

IPCV 2006

Summer School on Computer Vision

Dietrich Paulus and Wolfram Hans

Color Histogram Algorithms

27th August 2006
Universität Koblenz-Landau
Computervisualistik
Sommersemester / Summer 2002

In this course of two lectures we will discuss color histograms and their use for object recognition. This will lead us also to some related subjects, namely color normalization, color sensor calibration, and simple classification. We will introduce color histograms in various flavors, resolutions, and representations. We discuss different distance measures for color histograms and compare them. We learn how to compute histograms efficiently. Histogram intersection and histogram backprojection provide very simple and yet efficient ways to find objects in scenes. These methods can also be applied to image database queries.

Some of the computations require linear algebra. As the singular value decomposition can be used to compute many numerical solutions in linear algebra, we introduce this method and apply it so color calibration and color normalization problems.

In the exercises we implement the methods that are introduced in the lecture. These methods can be used to solve the task in the project work: recognize objects from a set of known objects in a database.

The task of the project will be solved in small groups (international members) of three students. The groups are free to choose their individual solution. On the final day of the summer school the groups will present their solution. In a contest we will determine the best solution.

Contents

4	Color Histograms	5
5	Color Normalization	15
6	Color Calibration	21
6.1	Color Constancy	21
6.2	Color Checker	23
6.3	Color Calibration	23
7	Classification	27
7.1	General Notes on Classifiers	28
7.2	Design of Classifiers	28
7.3	Linear Discriminants	28
7.4	Polynomial Classifiers	31
7.5	Bayesian Classifiers	31
7.6	From Bayesian to Geometric Classifiers	32
7.7	Nearest Neighbor Classifier	34
7.8	Testing a Classifier	35
7.8.1	Lerning and Testing	35
A	Project Budapest 2006	37
B	SVD	41
	Annotated Bibliography	45
B.1	Names and Symbols	47

This course material is a re-formatted version of the slides that are presented in the course. It is not a **book**. You will need additional notes or articles to understand the topics.

Chapter 4

Color Histograms

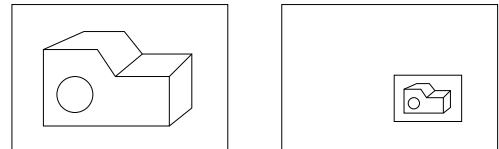
Object localization: common task in computer vision

Color as a cue for solving this problem presented by Swain and Ballard in [SB91]: histogram intersection and histogram backprojection

→ *distributions of color information, for object localization*

→ color histograms

General ideas:

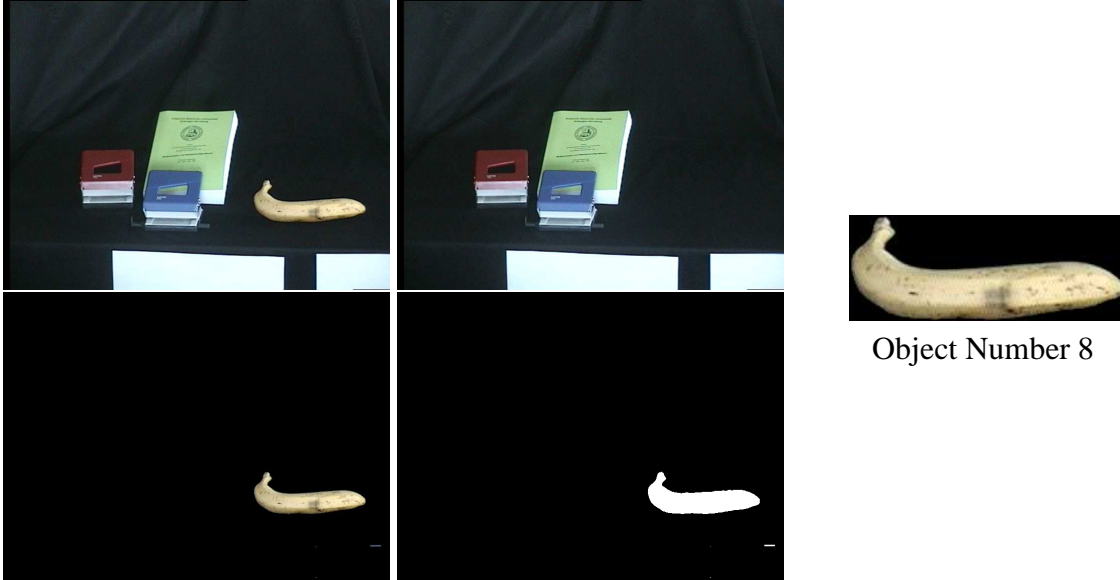


- *try to find that sub-image of the scene image, for which the histogram has the smallest distance from the object histogram.*

Size $N_M \times M_M$ of subimage has to be determined somehow!

- color histogram backprojection

Example:



- capture image f of some objects with known focal length
- add one object κ to scene
- capture image g
- $|f - g|$, threshold, erode, filter to get mask h
- extract object as $f \& h$, compute histogram T
- automatic detection of bounding box
- Sample: 20 objects in 3 illuminations

Color Histograms

Assume a close-up view of object given in the

image $f = [f_{ij}]_{i=1\dots M, j=1\dots N}$

where f_{ij} is a color pixel, i.e. $f_{ij} = (r_{ij}, g_{ij}, b_{ij})^T$.

This object is to be found in the scene f' .

We compute histograms S for a subimage of the scene and T for the object

$$S = [S_l]_{l=1\dots N_L} \quad T = [T_l]_{l=1\dots N_L} \quad (4.1)$$

where the number of bins N_L depends of the chosen quantization and the color space.

Histogram computation

Function ζ maps color pixel to the index in histogram in any color space

Example: for a RGB histogram with $4 \times 4 \times 4$ bins ($N_L = 64$) and for color components in the range from 0 to 255

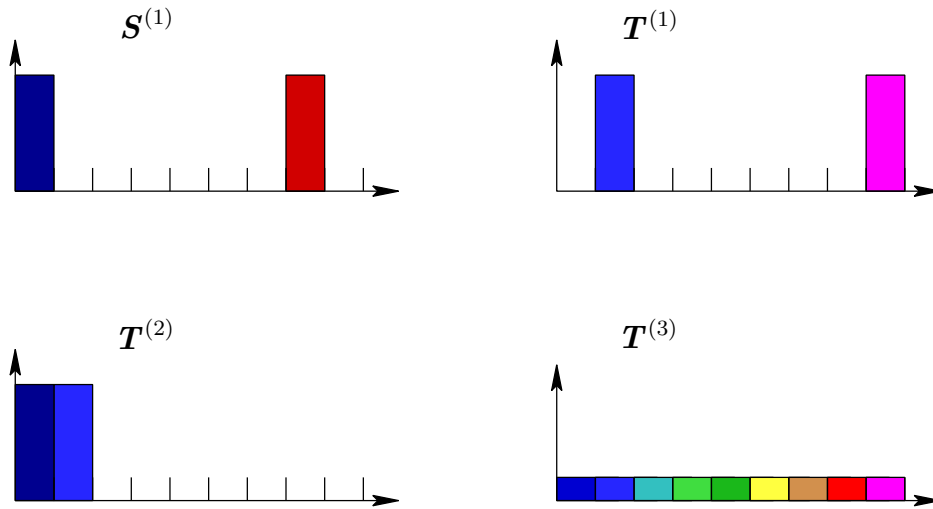
$$\zeta : \begin{cases} \mathbb{R}^3 & \rightarrow \{1, \dots, N_L\} \\ \mathbf{f}_{ij} = (r_{ij}, g_{ij}, b_{ij})^T & \rightarrow \lceil r_{ij}/64 \rceil * 16 + \lceil g_{ij}/64 \rceil * 4 + \lceil b_{ij}/64 \rceil \end{cases} \quad (4.2)$$

Elements of the histogram \mathbf{T}

$$T_l = |\{(i, j) | \zeta(\mathbf{f}_{ij}) = l, i = 1 \dots M_T, j = 1, \dots, N_T\}| \quad (4.3)$$

Analogously: \mathbf{S}, S_l for an image of size: $M_S \times N_S$

In order to compare the histograms we need distance measures



Histogram Intersection - a classical measure proposed in [SB91]. It is computationally inexpensive generalization of geometric L_1 Minkowski's distance defined by:

$$\cap(\mathbf{S}, \mathbf{T}) = \sum_{l=1}^{N_L} \min\{S_l, T_l\} \quad (4.4)$$

Extended to *focused intersection* and *active search* in [VMH97]

Active Search

- sliding sums algorithm for histogram comparison (computation independent of size of window)

- for those distance measures fulfilling triangle equation: cancel comparison when current optimum cannot be met

(this can be predicted e.g. for histogram intersection, [VMH97])

Sum of Squared Differences (SSD) - defined by:

