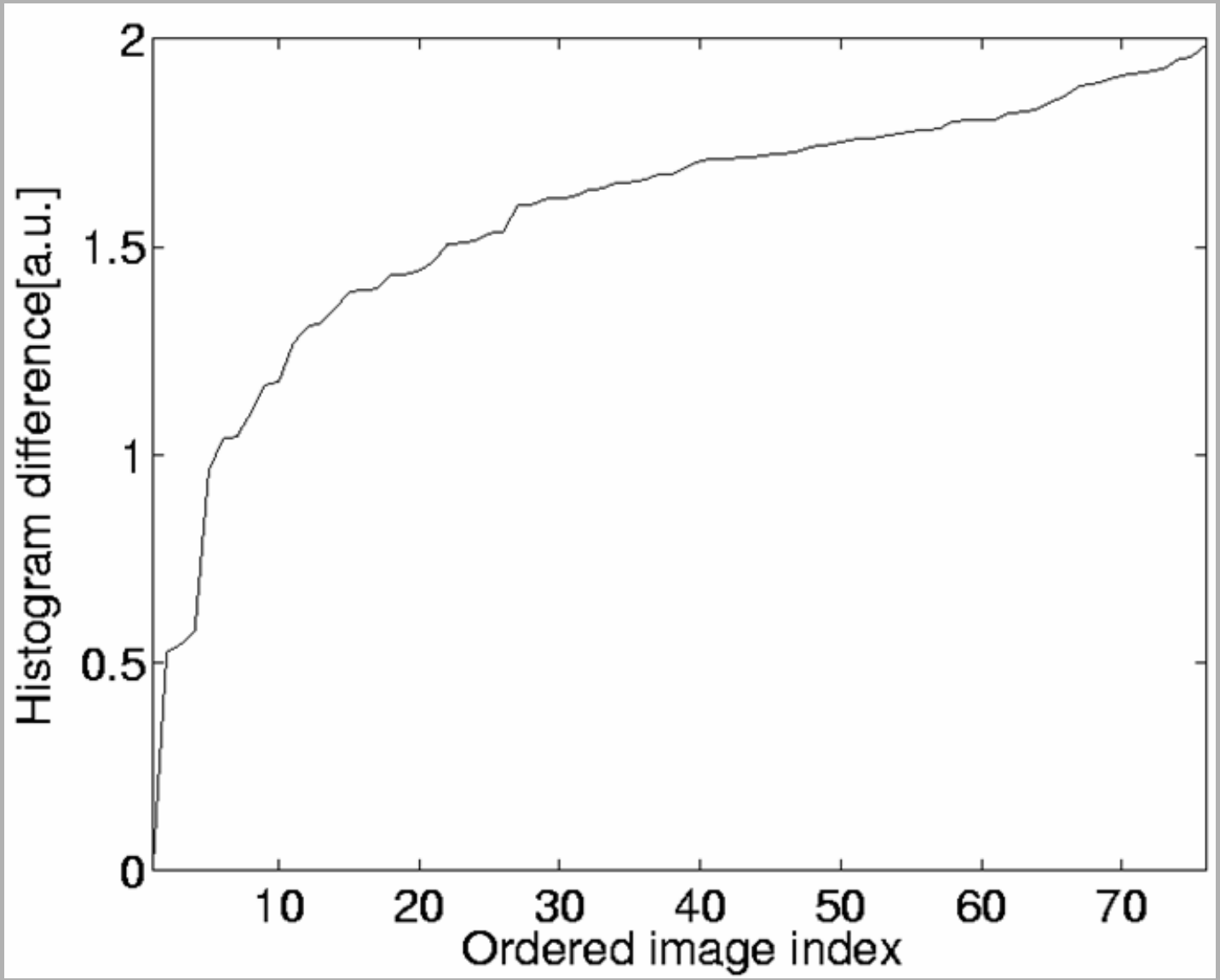
Example of BMU-histogram



Example of Search



Histogram Differences



Principle of SOM: finding the BMU

Mathematically the BMU is defined for input data vector, *x*, as follows:

$$\|x - w_{\mathrm{BMU}}\| = \min_i \|x - w_i\|$$

Euclidean distance is a typically used distance measure.

