*Computer Vision and Image Understanding*, Vol. 94(1-3):234–270, 2004

# Combining Spatial and Colour Information for Content Based Image Retrieval

Gunther Heidemann
Faculty of Technology, Neuroinformatics Group
Bielefeld University
Germany

October 17, 2005

**Abstract**

Colour is one of the most important features in content based image retrieval. However, colour is rarely used as a feature that codes local spatial information, except for colour texture. This paper presents an approach to represent spatial colour distributions using local principal component analysis (PCA). The representation is based on image windows which are selected by two complementary data driven attentive mechanisms: A symmetry based saliency map and an edge and corner detector. The eigenvectors obtained from local PCA of the selected windows form colour patterns that capture both low and high spatial frequencies, so they are well suited for shape as well as texture representation. Projections of the windows selected from the image database to the local PCs serve as a compact representation for the search database. Queries are formulated by specifying windows within query images. System feedback makes both the search process and the results comprehensible for the user.

Keywords: Image retrieval, image indexing, salient points, object recognition, unsupervised learning, vector quantisation, neural networks, local PCA, colour features, system feedback.

## 1 Introduction

Colour plays a central role in most systems for content based image retrieval (CBIR), e.g. in QBIC [1], VisualSEEK [2], Photobook [3], Virage [4], Blobworld [5], PicToSeek [6] or SIMPLIcity [7], to name but a few. Using colour extensively, CBIR is in this respect more advanced than other fields of computer vision, however, the potential of colour is not yet fully tapped.

In this contribution, a method for image retrieval is proposed that represents an image database using local principal component analysis (local PCA) that captures shape and colour jointly. An image is not represented as a whole, instead, salient image patches or "windows" are selected by data driven attentional mechanisms. The collection of all windows of all images forms a huge dataset in $\mathbb{R}^d$, where each window is

considered as a vector of dimension $d = 3 \cdot w \cdot h$ where $w, h$ are width and height of the window and three colour channels are given. This highly non-linear data distribution in $\mathbb{R}^d$ is approximated by local PCA, which can be considered a nonlinear extension of normal PCA [8].

PCA is a standard method in computer vision to project an image window onto a lower dimensional feature space. As PCA captures the directions of greatest variance in the data space, the resulting feature vectors are still highly specific, but reduce redundancy for the benefit of lower computational effort and better stability. A well known example is face classification using so called "eigenfaces", which are filters obtained by PCA from face sample images [9]. Normal PCA is the optimal linear method to approximate a data set in the least squares sense. However, linear methods are inappropriate for non-linear or even clustered distributions, as illustrated in Fig. 1. The required non-linear generalisation of PCA is *local PCA* [10, 11], which has been applied to dimension reduction [12], image interpolation [13], transform coding [14] and pattern recognition [15, 16, 17] and in particular face classification [18]. A probabilistic version of local PCA was proposed in [19].

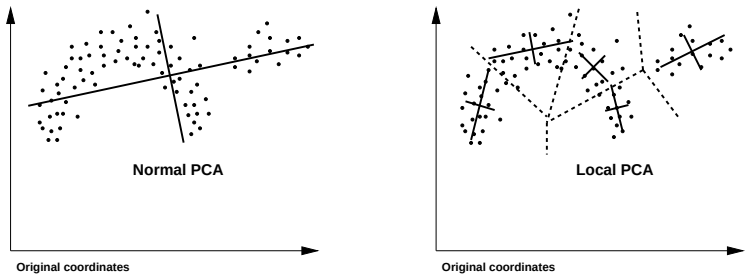


Figure 1: Left: Normal PCA cannot capture the structure of a non-linear data distribution. Right: Local PCA first approximates the data by a set of reference vectors, e.g. using clustering methods. Then within each Voronoi tessellation cell normal PCA is carried out. Note a 2D-sketch can only roughly illustrate the principle: Here, each of the local coordinate systems spans the entire 2D-plane. In high dimensional spaces, however, the data distribution has much more "directions to turn to", so the local PCs of different reference vectors mostly capture different directions.

In contrast to most other works on PCA and local PCA, in this contribution the full RGB colour information will be used for local PCA — not just grey level information. The resulting eigenvectors, which serve as filters or "feature detectors", are consequently not specialised only to (grey level–) shape, but to a "coloured shape". Fig. 8 gives an impression.