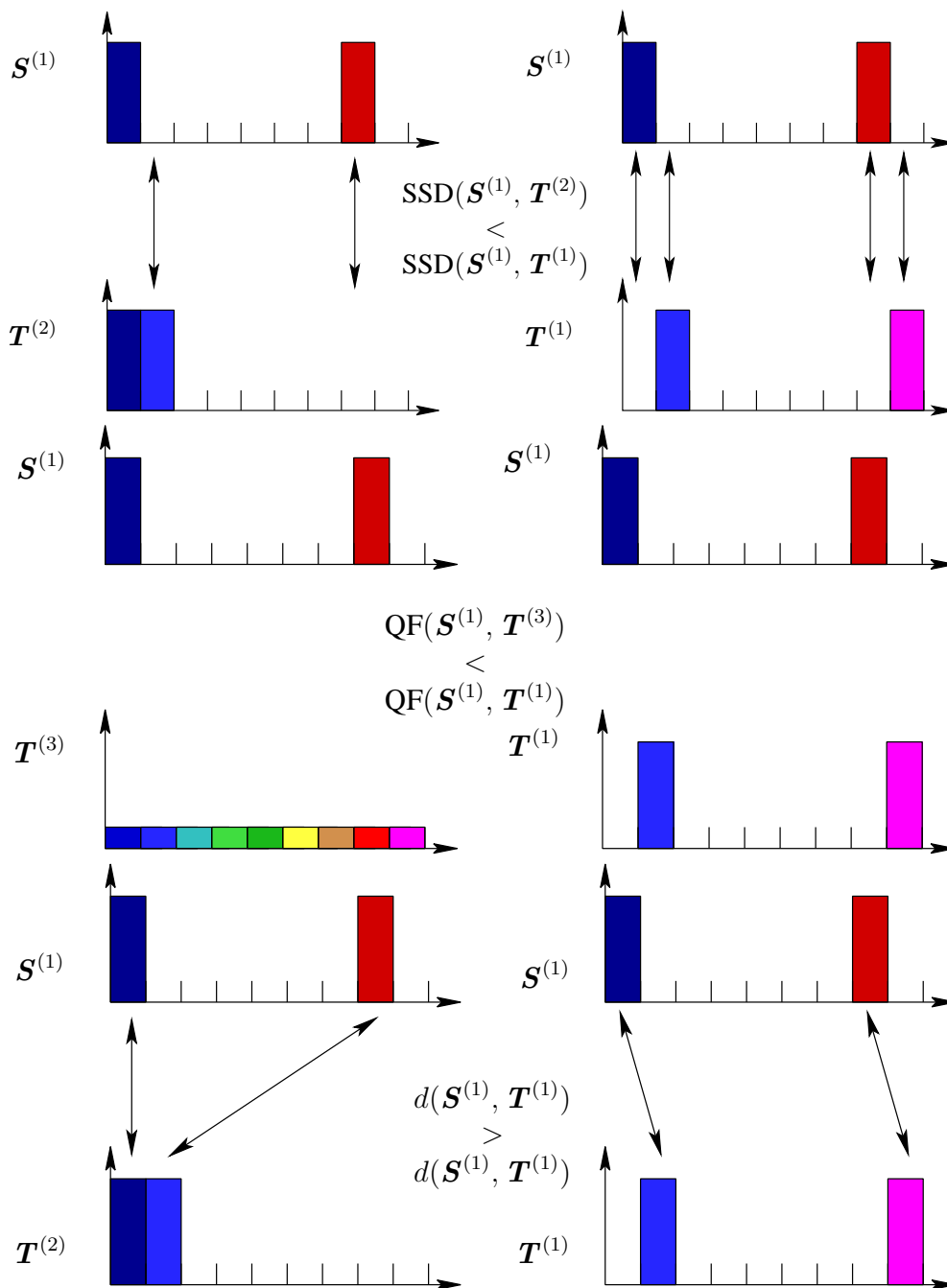
$$SSD(\mathbf{S}, \mathbf{T}) = \sum_{l=1}^{N_L} (T_l - S_l)^2 \quad (4.5)$$

Chi-square Test - statistical method. Different versions in literature [PBRT98, Sch97, PFTV88]

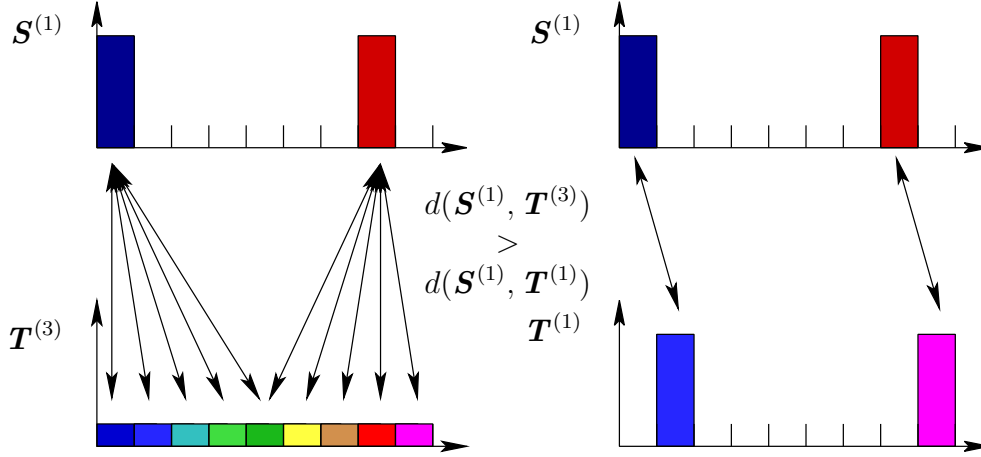
We use (like [Sch97])

$$\chi^2(\mathbf{S}, \mathbf{T}) = \sum_{l=1}^{N_L} \frac{(S_l - T_l)^2}{T_l} \quad (4.6)$$

It was shown that using this definition, the best results were obtained with respect to noise, blur, and image plane rotation [Sch97].



This picture shows the desired properties of a histogram distance for another pair of comparisons



This picture shows the desired properties of a histogram distance for another pair of comparisons

1. *Minkowski-form distance* [KKIK03] The Minkowski-form distance L_p is a generalised metric distance. Depending on its order p , it represents different distance functions. As the level of p decreases, the weight of large differences between single attribute values increases. The distance d between \mathbf{x} and \mathbf{y} is defined as

$$L_p(\mathbf{x}, \mathbf{y}) = \sqrt[p]{\sum_{i=1}^n |\mathbf{x}_i - \mathbf{y}_i|^p}, \quad \mathbf{x}, \mathbf{y} \in \mathbb{R}^n \quad (4.7)$$

The computational complexity of the Minkowski-distances depends on the selection of p . Specifications of the Minkowski-metric are the following distance measures:

- *absolute distance* (also referred to as City-Block or Manhattan distance)

$$d(\mathbf{x}, \mathbf{y}) = L_1(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^n |\mathbf{x}_i - \mathbf{y}_i| \quad (4.8)$$

- *Euclidean distance*

$$d(\mathbf{x}, \mathbf{y}) = L_2(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\| = \sqrt{\sum_{i=1}^n (\mathbf{x}_i - \mathbf{y}_i)^2} \quad (4.9)$$

2. χ^2 -statistic [Pap02]

$$d(\mathbf{x}, \mathbf{y}) = \chi_1^2(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^n \frac{(\mathbf{x}_i - \mathbf{y}_i)^2}{\mathbf{x}_i} \quad (4.10)$$

$$d(\mathbf{x}, \mathbf{y}) = \chi_2^2(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^n \frac{(\mathbf{x}_i - \mathbf{y}_i)^2}{\mathbf{x}_i + \mathbf{y}_i} \quad (4.11)$$

3. *Kullback-Leibler divergence* [KL51]

$$d(\mathbf{x}, \mathbf{y}) = KL(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^n x_i \log \frac{x_i}{y_i} \quad (4.12)$$

4. *Jeffrey divergence* (also referred to as Jensen-Shannon divergence) [PBRT99]

$$d(\mathbf{x}, \mathbf{y}) = JF(\mathbf{x}, \mathbf{y}) = \left(\sum_{i=1}^n x_i \log \frac{2x_i}{x_i + y_i} + y_i \log \frac{2y_i}{x_i + y_i} \right) \quad (4.13)$$

5. *Bhattacharyya distance* [Kai67]

$$d(\mathbf{x}, \mathbf{y}) = BA(\mathbf{x}, \mathbf{y}) = \sqrt{1 - \rho[\mathbf{x}, \mathbf{y}]} \quad \text{where} \quad \rho[\mathbf{x}, \mathbf{y}] = \sum_{i=1}^n \sqrt{x_i y_i} \quad (4.14)$$

Histogram backprojection

Histogram backprojection

Given: image histogram $\mathbf{T} = [T_k]_{k=1\dots K}$ of an object,
Wanted: object position (i_t, j_t)
Compute color histogram $\mathbf{H} = [H_k]_{k=1\dots K}$ of given image
FOR Each bin $k \in \{1, \dots, K\}$
$R_k = \min\{\frac{T_k}{H_k}, 1\}$ (compute ratio histogram $\mathbf{R} = [R_k]_{k=1\dots K}$)
FOR All positions (i, j) in the image
$A_{i,j} := R_h(\mathbf{f}_{i,j})$, where $\mathbf{f}_{i,j}$ denotes the color vector at position (i, j)
$\mathbf{B} := \mathbf{D}_r \star \mathbf{A}$, where \star denotes convolution
$(i_t, j_t) := \operatorname{argmax}_{i,j}(B_{i,j})$

\mathbf{D}_r denotes a mask of the size of the object

Let N_o Objects $\{O_1, \dots, O_{N_o}\}$ be given, deren Auftrittswahrscheinlichkeiten durch die a-priori Wahrscheinlichkeiten der Objekte $p(O_i)$ beschrieben wird. Für jedes Objekt O_i gibt die bedingte Wahrscheinlichkeit $p(\mathbf{f}|O_i)$ ($i \in \{1, \dots, N_o\}$) die Wahrscheinlichkeiten der Merkmale — in diesem Fall der Farbvektoren \mathbf{f} — an, die durch Histogramme von Bildern der Objekte geschätzt werden. Gegeben ist nun ein Bild von Farbpixeln \mathbf{f} , bzw. die Verteilung der Farbwerte $p(\mathbf{f})$ als Histogramm. Gesucht ist die Position des Objekts im Bild. Die Umformung der Formel für die bedingte Wahrscheinlichkeit

$$p(O_i)p(\mathbf{f}|O_i) = p(\mathbf{f})p(O_i|\mathbf{f})$$

in

$$p(O_i|\mathbf{f}) = p(O_i) \frac{p(\mathbf{f}|O_i)}{p(\mathbf{f})}$$

ergibt nun die Wahrscheinlichkeit eines Objekts bei beobachtetem Farbvektor \mathbf{f} als die a-priori Wahrscheinlichkeit des Objekts mal einem Quotienten. Der Quotient wird durch das Verhältnishistogramm geschätzt. Die a-priori Wahrscheinlichkeit $p(O_i)$ des Objekts wird für jeden Ort im Bild als gleich angenommen und muss daher in der Maximierung nicht berücksichtigt werden.