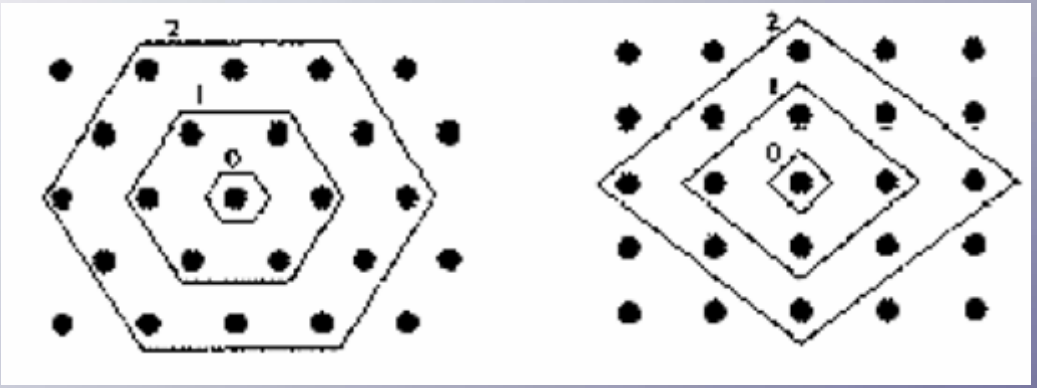
Principle of SOM: updating the weight vectors

$$w_i(t+1) = \begin{cases} w_i(t) + \alpha(t)[x(t) - w_i(t)], & \text{if } i \in N_{\text{BMU}}(t) \\ w_i(t), & \text{otherwise} \end{cases}$$