In the following, first an overview of the proposed system will be given, then the relations to existing methods will be discussed. An overview of the organisation of the paper is given at the end of the section.

## 1.1 Overview of the image retrieval system ColoSeek

This paper introduces the retrieval system ColoSeek (**Col**our**O**bjectSeek). Though ColoSeek can work stand-alone in principle, it has to be integrated into a larger image search machine which involves more types of features. ColoSeek is presented here in a "pure" state to demonstrate what the approach can achieve. In this section a qualitative overview of the retrieval techniques used in ColoSeek will be given, before going into details in the following sections.

In ColoSeek, both internal image representation and queries for images by the user are based on *windows*. A window $\vec{W}$ is defined as a square patch of a colour image, which can be considered as a vector in a $d$-dimensional space: $\vec{W} \in \mathbb{R}^d$. Since there are three colour channels, the window-vector has the dimension $d = 3 \cdot w \cdot h$, where $w, h$ are the window dimensions.

A user can formulate a query in terms of windows which contain examples of what is searched for. It is provided that the user is able to supply an initial set of images from which suitable windows can be cut out. The user may make up a query from an arbitrary number of sample windows. There are four pre-defined window sizes which have to be used for the query. The different window sizes are internally sub-sampled all to the same size, by this means all windows have the same dimension $d$ and can be represented in the same vector space $\mathbb{R}^d$. The user interaction of ColoSeek is described in sections 4 and 5.

The system needs a compact representation of the database that facilitates the search for images. Given a new database, building the representation starts with the extraction of windows. Since it would be neither possible nor useful to extract and represent all possible windows of all images, a different solution has to be found: The system has to select windows which are likely to comprise relevant objects or entities. Since the step of building the representation has to be performed without human assistance, only data driven methods can be used to find "interesting" windows. Such methods are saliency detection from natural and artificial symmetry and from the auto-correlation matrix of the signal. The data driven window selection is described in section 2. The process of "exploring" a new database is depicted in Fig. 2.

Once the system has collected sample windows, an internal representation can be built up, which consists of two modules: (1) A local PCA based data representation obtained by unsupervised learning methods, which serves for the extraction of features, and (2) a search database. The data representation by local PCA is an approximation of the entire set of windows in $\mathbb{R}^d$ as described in section 3. When the local PCA–approximation is ready, the search database can be built up. The search database contains one entry for each image together with the feature vectors extracted by local PCA from the windows of the image (section 3.3).

When a query is made, the chosen windows are first projected onto the local PCA representation. From the projections a "prototypic search image" is reconstructed which serves as a feedback to the user. It gives a rough impression of what the machine is currently searching for. In other words, the user sees how the machine "understands" the chosen query windows. This back-projection will be referred to as *search state visualisation*, SSV. Additionally, the user can give different *weights* to the chosen windows and monitor the effect on the SSV. So while the SSV is a *system feedback*, assigning different weights to query windows plays the role of the users *relevance feedback*. Fig. 3 illustrates the retrieval process.

When the user has made up a query from windows using the SSV, the best match of the windows (in terms of the local PCA – projection) has to be found in the search database, additionally the second best match and so on. The result of the query is a set of images in the order of similarity to the query in which the windows are indicated. So the result is actually a set of windows but combined with the images from which they were cut out. Therefore, it is possible to search for single objects like a cup in entirely different sceneries. However, the set of possible query results is restricted to those windows which were originally found by the window selection.