New Distance Measures

Computation of EMD involves the solution of the *transportation problem*, where a matrix $\mathbf{F} = [F_{\mu,\nu}]_{1,\dots,N_L,1,\dots,N_L}$ represents the flow from bin S_μ to T_ν

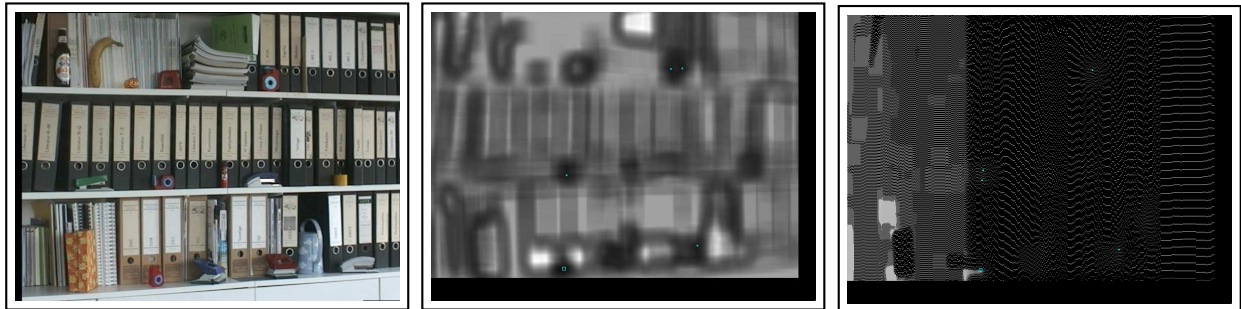
Constraints:

The flow cannot be negative ($0 \leq F_{\mu,\nu}$) and is further constrained by:

$$\sum_{\mu=1}^{N_L} F_{\mu\nu} = T_\nu / (M_T N_T) \quad \sum_{\nu=1}^{N_L} F_{\mu\nu} = S_\mu / (M_S N_S) \quad \sum_{\mu=1}^{N_L} \sum_{\nu=1}^{N_L} F_{\mu\nu} = 1 \quad (4.15)$$

It makes EMD the most computationally expensive among exploited measures. It can be implemented by means of linear programming methods, such as the simplex algorithm.¹ (requires $\mathcal{O}(N_L^4)$ memory)

Feasible: $N_L \leq 64$



Left: scene, middle: histogram intersection, right: active search

Result of active search for object number 11. Only 13% of the total number of comparisons is required for active search

We tested:

- various color spaces
- various quantizations
- same scene under different illuminations

¹We used the Y. Rubner's code for EMD found at <http://robotics.stanford.edu/~rubner/emd/default.htm>

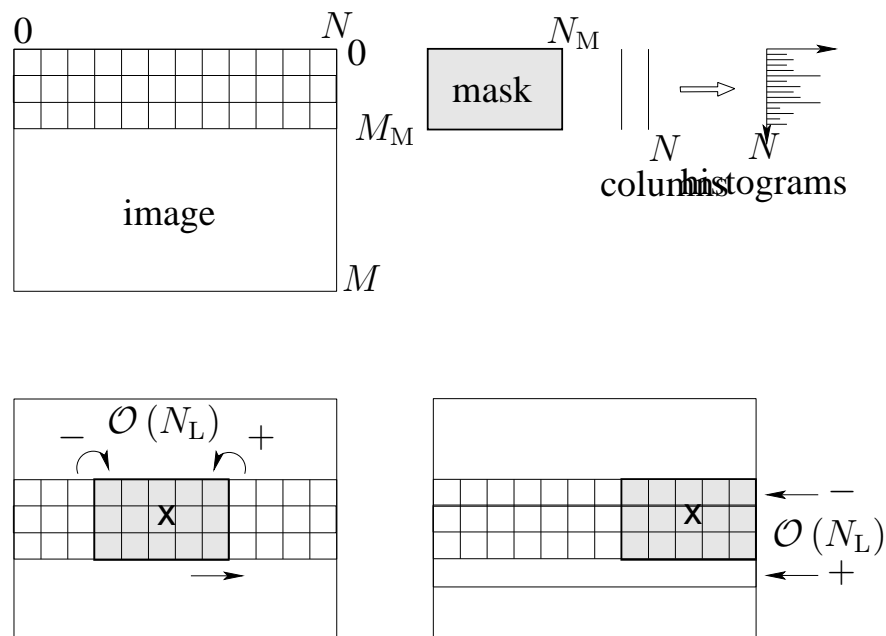


Figure 4.1: Fast computation of histograms. Top: Initialization. left bottom: Advance by one pixel. right bottom: Advance one line.

- SSD, χ^2 , HI, EMD, DP, Kullback-Divergenz, ...
- ≈ 20 objects and 20 Scenes
- altogether several thousands of experiments

Chapter 5

Color Normalization

Problem definition

Color image processing:

Color changes due to changes in lighting and position

Many algorithms are *very* sensitive to color changes

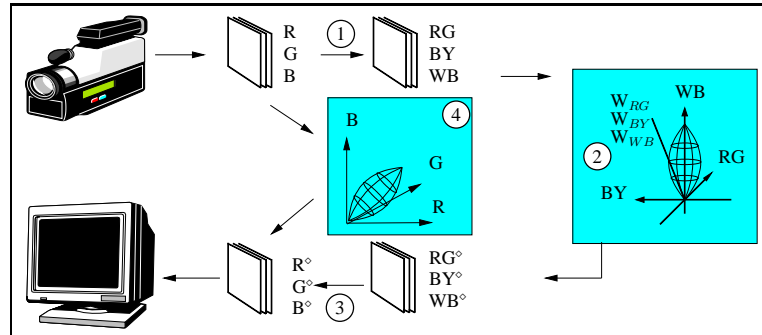
Goal 1: invariance of image processing results with respect to lighting changes by *normalization*

Goal 2: normalization without user interaction or knowledge about the problem domain *data driven*

Outline: "gray-world assumption": the world is gray (in average). If the reality (the image) is not gray, we make it gray. In *RGB*, this means we produce a color distribution which is clustered around the main diagonal of the *RGB* cube.

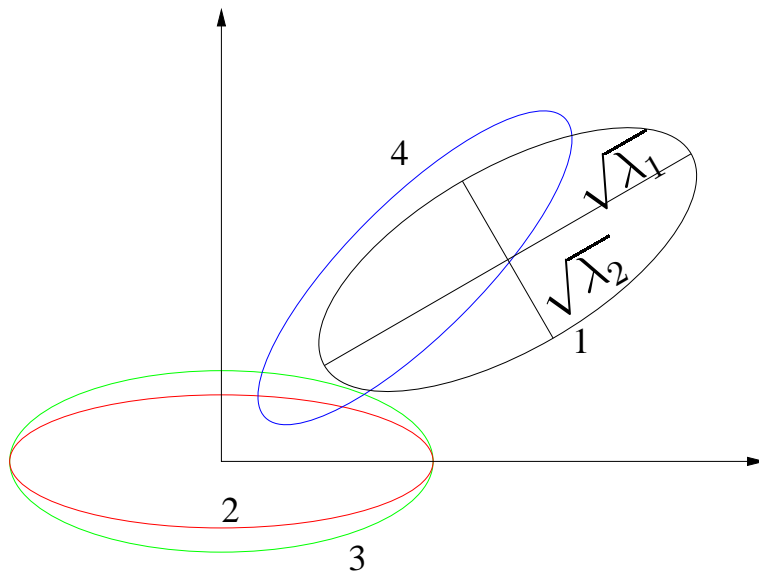
Algorithm: do this transformation in a linear transformation of the *RGB* space, then return to *RGB*

- *Given*: $\tilde{\mathbf{f}} = (RG, BY, WB)^T \in \mathbb{R}^3$ (some color space)
 \mathbf{A} : (3×3) -transformation matrix *RGB* – *RG, BY, WB* ($\tilde{\mathbf{f}} = \mathbf{A}\mathbf{f}$)
- *Wanted*: Principal component (PC) of color vectors $\tilde{\mathbf{f}}$
- *Principle*: eigenvector for greatest eigenvalue of
$$\tilde{\mathbf{C}} = E\{\tilde{\mathbf{f}}\tilde{\mathbf{f}}^T\}$$
- *Solution for PC analysis*: ANN [PG95] as proposed in [OP84]



Geometric interpretation

In \mathbb{R}^2 :



In \mathbb{R}^3 : $[0, \dots, 255]^3 \subset \mathbb{R}^3$:

$\lambda_i, i \in 1, 2, 3$: eigenvalues

Scaling matrix

$$A = \begin{pmatrix} \frac{255}{\sqrt{\lambda_1}} & 0 & 0 \\ 0 & \frac{255}{\sqrt{\lambda_2}} & 0 \\ 0 & 0 & \frac{255}{\sqrt{\lambda_3}} \end{pmatrix}$$

converts club to bowl

- 1 \rightarrow 2 : PCA
- 2 \rightarrow 3 : scaling and normalization How to compute the PCA of color vectors f
- 3 \rightarrow 4 : pose normalization

1. compute mean of the vectors $m = E\{f\}$ where $E\{\cdot\}$ denotes the expectation value

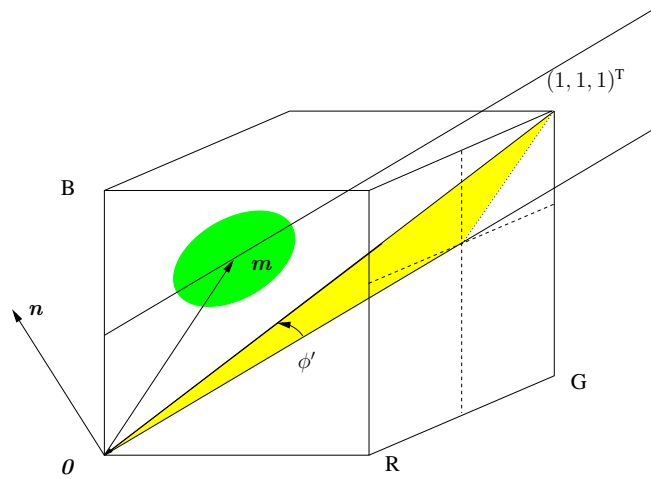


Figure 5.1: Color rotation

2. compute covariance matrix C of the vectors by $C = E\{(\mathbf{f} - \mathbf{m})(\mathbf{f} - \mathbf{m})^T\}$
3. do an Eigenvalue analysis of C
4. read the math books how to do this