Learning rate: **product of learning rate parameter & neighborhood function:**

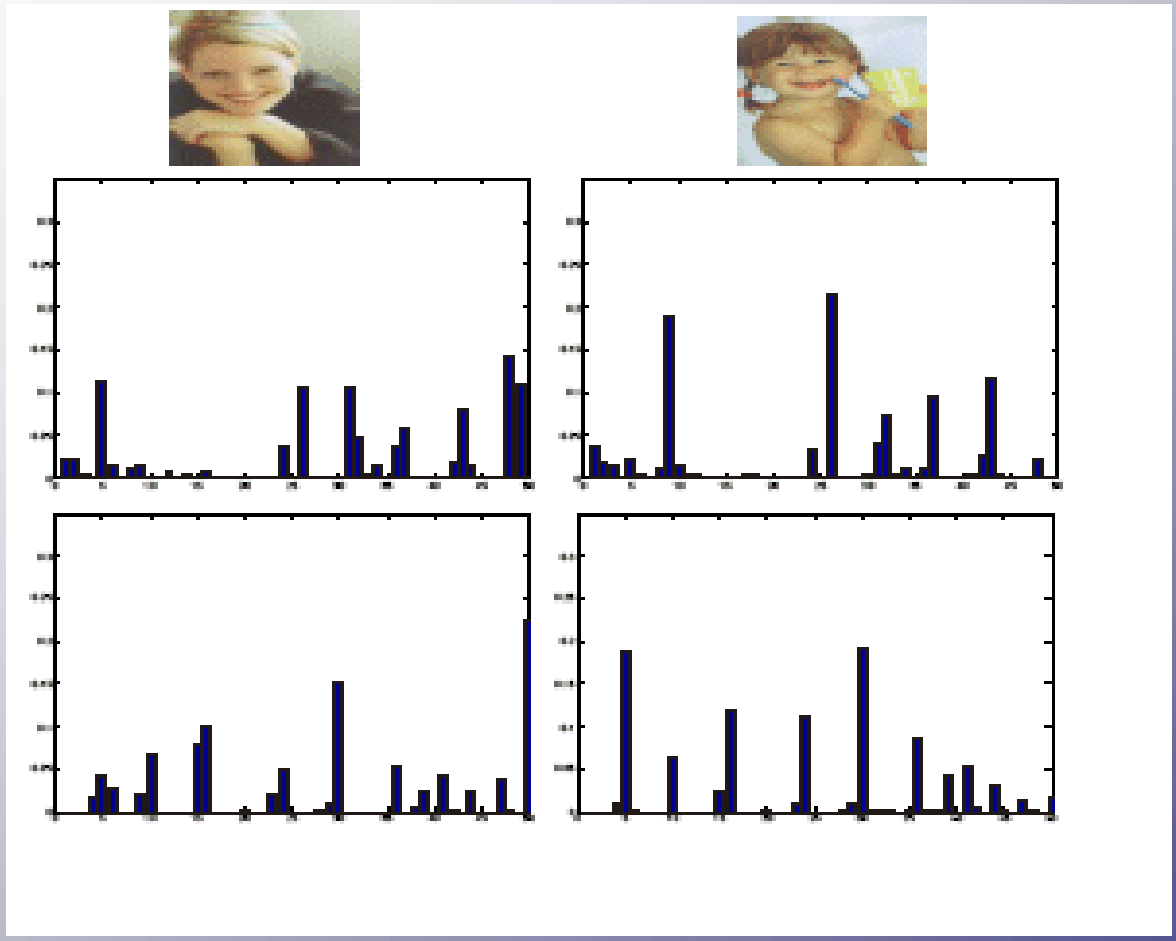
$$\alpha(t) = \eta(t)h(t)$$

Principle of SOM: Lattice structure



Lattice structures: hexagonal & rectangular

Connections between images and histograms:

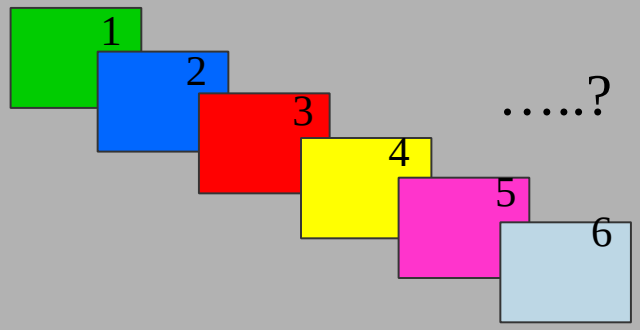


non-weighted

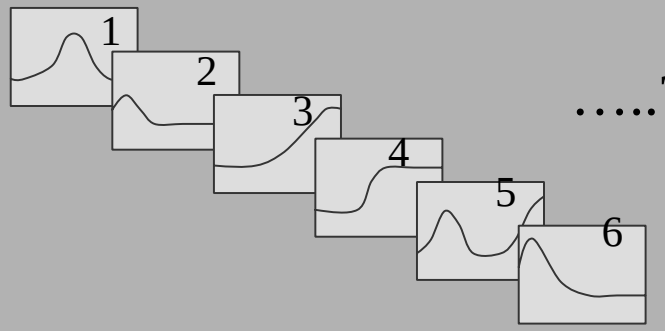
weighted

Ordering of Color Spectra

What is order of color?

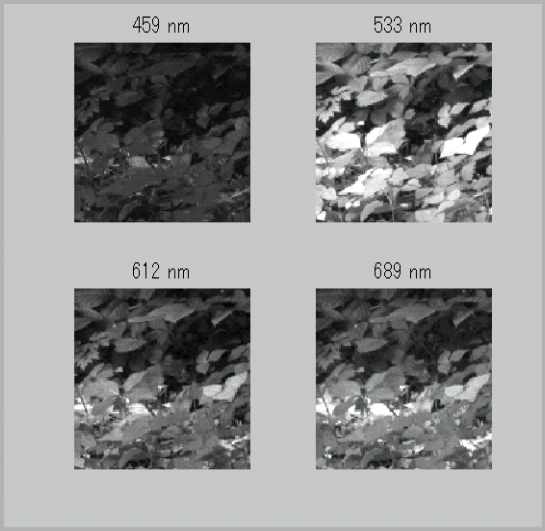


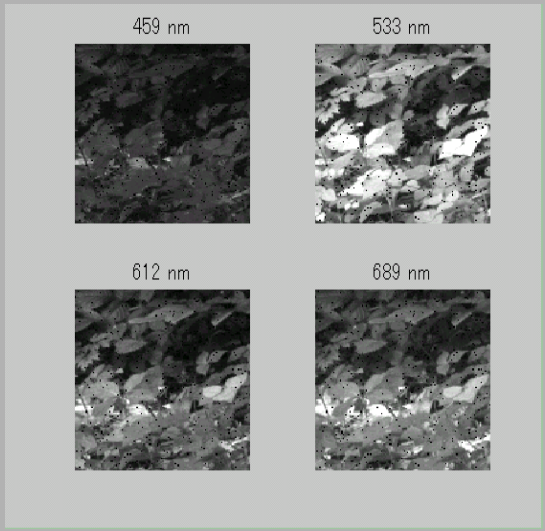
What is order of color spectra?