How to rotate around an axis \mathbf{n} by an angle ϕ : $\mathbf{R}_3(\phi, \mathbf{n}) = \mathbf{I} - \sin \phi \mathbf{U}(\mathbf{n})$
 use Rodrigues formula (see math books)

New idea

1. No change in color space:

$$\mathbf{m} = E\{\mathbf{f}\} \quad \text{and} \quad \mathbf{C} = E\{(\mathbf{f} - \mathbf{m})(\mathbf{f} - \mathbf{m})^T\}$$

2. \mathbf{C} is (3×3) -matrix \rightarrow eigenvalues (sorted) eigenvector[1] $(a, b, c)^T \rightarrow \mathbf{n}' \quad \mathbf{n}' = (a, b, c)^T \times \frac{1}{\sqrt{3}}(1, 1, 1)^T$

3. $\cos \phi' = (a, b, c)^T \cdot \frac{1}{\sqrt{3}}(1, 1, 1)^T$

4. $\mathbf{R}_3(\phi, \mathbf{n}) = \mathbf{I} - \sin \phi \mathbf{U}(\mathbf{n}) + (1 - \cos \phi) \mathbf{U}^2(\mathbf{n})$
 $\mathbf{U}^2(\mathbf{n}) = \mathbf{n}\mathbf{n}^T - \mathbf{I}$

$$\mathbf{U}(\mathbf{n}') = \frac{\sqrt{3}}{3} \begin{pmatrix} 0 & b-a & c-a \\ a-b & 0 & c-b \\ a-c & b-c & 0 \end{pmatrix}$$

5.
 - $\mathbf{m} \rightarrow O$ (shift to origin)
 - $\mathbf{R}_3(\phi', \mathbf{n}')$ (rotate)

- $O \rightarrow \frac{\|m\|}{\cos \phi'}(1, 1, 1)^T$ (shift back on diagonal)

This algorithm explains quite a lot of small details:

1. How to rotate by an angle around an axis in 3-D
2. How to apply the Rodrigues Formula
3. How to interpret Eigenvalues in geometric context
4. How to use dot products to express angles

All this is very important for other problems in computer vision that deal with 3-D transformations, such as all the problems related to 3-d reconstruction.

Comprehensive Color Normalization (CCN)

Eliminate changes which are due to changes in light position and light color/intensity

Algorithm (below) works iteratively

Step 1 (local) eliminates intensity changes

Step 2 (global) eliminates color changes due to color changes of the light

Assumes that intensity at a point (i, j) only depends on the object and the light source — and not on the other objects

Definitions:

CompNorm(f) at time $t, t + 2, t + 4, \dots$ color image at time t $f^{(t)} = [f_{ij}^{(t)}]_{i=1\dots N, j=1\dots M}$ color vector at time (t) $f_{ij}^{(t)} = (r_{ij}^{(t)}, g_{ij}^{(t)}, b_{ij}^{(t)})^T$

Iterative algorithm:

1. (local) $S_{ij} := r_{ij}^{(t)} + g_{ij}^{(t)} + b_{ij}^{(t)}$

$$r_{ij}^{(t+1)} = r_{ij}^{(t)} / S_{ij}$$

$$g_{ij}^{(t+1)} = g_{ij}^{(t)} / S_{ij}$$

$$b_{ij}^{(t+1)} = b_{ij}^{(t)} / S_{ij}$$

2. (global) $\hat{R} = \frac{3}{N * M} * \sum_1^N \sum_1^M r_{ij}^{(t+1)}$

$$r_{ij}^{(t+2)} = r_{ij}^{(t+1)} / \hat{R}$$

$$\hat{G} = \frac{3}{N * M} * \sum_1^N \sum_1^M g_{ij}^{(t+1)}$$

$$g_{ij}^{(t+2)} = g_{ij}^{(t+1)} / \hat{G}$$

$$\hat{B} = \frac{3}{N * M} * \sum_1^N \sum_1^M b_{ij}^{(t+1)}$$

$$b_{ij}^{(t+2)} = b_{ij}^{(t+1)} / \hat{B}$$

3. (stop) when $\sum_1^N \sum_1^M \left((r_{ij}^{(t+2)} - r_{ij}^{(t)})^2 + (g_{ij}^{(t+2)} - g_{ij}^{(t)})^2 + (b_{ij}^{(t+2)} - b_{ij}^{(t)})^2 \right) < \epsilon$

Properties

- Convergence proved in [FSC98]
- Uniqueness: $(\text{CompNorm}(\mathbf{f}_1) = \text{CompNorm}(\mathbf{f}_2)) \rightarrow (\mathbf{f}_1 \sim \mathbf{f}_2)$

Chapter 6

Color Calibration

6.1 Color Constancy

Sensor model according to [JKS95, S. 284 ff.]:

Let the illumination have a spectral power distribution $E(\lambda)$.

Prerequisite: Any point \mathbf{x} in the image corresponds uniquely to a point in the scene.

Let the fraction of the light reflected in point \mathbf{x} by a surface be $S(\mathbf{x}, \lambda)$.

Incident light in each image point

$$S(\mathbf{x}, \lambda) \cdot E(\lambda) \quad . \quad (6.1)$$

K sensors per pixel, (e.g. one for red, green, and blue), have spectral sensitivity $R_k(\lambda)$, ($k = 1, \dots, K$).

each sensor k samples at point \mathbf{x} the following energy distribution:

$$\rho_k(\mathbf{x}) = \int_0^\infty R_k(\lambda) \cdot S(\mathbf{x}, \lambda) \cdot E(\lambda) d\lambda \quad (6.2)$$

Surface reflection

$$S(\mathbf{x}, \lambda) = \sum_{j=1}^m \sigma_j(\mathbf{x}) L_j(\lambda) \quad , \quad (6.3)$$

where $L_j(\lambda)$ are basis functions and $\sigma_j(\mathbf{x})$ the spatially varying reflection for one of the m components. Often $m = 3$. combining (6.2) with (6.3) yields for $\boldsymbol{\rho}(\mathbf{x}) = (\rho_1, \dots, \rho_K)^T$ and $\boldsymbol{\sigma}(\mathbf{x}) = (\sigma_1, \dots, \sigma_m)^T$ the linear relation

$$\boldsymbol{\rho}(\mathbf{x}) = \mathbf{A}\boldsymbol{\sigma}(\mathbf{x}) \quad , \quad (6.4)$$

where the matrix $\mathbf{A} = [A_{ij}]_{1 \leq i \leq K, 1 \leq j \leq m}$ is defined as:

$$A_{ij} = \int_0^\infty E(\lambda) L_i(\lambda) R_j(\lambda) d\lambda \quad . \quad (6.5)$$

In many cases we choose $m = K = 3$.

For two different illuminations E_1 and E_2 of a planar surface we obtain two matrices \mathbf{A}_1 and \mathbf{A}_2 . If \mathbf{A}_2 is invertible, we obtain for the two corresponding images

$$\boldsymbol{\rho}_1(\mathbf{x}) = \mathbf{A}_1 \mathbf{A}_2^{-1} \boldsymbol{\rho}_2(\mathbf{x}). \quad (6.6)$$

→ Argument for color rotation

With simplifying assumptions, such as highly band-limited sensitivity of sensors, we conclude from (6.2) for the sensor response k at location \mathbf{x}

$$\rho_k(\mathbf{x}) = E_k(\mathbf{x}) \cdot S_k(\mathbf{x}) \quad . \quad (6.7)$$

for two adjacent pixels \mathbf{x} and \mathbf{y} : (illumination assumed to be constant locally) i.e., $E_k(\mathbf{x}) = E_k(\mathbf{y})$:

$$\frac{\rho_k(\mathbf{x})}{\rho_k(\mathbf{y})} = \frac{S_k(\mathbf{x})E_k(\mathbf{x})}{S_k(\mathbf{y})E_k(\mathbf{y})} = \frac{S_k(\mathbf{x})}{S_k(\mathbf{y})} \quad (6.8)$$

taking the logarithm of (6.8) yields

$$\ln \rho_k(\mathbf{x}) - \ln \rho_k(\mathbf{y}) = \ln S_k(\mathbf{x}) - \ln S_k(\mathbf{y}) \quad . \quad (6.9)$$

→ independent of illumination!

where $E_k(\mathbf{x})$ is the ambient light, $S_k(\mathbf{x})$ is the reflectivity coefficient for sensor k .

Given: Color image $\mathbf{f} = [\mathbf{f}_{ij} = (r_{ij}, g_{ij}, b_{ij})^T]_{i=1 \dots M, j=1 \dots N}$	
1) Take logarithm	$\mathbf{h}_{ij} := (\ln r_{ij}, \ln g_{ij}, \ln b_{ij})^T$
2) Compute derivatives	<ul style="list-style-type: none"> a) $\mathbf{h}'_{ij} := \nabla^2 \mathbf{h}_{ij}$ b) $\mathbf{h}'_{ij} := (\nabla^2 G) * \mathbf{h}_{ij}$ c) $\mathbf{h}'_{m,ij} := \nabla_m \mathbf{h}_{ij}, \quad m = 1 \dots 4$
3) Describe by histogram	<ul style="list-style-type: none"> a,b) $T_{\zeta(l_1, l_2, l_3)} := \sum_{i,j} z, \quad z = \begin{cases} 1, & \text{if } \mathbf{h}'_{ij} = (l_1, l_2, l_3)^T \\ 0, & \text{otherwise} \end{cases}$ c) $T_{\zeta(l_1, l_2, l_3)} := \sum_{m=1}^4 \sum_{i,j} z, \quad z = \begin{cases} 1, & \text{if } \mathbf{h}'_{m,ij} = (l_1, l_2, l_3)^T \\ 0, & \text{otherwise} \end{cases}$
4) Histogram-Backprojection	

Figure 6.1: Color constant color indexing

Nr	Name	red	green	blue	Nr	Name	red	green	blue
1	dark skin	94	28	13	13	blue	0	0	142
2	light skin	241	149	108	14	green	64	173	38
3	blue sky	97	119	171	15	red	203	0	0
4	foliage	90	103	39	16	yellow	255	217	0
5	blue flower	164	131	196	17	magenta	207	3	124
6	bluish green	140	253	153	18	cyane	0	148	189
7	orange	255	116	21	19	white	255	255	255
8	purplish blue	7	47	122	20	light gray	249	249	249
9	moderate red	222	29	42	21	light-medium gray	180	180	180
10	purple	69	0	68	22	medium gray	117	117	117
11	yellow green	187	255	19	23	dark gray	53	53	53
12	orange yellow	255	142	0	24	black	0	0	0

Table 6.1: Names and RGB-values for areas on the color checker

[FF95] reports on experiments of different operators for computation of the derivatives. The laplacian of the Gaussian-filtered image ($\nabla^2 \mathbf{G}$) as well as the discrete derivative of all for directions (∇^2) are used. Comparison uses the ratio histogram (??) The major steps of the algorithm are shown in Figure 6.1 where the function (??) is used.

m is the index for the direction

Find McBeth Color Checker using color ratios:

1. Region segmentation
2. RAG, nodes contain color ratio
3. search for subgraph

6.2 Color Checker

Here we will see how to find the McBeth color checker using Hough transformation.

The following material has been taken from the student's thesis of Matthias Grobe, Erlangen 2000

6.3 Color Calibration

Back to (6.2):

As noted in [AF02], the discrete version of (6.2) is often written as a sum of 31 samples

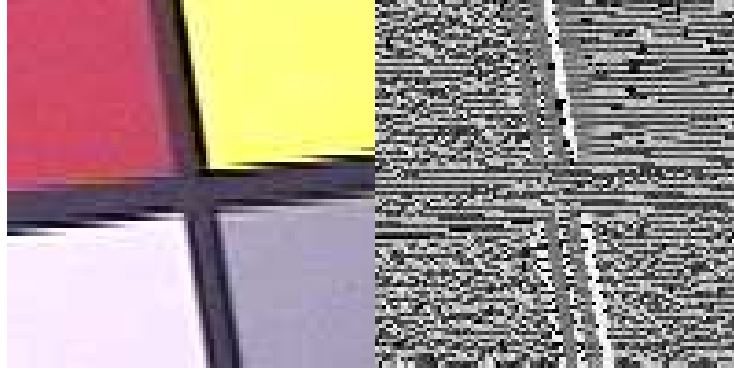


Figure 6.2: Color borders and incorrect edge directions



Figure 6.3: Parameter in hough space (inverted for display)

$$\rho_k(\mathbf{x}) = \sum_{\lambda=1}^{31} E_{\lambda} \cdot S_{\lambda}(\mathbf{x}) \cdot R_{k,\lambda} \cdot \Delta\lambda \quad , \quad (6.10)$$

which can be written in Matrix notation as

$$\boldsymbol{\rho} = \mathbf{C} \mathbf{r}.$$



Figure 6.4: Example for “5 lines on 3 lines”-criterion

Chapter 7

Classification

Sample:

- Train
- Test
- (Evaluation /verification)

Simple object recognition using color histograms:

Features: Histograms

Question: How to classify?

Question: How to learn?

The following material is taken from [PH03, Chapter 21ff]

Let us assume that the numerical feature c is a d -dimensional real-valued vector. A classifier is

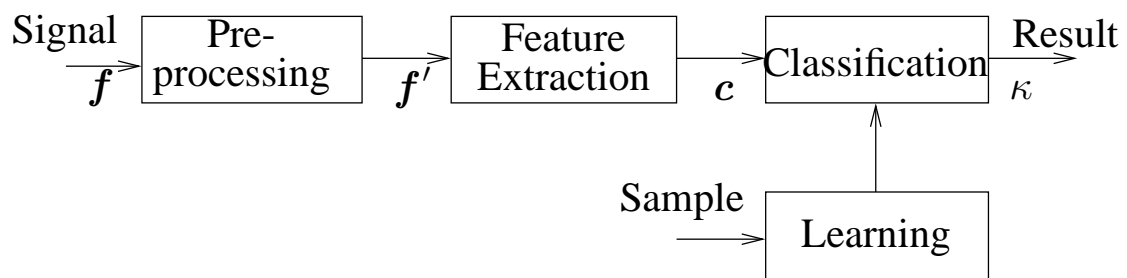


Figure 7.1: The architecture of a simple classification system [Nie83] together with the notation for signals, feature vectors, and class index

mathematically defined by the discrete mapping

$$\zeta : \begin{cases} \mathbb{R}^d & \rightarrow \{1, 2, \dots, K\} \\ \mathbf{c} & \mapsto \kappa \end{cases} \quad (7.1)$$

which assigns a class index κ , i.e., a discrete value, to a feature vector \mathbf{c} . The assignment ζ is the so-called *decision function* also called the *decision rule*.

7.1 General Notes on Classifiers

Before we start with various definitions of the decision function ζ , we have to explain some basic concepts of classification theory.

The training of classifiers based on unlabeled feature vectors is incomparably harder than the supervised case of learning [Rip96]. The classification of previously unobserved features is called the *generalization* property of classifiers.

The error probability p_B of this optimal classifier is the so-called *Bayesian error probability*. The major problem is the computation of a decision rule that results in an optimal classifier.

7.2 Design of Classifiers

The design of classifiers is based on the fundamental assumption that feature vectors of the same class have a small distance with respect to a suitably defined distance function. This measure might be a geometric distance function (e.g., the Euclidean distance), some probability measure (e.g., a posteriori probability) or others.

7.3 Linear Discriminants

The discussion of linear discriminants is first restricted to two classes, i.e., we deal only with binary classes Ω_1 and Ω_2 . This simplifies the understanding of linear classifiers, and allows an easier and clearer insight into basic concepts.

For a formal definition, we set $\mathbf{c} = (c_1, c_2, \dots, c_d)^T \in \mathbb{R}^d$ to be the observed, d -dimensional feature vector. A linear discriminant classifier applies the decision rule

$$\zeta(\mathbf{c}) = \begin{cases} 1, & \text{if } q(\mathbf{c}) > 0 \\ 2, & \text{otherwise} \end{cases}, \quad (7.2)$$

with a *splitting function*

$$q(\mathbf{c}) = q_0 + \sum_{i=1}^d q_i c_i = q_0 + (q_1, q_2, \dots, q_d) \cdot (c_1, c_2, \dots, c_d)^T, \quad (7.3)$$

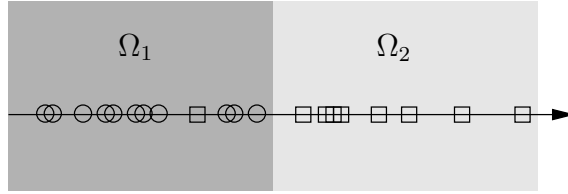


Figure 7.2: Two classes in a one-dimensional feature space

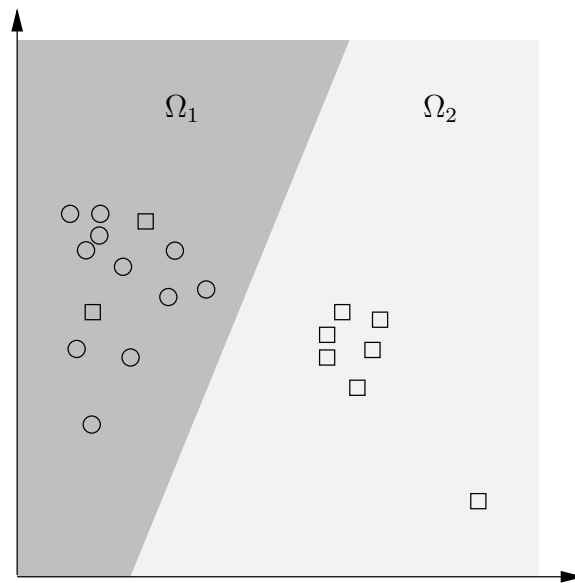


Figure 7.3: Two classes in a two-dimensional feature space

and $q_i \in \mathbb{R}$.

$$\zeta(\mathbf{c}) = \operatorname{argmax}_{\lambda} \tilde{q}_{\lambda}(\mathbf{c}) = \operatorname{argmax}_{\lambda} \left(\tilde{q}_{\lambda,0} + \sum_{i=1}^d \tilde{q}_{\lambda,i} c_i \right), \quad (7.4)$$

where \tilde{q}_{λ} denotes the *splitting polynomial* of class Ω_{λ} which is defined by the $(d+1)$ -dimensional vector $\tilde{\mathbf{q}}_{\lambda}^T = (\tilde{q}_{\lambda,0}, \tilde{q}_{\lambda,1}, \dots, \tilde{q}_{\lambda,d})$.

be the set of sample data assigned to class Ω_{κ} , i.e., we observe N_{κ} samples for classes Ω_{κ} , $\kappa = 1, 2, \dots, K$. The complete set of samples is

$$\omega = \bigcup_{\kappa=1}^K \omega_{\kappa}. \quad (7.5)$$

$$\chi_\kappa(\mathbf{c}) = \begin{cases} 1 & , \text{ if } \mathbf{c} \text{ belongs to } \Omega_\kappa \\ 0 & , \text{ otherwise } \end{cases} . \quad (7.6)$$