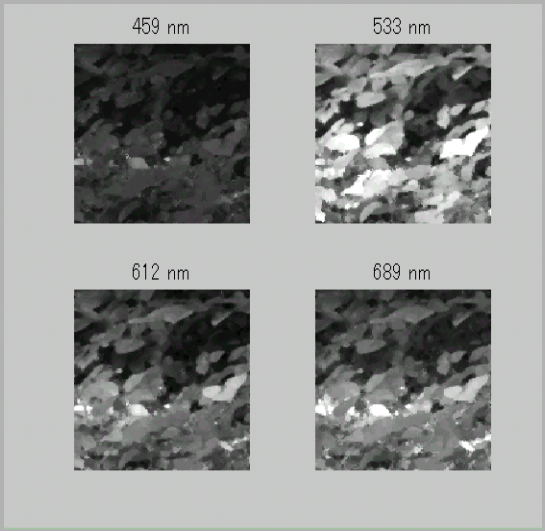


We have studied
Ordering of color spectra
using
Self-Organizing Map (1-D SOM) and

Spectral Image Filtering







Original → **with noise** → **Filtered**

Put pepper noise

Filtered by using MEDIAN filter

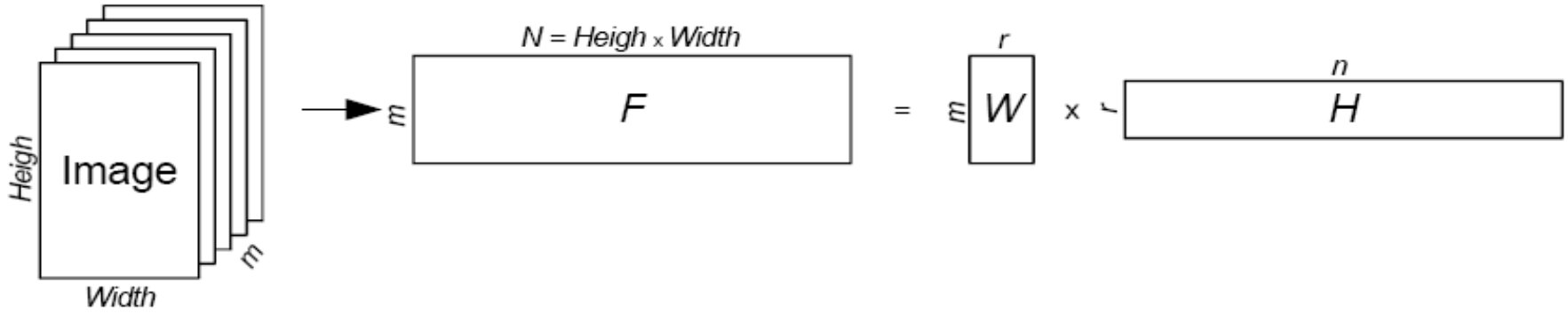
The medians are chosen according to the order defined by the SOM

Non-negative Matrix Factorization

$F = WH$,

where the elements in W and H are all positive values and the goal is to minimize the reconstruction error

$Min\ ||F - WH||; \quad W,H \geq 0$



Results

r = 1;

r = 2;

r = 3;

r = 5;

r = 81;