Let ${}^j\mathbf{c}_\lambda$ be the j -th sample feature of ω_λ which is known to belong to class Ω_λ . For each d -dimensional feature vector ${}^j\mathbf{c}_\lambda = ({}^j c_{\lambda,1}, {}^j c_{\lambda,2}, \dots, {}^j c_{\lambda,d})^\top$ ($\lambda = 1, 2, \dots, K$, and $j = 1, 2, \dots, N_\lambda$) of the training set we get K equations ($\kappa = 1, 2, \dots, K$):

$$q_\kappa({}^j\mathbf{c}_\lambda) = q_{\kappa,0} + \sum_{i=1}^d q_{\kappa,i} {}^j c_{\lambda,i} = \chi_\kappa({}^j\mathbf{c}_\lambda) , \quad (7.7)$$

which are *linear* in the coefficients of splitting functions. This system of linear equations can be written in matrix notation. For that purpose, we define extended features by

$${}^j\tilde{\mathbf{c}}_\lambda = (1, {}^j c_{\lambda,1}, {}^j c_{\lambda,2}, \dots, {}^j c_{\lambda,d})^\top$$

by just adding the component 1. This trick allows us to introduce the a matrix $\mathbf{A} \in \mathbb{R}^{D \times K(d+1)}$ which is shown below in (7.12) where

$$D = K \cdot \sum_{\kappa=1}^K N_\kappa . \quad (7.8)$$

Furthermore we set the vector $\mathbf{x} \in \mathbb{R}^{K(d+1)}$ to

$$\mathbf{x} = (q_{1,0}, q_{1,1}, \dots, q_{1,d}, q_{2,0}, q_{2,1}, \dots, q_{2,d}, \dots, q_{K,0}, q_{K,1}, \dots, q_{K,d})^\top \quad (7.9)$$

Using the above notation, the computation of linear splitting functions now corresponds to solving the system of linear equations:

$$\mathbf{A} \mathbf{x} = \mathbf{b} . \quad (7.10)$$

For real data, the probability that the vector \mathbf{b} is not in the range of the matrix \mathbf{A} , is one. Therefore, we compute the solution \mathbf{x} which minimizes the residual $\|\mathbf{A}\mathbf{x} - \mathbf{b}\|$. The coefficients \mathbf{x} of the linear splitting functions are thus given by

$$\mathbf{x} = \mathbf{A}^+ \mathbf{b} . \quad (7.11)$$

The pseudo-inverse \mathbf{A}^+ is usually computed using SVD (see Sect. ??).

$$\mathbf{A} = \begin{pmatrix}
{}^1\tilde{\mathbf{c}}_1^T & \mathbf{0}^T & \dots & \mathbf{0}^T \\
\mathbf{0}^T & {}^1\tilde{\mathbf{c}}_1^T & \dots & \mathbf{0}^T \\
\vdots & \vdots & \ddots & \vdots \\
\mathbf{0}^T & \dots & \mathbf{0}^T & {}^1\tilde{\mathbf{c}}_1^T \\
{}^2\tilde{\mathbf{c}}_1^T & \mathbf{0}^T & \dots & \mathbf{0}^T \\
\mathbf{0}^T & {}^2\tilde{\mathbf{c}}_1^T & \dots & \mathbf{0}^T \\
\vdots & \vdots & \ddots & \vdots \\
\mathbf{0}^T & \dots & \mathbf{0}^T & {}^2\tilde{\mathbf{c}}_1^T \\
\vdots & \vdots & \ddots & \vdots \\
{}^{N_1}\tilde{\mathbf{c}}_1^T & \mathbf{0}^T & \dots & \mathbf{0}^T \\
\mathbf{0}^T & {}^{N_1}\tilde{\mathbf{c}}_1^T & \dots & \mathbf{0}^T \\
\vdots & \vdots & \ddots & \vdots \\
\mathbf{0}^T & \dots & \mathbf{0}^T & {}^{N_1}\tilde{\mathbf{c}}_1^T \\
\vdots & \vdots & \ddots & \vdots \\
{}^1\tilde{\mathbf{c}}_K^T & \mathbf{0}^T & \dots & \mathbf{0}^T \\
\mathbf{0}^T & {}^1\tilde{\mathbf{c}}_K^T & \dots & \mathbf{0}^T \\
\vdots & \vdots & \ddots & \vdots \\
\mathbf{0}^T & \dots & \mathbf{0}^T & {}^1\tilde{\mathbf{c}}_K^T \\
\vdots & \vdots & \ddots & \vdots \\
{}^{N_K}\tilde{\mathbf{c}}_1^T & \mathbf{0}^T & \dots & \mathbf{0}^T \\
\mathbf{0}^T & {}^{N_K}\tilde{\mathbf{c}}_1^T & \dots & \mathbf{0}^T \\
\vdots & \vdots & \ddots & \vdots \\
\mathbf{0}^T & \dots & \mathbf{0}^T & {}^{N_K}\tilde{\mathbf{c}}_1^T
\end{pmatrix} \quad \mathbf{b} = \begin{pmatrix}
\chi_1({}^1\mathbf{c}_1) \\
\chi_2({}^1\mathbf{c}_1) \\
\vdots \\
\chi_K({}^1\mathbf{c}_1) \\
\chi_1({}^2\mathbf{c}_1) \\
\chi_2({}^2\mathbf{c}_1) \\
\vdots \\
\chi_K({}^2\mathbf{c}_1) \\
\vdots \\
\vdots \\
\chi_1({}^{N_K}\mathbf{c}_K) \\
\chi_2({}^{N_K}\mathbf{c}_K) \\
\vdots \\
\chi_K({}^{N_K}\mathbf{c}_K)
\end{pmatrix}$$

(7.12)

7.4 Polynomial Classifiers

An obvious generalization of linear discriminant classifiers is possible, if multivariate polynomials of higher degrees are used instead of linear functions. A multivariate polynomial

$$q_\lambda(\mathbf{c}) = \sum_{i_1, i_2, \dots, i_d=1}^m q_{\lambda, i_1, i_2, \dots, i_d} c_1^{i_1} c_2^{i_2} \cdots c_d^{i_d} \quad , \quad (7.13)$$

is attached to each class Ω_λ , and the decision rule (7.4) remains unchanged.

7.5 Bayesian Classifiers

The a posteriori probability

$$p(\Omega_\kappa | \mathbf{c}) = \frac{p(\Omega_\kappa) p(\mathbf{c} | \Omega_\kappa)}{p(\mathbf{c})} = \frac{p(\Omega_\kappa) p(\mathbf{c} | \Omega_\kappa)}{\sum_{\lambda=1}^K p(\Omega_\lambda) p(\mathbf{c} | \Omega_\lambda)} \quad . \quad (7.14)$$

summarizes the probability that the class Ω_κ is present, if the feature vector \mathbf{c} is observed. This discrete measure is the basic component of the Bayesian classifier and its decision rule.

$$\zeta(\mathbf{c}) = \operatorname{argmax}_\lambda p(\Omega_\lambda | \mathbf{c}) = \operatorname{argmax}_\lambda p(\Omega_\lambda) p(\mathbf{c} | \Omega_\lambda) \quad . \quad (7.15)$$

The principle of Bayesian classifiers is illustrated in Figure 7.4.

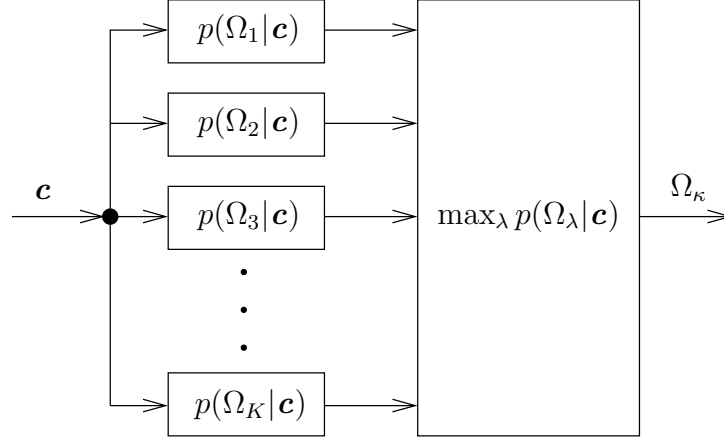
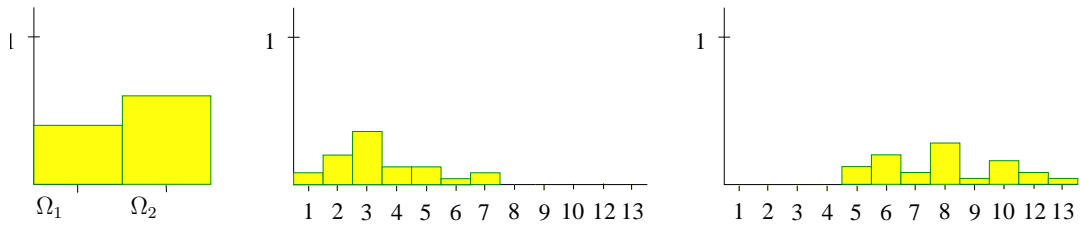


Figure 7.4: The principle of the Bayesian classifier

Figure 7.5: A priori probabilities (left), probabilities of scalar discrete features corresponding to class Ω_1 (middle), and class Ω_2 (right)

$$\begin{aligned} \zeta(\mathbf{c}) &= \operatorname{argmax}_\lambda p(\Omega_\lambda | \mathbf{c}) = \operatorname{argmax}_\lambda p(\Omega_\lambda) p(\mathbf{c} | \Omega_\lambda) \\ &= \operatorname{argmax}_\lambda p(\mathbf{c} | \Omega_\lambda) . \end{aligned} \quad (7.16)$$

$$p(\mathbf{c} | \Omega_\kappa) = \frac{1}{\sqrt{|\det 2\pi \boldsymbol{\Sigma}_\kappa|}} e^{-\frac{(\mathbf{c} - \boldsymbol{\mu}_\kappa)^\top \boldsymbol{\Sigma}_\kappa^{-1} (\mathbf{c} - \boldsymbol{\mu}_\kappa)}{2}} . \quad (7.17)$$

7.6 From Bayesian to Geometric Classifiers

By specialization, the Bayesian classifier which uses normally distributed features can be reduced to a simple classifier. In this special classifier, decision making depends on Euclidean distances only. Let us consider two classes Ω_1 and Ω_2 .

The discrete prior probabilities $p(\Omega_1)$ and $p(\Omega_2)$ as well as the Gaussian densities $p(\mathbf{c}|\Omega_1)$ and $p(\mathbf{c}|\Omega_2)$ are assumed to be known. In this situation the Bayesian classifier decides for class Ω_1 if

$$p(\Omega_1)p(\mathbf{c}|\Omega_1) > p(\Omega_2)p(\mathbf{c}|\Omega_2) \quad . \quad (7.18)$$

Now we specialize this decision rule by setting

$$\Sigma_1 = \Sigma_2 = \Sigma \quad ; \quad (7.19)$$

taking the logarithm of both sides in (7.18), we get

$$\log p(\Omega_1) + \boldsymbol{\mu}_1^T \Sigma^{-1} \mathbf{c} - \frac{1}{2} \boldsymbol{\mu}_1^T \Sigma^{-1} \boldsymbol{\mu}_1 > \log p(\Omega_2) + \boldsymbol{\mu}_2^T \Sigma^{-1} \mathbf{c} - \frac{1}{2} \boldsymbol{\mu}_2^T \Sigma^{-1} \boldsymbol{\mu}_2. \quad (7.20)$$

This decision rule proves the following important result: both splitting functions are linear in the components of the feature vector, and thus the Bayesian classifier reduces to a *linear* splitting function if the features are normally distributed and have the same covariance matrix. The coefficients of the polynomials are defined by functions of prior probabilities, of the mean vectors, and of the covariance matrix. For practical applications, this observation has an important consequence. For d -dimensional feature vectors, the linear splitting functions require the estimation of $K(d+1)$ parameters in the presence of K pattern classes. In contrast, the use of normal distributions expects the computation of $d(d+1)/2 + K(d+1)$ parameters. Therefore, for high-dimensional feature vectors and small values of K , the learning of linear splitting function parameters might lead to more robust estimates.

A further specialization results from uniform priors:

$$p(\Omega_1) = p(\Omega_2) \quad . \quad (7.21)$$

Using this constraint we obtain the discriminant

$$\boldsymbol{\mu}_1^T \Sigma^{-1} \mathbf{c} - \frac{1}{2} \boldsymbol{\mu}_1^T \Sigma^{-1} \boldsymbol{\mu}_1 > \boldsymbol{\mu}_2^T \Sigma^{-1} \mathbf{c} - \frac{1}{2} \boldsymbol{\mu}_2^T \Sigma^{-1} \boldsymbol{\mu}_2 \quad (7.22)$$

which is a simplified version of the

$$(\mathbf{c} - \boldsymbol{\mu}_1)^T \Sigma^{-1} (\mathbf{c} - \boldsymbol{\mu}_1) < (\mathbf{c} - \boldsymbol{\mu}_2)^T \Sigma^{-1} (\mathbf{c} - \boldsymbol{\mu}_2) \quad . \quad (7.23)$$

For implementation purposes, of course, we prefer (7.22) to (7.23) because these splitting functions are linear in the components of \mathbf{c} .

If we additionally assume that the covariance matrix Σ is the identity matrix, the classification is based on the inequality

$$(\mathbf{c} - \boldsymbol{\mu}_1)^T (\mathbf{c} - \boldsymbol{\mu}_1) < (\mathbf{c} - \boldsymbol{\mu}_2)^T (\mathbf{c} - \boldsymbol{\mu}_2) \quad . \quad (7.24)$$

This decision rule compares the squared Euclidean distances

$$\|\mathbf{c} - \boldsymbol{\mu}_1\|^2 < \|\mathbf{c} - \boldsymbol{\mu}_2\|^2, \quad (7.25)$$

between feature and mean vectors, i.e., class centers.

The final specialization has shown that statistical classifiers lead to a simple distance measure for restricted statistical assumptions. The minimum distance classifier with respect to class dependent mean vectors is optimal if the feature vectors used are normally distributed with covariance matrices $\boldsymbol{\Sigma}$ that are equal to the identity matrix.

Classifiers based on parametric densities have the disadvantage that a parametric distribution of the feature vectors must be known. This causes some problems, especially for features that are not normally distributed. In general, there exist three possibilities to verify a density assumption:

1. the distribution of features is known by construction,
2. the hypothesized parametric density is proven by statistical tests or
3. the recognition rate of the resulting classifier suggests the correctness.

The use of the Euclidean distance to a reference vector — the mean vector of each class — motivates the introduction of nearest neighbor classifiers. Instead of computing mean vectors for each class and using a distance measure to mean vectors for discrimination, we utilize all observed feature vectors of the training set as a reference. The resulting classifier is the nearest neighbor classifier.

7.7 Nearest Neighbor Classifier

The nearest neighbor classifier requires a set of classified sample data, i.e., for each element \mathbf{c}_i of $\mathcal{C} = \{\mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_n\}$ the class number $\zeta(\mathbf{c}_i)$ is known. For a new feature vector, the class that points to the reference vector with the closest distance to the new vector is chosen. Thus, the decision rule is defined by

$$\zeta(\mathbf{c}) = \operatorname{argmin}_{\zeta(\mathbf{c}_i)} \{\|\mathbf{c} - \mathbf{c}_i\| \mid i = 1, 2, \dots, n\}. \quad (7.26)$$

$$p_B \leq p_{NN} \leq 2p_B, \quad (7.27)$$

There exist various modifications of the nearest neighbor classifier.

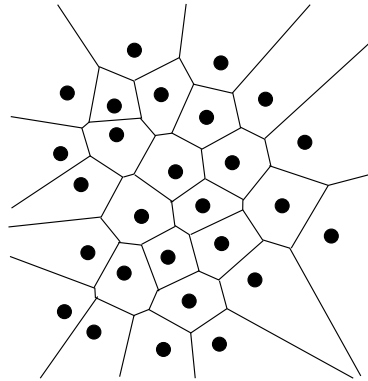


Figure 7.6: Voronoi diagram with 27 two-dimensional reference vectors

FOR $\forall \omega \in \Omega_p \cup \Omega_n$
Train classifier with $\Omega_p \setminus \omega$ and $\Omega_n \setminus \omega$
Classify ω and record recognition rates
Average recognition of the previous tests

Figure 7.7: Leave one out Strategy

7.8 Testing a Classifier

Now as we have explained our classifier, how do we examine its properties? Measures characterizing a classifier are recognition rates or probabilities of mis-classification. In a two-class problem for positive and negative samples often four numbers are given to describe a classification result:

1. the rate or probability of correctly classifying positive samples (true positive)
2. the rate or probability of incorrectly classifying negative samples as positive (false positive)
3. the rate or probability of correctly recognizing negative samples (true negative)
4. the rate or probability of incorrectly classifying positive samples as negative (false positive)

N-Fold cross validation

7.8.1 Learning and Testing

- Have two sets: training and test set (need to be disjoint!)
- leave one out
- n-fold cross validation

Appendix A

Project Budapest 2006

Object Recognition Contest

Wolfram Hans

Dietrich Paulus

Goal

Given an image

→

Recognize object

Restrictions

- Small set of possible objects (approx. 12)
- Only planar objects (2D processing)
- Homogeneous background

Problems

- size / orientation changes
- illumination changes
- different background

Some Details on the Strategy

Given an image ...

- try to isolate image from background
- compute so-called *features* (e.g. mean of the object region)

Given several images of one object type (a so-called *class*) ...

- compute features of each image
- store set of features somehow for that class

Given one images and a set of known classes

- compute features for the image
- compute „distance” of features to all known feature sets
- select „closest” set

Details will be given in the lectures

Organization

- Work in small groups (3 students)
- Groups should be multi-national
- Approach is up to you – we only give hints
- During the two weeks
 - you will get to know nice features in the lectures
 - implement them in the exercises and project work
 - create a tool box for feature extraction
- Contest on last friday

Example images will be provided.

- size, orientation may vary
- image size may be different
- background may be different or textured

Hint: start working on the simple cases!

Schedule

- Mo: build groups – view sample images
- Tu-Fr: start implementing feature extractors
- Fr: create image database
- Mo-Tu: work on the database
- We: Define test framework
- Th: learn about color histograms, integrate them
- Fr: contest

Hints:

- Start to work *today!*
- Images will be in color
- First tasks:
 - separate object from background or identify background
 - compute simple features to test your system (e.g. mean color)
 - plot the feature vectors (e.g. with gnuplot) to see whether
 - * match for object of one class
 - * they differ for different classes

Appendix B

SVD

- Powerful normal form for matrices
- Method of numerical linear algebra
 - invented in the 19th century,
 - rediscovered and pushed for practical application by Gene Golub,
 - established in computer vision by Tomasi's factorization algorithm [TK92]
- Advantage: SVD can be applied to solve most problems in linear algebra (not always in an optimal manner)
 - solution of overdetermined of linear equations
 - computation of condition numbers
 - enforcing rank criterion, etc.

SVD known to image processors since [TK92]

POOR MAN's SVD

Every $M \times N$ matrix A can be decomposed into a product

$$A = U D V^T ,$$

where

- $U \in \mathbb{R}^{M \times M}$, $V \in \mathbb{R}^{N \times N}$ square and orthogonal,
- $D \in \mathbb{R}^{M \times N}$ diagonal matrix.

With D :

$$D = \begin{pmatrix} \sigma_1 & 0 & 0 & 0 \\ 0 & \sigma_2 & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & \dots & \sigma_{\min(m,n)} \end{pmatrix} \quad \begin{array}{l} \sigma_i: \text{ singular values} \\ \sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_{\min(m,n)} \geq 0 \end{array} \quad (\text{B.1})$$

- Rank of A : $\text{rank}(A) = \#\{\sigma_i > 0\}$
- Numerical ϵ -rank of A : $\text{rank}_\epsilon(A) = \#\{\sigma_i > \epsilon\}$
- kernel of A
 - Spanned by the column vectors v_i of V , where $\sigma_i = 0$.
 - Condition number

where σ_k is smallest non-zero singular value.

- Range of A spanned by the column vectors u_i of U , where σ_i are non-zero singular values.
- Orthogonalization of A
 - Let $A = U D V^T$ numerically orthogonal,
 - Set all $\sigma_i = 1$,
 - Compute $A' = U D' V^T$,
 - Then A' is exactly orthogonal and minimizes

$$\|A - A'\|_F \quad (\text{B.2})$$

where $\|\cdot\|_F$ denotes the Frobenius norm.