original

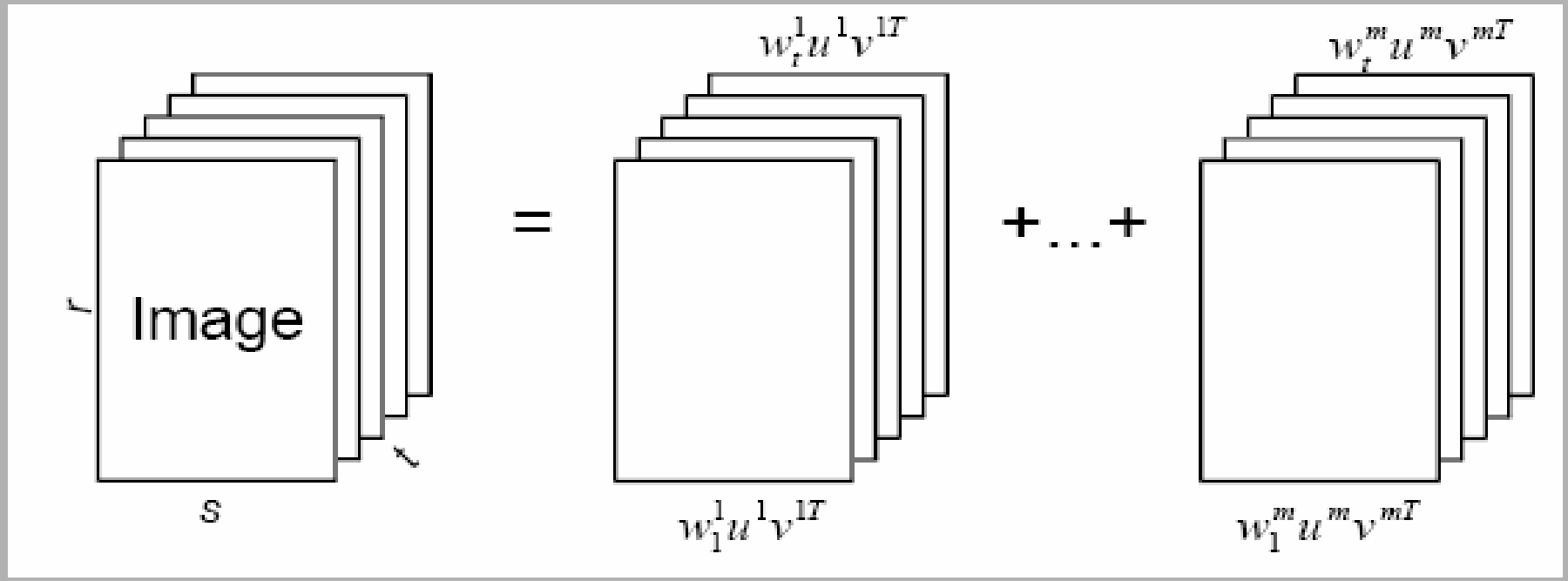
Problems

- NMF don’t care about image content information. Only color.
- NMF is iterative method and number of iterations and rank of factorization ***r*** is not clear
- Initial state is random and process can converge to a local minimum.

Non-negative Tensor Factorization

Image $G = \{G_{rst}\}$ can be expressed as

$$G = \sum_{j=1}^{k} U_1^j \otimes U_2^j \otimes \ldots \otimes U_n^j$$



$$u^j \otimes v^j \otimes w^j$$

Solution and Update rules

It was considered the following least-square problem:

$$\min_{u^m, v^m, w^m} \frac{1}{2} \left\| G - \sum_{m-1}^{k} u^m \otimes v^m \otimes w^m \right\|_F^2$$

$$subject\ to: u^m, v^m, w^m \geq 0$$

It was used a gradient decent scheme with a mixture of Jacobi and Gauss-Seidel update scheme and positive preserving update rule.

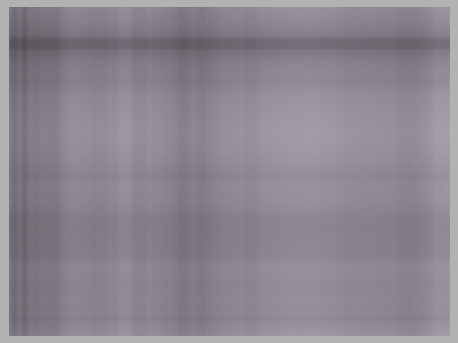
The update rule provided by Lee and Seung preserves non-negatively provided that the initial guess for the vectors u^m, v^m, w^m are non-negative:

$$u_i^j \leftarrow \frac{u_i^j \sum_{s,t} G_{i,s,t} v_s^j w_t^j}{\sum_{m=1}^{k} u_i^m < v^m, v^j >< w^m, w^j >}$$

$$v_i^j \leftarrow \frac{v_i^j \sum_{r,t} G_{r,i,t} u_r^j w_t^j}{\sum_{m=1}^{k} v_i^m < u^m, u^j >< w^m, w^j >}$$

$$w_i^j \leftarrow \frac{w_i^j \sum_{r,s} G_{r,s,i} u_r^j v_r^j}{\sum_{m=1}^{k} w_i^m < u^m, u^j >< v^m, v^j >}$$

Results



m = 1;

m = 5;

m = 10;

m = 50;

m = 100;

original

Problems

- NTF is iterative method and number of iterations and rank of factorization is not clear as well as in NMF.
- The converge process has not been studied yet by me.