RICH MAN'S SVD

every M (rows) \times N (columns) matrix $A^{M \times N}$ can be decomposed as

$$A^{M \times N} = U^{M \times N} D^{N \times N} V^T{}^{N \times N} = \sum_{i=1}^N \sigma_i u_i v_i^T$$

where the three matrices U , D , V have the following properties

- the columns of the $M \times N$ matrix U are the *eigenvectors* \mathbf{u}_i of $\mathbf{A}\mathbf{A}^T$ (hence, $U^T U = I$); i.e. the column vector \mathbf{u}_i is obtained from

$$\lambda_i \mathbf{u}_i = (\mathbf{A}\mathbf{A}^T) \mathbf{u}_i$$

and λ_i is the i -th *eigenvalue* of $\mathbf{A}\mathbf{A}^T$

- matrix D is diagonal with elements $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_r > \sigma_{r+1} = \dots = \sigma_N = 0$, $r \leq N$; the elements σ_i of D are the *singular values*; they are the square-roots of the *eigenvalues* of $\mathbf{A}\mathbf{A}^T$, i.e. $\sigma_i = \sqrt{\lambda_i}$;
- the columns of the $N \times N$ matrix V are the *eigenvectors* \mathbf{v}_i of $\mathbf{A}^T \mathbf{A}$ (hence, $V^T V = I$);
- if $M < N$, $\sigma_i = 0$, $i = M + 1, \dots, N$; corresponding columns of U are zero;

- the *condition number* $c = \frac{\sigma_1}{\sigma_N}$ measures the “degree of singularity” of \mathbf{A} ; if $1/c$ is comparable to the arithmetic precision of the computer, \mathbf{A} is *ill-conditioned* and “singular for practical purposes”;
- columns of U corresponding to non-zero singular values span the range of \mathbf{A} ; columns of V corresponding to zero singular values span the kernel of \mathbf{A} ;
- denoting the columns of U and V by \mathbf{u}_i and \mathbf{v}_i , respectively, we have $\mathbf{A}^T \mathbf{u}_i = \sigma_i \mathbf{v}_i$ and also $\mathbf{A} \mathbf{v}_i = \sigma_i \mathbf{u}_i$;
- the *pseudo-inverse* \mathbf{A}^+ of \mathbf{A} is

$$\mathbf{A}^+ = \mathbf{V} \mathbf{D}'^{-1} \mathbf{U}^T \quad (\text{B.3})$$

with entries of \mathbf{D}'^{-1} equal to \mathbf{D}^{-1} for nonzero singular values and zero otherwise

- the Frobenius-norm of a matrix is

$$\|\mathbf{A}\|_F = \sum_{i,j} \sqrt{a_{i,j}^2} = \sum_i \sqrt{\sigma_i^2}$$

consider the linear equation $\mathbf{A}\mathbf{x} = \mathbf{a}$ with \mathbf{A} a square matrix;

1. if \mathbf{A} is *nonsingular*, we get from SVD

$$\mathbf{x} = \mathbf{A}^{-1} \mathbf{a} = \mathbf{V} \mathbf{D}^{-1} \mathbf{U}^T \mathbf{a} \quad \text{with } \mathbf{D}^{-1} = \text{diag}(1/\sigma_i)$$

2. if \mathbf{A} is *singular* and \mathbf{a} is in the range of \mathbf{A} , we get from SVD the solution of minimal norm

$$\mathbf{x} = \mathbf{V} \mathbf{D}'^{-1} \mathbf{U}^T \mathbf{a} \quad (*)$$

with $\mathbf{D}'^{-1} = \text{diag}(d'_i)$, with $d'_i = 1/\sigma_i$, if $\sigma_i \neq 0$, and $d'_i = 0$, if $\sigma_i = 0$;

3. if \mathbf{A} is *singular* and \mathbf{a} is *not* in the range of \mathbf{A} , we get from (*) the solution with minimal mean square error $\varepsilon = |\mathbf{A}\mathbf{x} - \mathbf{a}|$
4. if \mathbf{A} is *ill-conditioned*, then it is often advisable to obtain a solution from

$$\mathbf{x} = \mathbf{V} \mathbf{D}''^{-1} \mathbf{U}^T \mathbf{a} \quad (\text{B.4})$$

with $\mathbf{D}''^{-1} = \text{diag}(d''_i)$, with $d''_i = 1/\sigma_i$, if $\sigma_i \geq \Delta$, and $d''_i = 0$, if $\sigma_i < \Delta$;

now we consider a linear equation with an $M \times N$ matrix \mathbf{A} :

- 5 if \mathbf{A} is an $M \times N$ matrix with $M < N$ (underdetermined system), there is no unique solution;
 in the SVD of \mathbf{A} there will be $N - M$ zero singular values σ_i ;
 and there may be additional zero (or close to zero) singular values;
 one solution is obtained from (B.4),
 the solution space is obtained by adding the linear combination of the kernel of \mathbf{A} (see above: a basis of this kernel are the columns of \mathbf{V} corresponding to zero or zeroed σ_i);
- 6 if \mathbf{A} is an $M \times N$ matrix with $M > N$ (more equations than unknowns, i.e. overdetermined system), we want the solution which minimizes the mean square error;
 again it is obtained from (*);

conclusion: SVD is a kind of panacea for linear equations — it cannot fail (but in ‘easy’ cases there may be faster solutions);

for numerical computation of SVD see “Numerical Recipes in C”.¹

¹<http://www.nr.com>

Annotated Bibliography

- [AF02] A. Alsam and G. Finlayson. Recovering spectral sensitivities with uncertainty. In *Proceedings of the First international Conference CGIV*, pages 22–26, Poitiers, France, 2002.
- [BN98] H. Burkhard and B. Neumann, editors. *Computer Vision — ECCV '98*, number 1406 in Lecture Notes in Computer Science, Heidelberg, 1998. Springer.
- [FF95] B. V. Funt and G. D. Finlayson. Color constant color indexing. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 17(5):522–529, 1995.
- [FSC98] G.D. Finlayson, B. Schiele, and J.L. Crowley. Comprehensive colour image normalization. In Burkhard and Neumann [BN98], pages I/475–490.
- [JKS95] R. Jain, R. Kasturi, and B. G. Schunk. *Machine Vision*. McGraw-Hill, Inc., 1 edition, 1995.
- [Kai67] Thomas Kailath. The divergence and bhattacharyya distance measures in signal selection. *IEEE Transactions on Communication Technology*, 15:52–60, 1 1967.
- [KKIK03] Joni-Kristian Kamarainen, Ville Kyrki, Jarmo Illonen, and Heikki Käviäinen. Improving similarity measures of histograms using smoothing projections. *Pattern Recognition Letters*, 24:2009–2019, 2003.
- [KL51] Solomon Kullback and Richard Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22(1):79–86, 3 1951.
- [Nie83] H. Niemann. *Klassifikation von Mustern*. Springer, Heidelberg, 1983.
- [OP84] E. Oja and J. Parkkinen. On Subspace Clustering. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing*, pages 692–695, San Diego, 1984.
- [Pap02] Athanasios Papoulis. *Probability, Random Variables and Stochastic Processes*. McGraw Hill, 4 edition, 2002.
- [PBRT98] J. Puzicha, J.M. Buhmann, Yossi Rubner, and Carlo Tomasi. Empirical evaluation of dissimilarity measures for color and texture. In H. Burkhard and B. Neumann, editors,

- Computer Vision — ECCV '98*, number 1406 in 1, pages 563–577, Heidelberg, 1998. Springer Verlag.
- [PBRT99] Jan Puzicha, Joachim M. Buhmann, Yossi Rubner, and Carlo Tomasi. Empirical evaluation of dissimilarity measures for color and texture. In *ICCV 99: Proceedings of the International Conference on Computer Vision-Volume 2*, pages 1165–1173, Washington, DC, USA, 1999. IEEE Computer Society.
- [PFTV88] W.H. Press, B.P. Flannery, S. Teukolsky, and W.T. Vetterling. *Numerical Recipes - the Art of Numerical Computing, C Version*. Cambridge University Press, Cambridge, 1988.
- [PG95] T. Pomierski and H. M. Gross. Biological neural architecture for chromatic adaptation resulting in constant color sensations. In *International Conference on Neural Networks*, Berlin, 1995. Springer.
- [PH03] Dietrich Paulus and Joachim Hornegger. *Applied pattern recognition: A practical introduction to image and speech processing in C++*. Advanced Studies in Computer Science. Vieweg, Braunschweig, 4 edition, 2003.
- [Rip96] B. D. Ripley. *Pattern Recognition and Neural Networks*. Cambridge University Press, Cambridge, 1996.
- [SB91] M. J. Swain and D. H. Ballard. Color indexing. *International Journal of Computer Vision*, 7(1):11–32, November 1991.
- [Sch97] B. Schiele. *Object Recognition using Multidimensional Receptive Field Histograms (English translation)*. PhD thesis, Institut National Polytechnique de Grenoble, Grenoble Cedex, 1997.
- [TK92] Carlo Tomasi and Takeo Kanade. Shape and Motion from Image Streams: a Factorization Method, Full Report on the Orthographic Case. Technical Report CMU-CS-92-104, Carnegie Mellon University, 3 1992.
- [VMH97] V. V. Vinod, Hiroshi Murase, and Chie Hashizume. Focussed color intersection with efficient searching for object extraction. In *Pattern Recognition*, volume 30, pages 1787–1797, 1997.

B.1 Names and Symbols

The following list shows all important symbols in the sequence of their definition in the text.

N_M	Geometric distance measure 10, 11
Mask size Y 5, 13	
N_M	R
Sub-image size vertically 5, 13	Ratio histogram 11
M_M	f
Mask size X 5, 13	Color image function 11
M_M	D_r
Sub-image size horizontally 5, 13	Object mask 11
M	N_o
Image size Y 6, 7, 12, 13, 22	Number of objects 11
N	O
Image size X 6, 7, 12, 13, 22	Segmentierungsobjekte 11
l	E
Index of Bins 6, 7, 8	Spectral energy distribution of illumination 21, 22, 23
N_L	S
Number of Bins 6, 7, 8, 12, 13	Reflection coefficient 21, 22, 23
ζ	G
Color mapping function to bin 6, 7, 22	Gauß-Maske 22
d	κ
	Class Index 27, 28, 29, 30, 31, 32