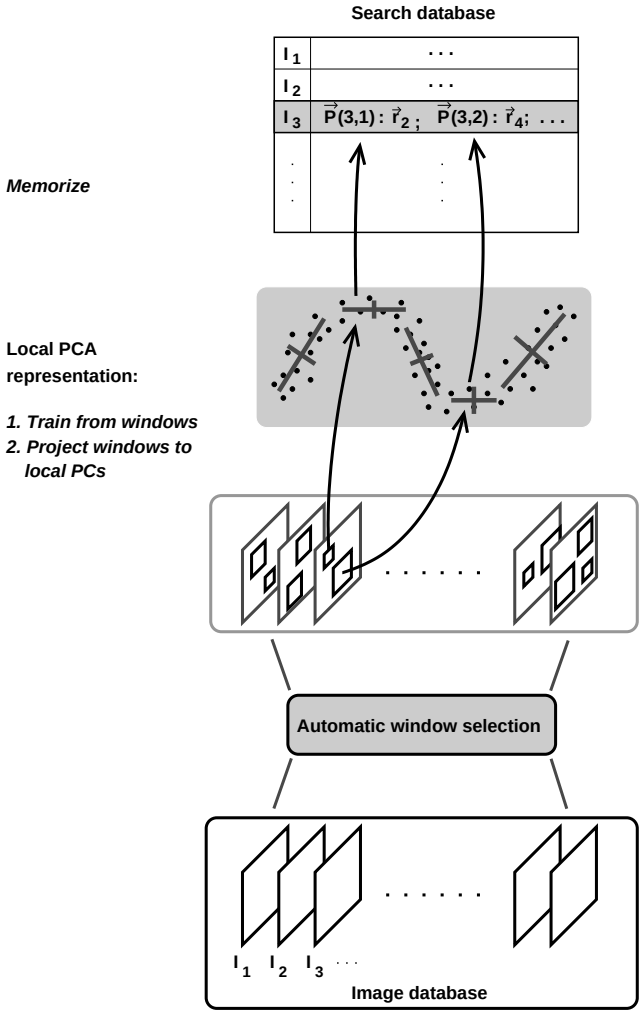


Figure 2: In the acquisition phase, windows are selected data driven from all images of the database. The local PCA-representation is first derived as an approximation of the set of windows, subsequently all windows are projected to the local PCs. The projections are memorised as features in a database.

ColoSeek facilitates also complex queries for several sets of windows at a time, which can be combined using the Boolean operators AND, OR and NOT or the operator ATLEAST($n$) as described in section 4.4. So it is possible, e.g., to define the scene of a beach as a window on a large scale and combine it using AND with a query for a window containing palm trees.

## 1.2 Aims of the new approach and relations to previous research

ColoSeek is related to the *view based* approach in object recognition. This technique is a powerful tool when geometrical modelling of an object domain is intractable. The key idea is memorizing features extracted from object views using filters generated from samples, instead of memorizing representations derived from human expertise. In view based recognition systems, eigenspace representations have been successful, for example in the well-known object classification approach by Murase and Nayar [20], the face classification

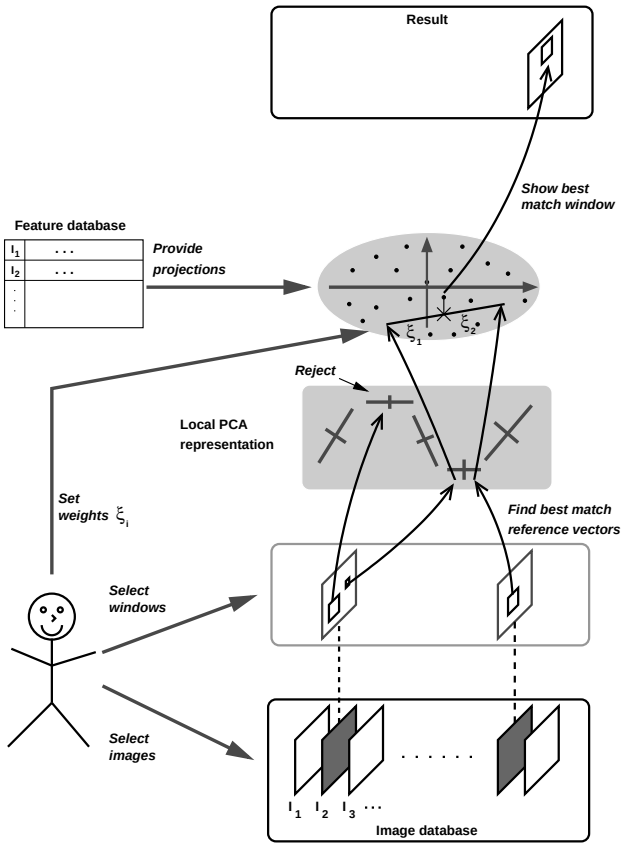


Figure 3: For a query, the user selects windows from images of the database, which are projected to the local PCA – representation. The actual query is composed of those windows which are best match to the reference vector with the most hits from the query. Other windows are rejected, if there are too many of them, the user is asked to make a more coherent query. Using the weights $\xi_i$ specified by the user, a linear combination of the query windows is calculated in the local PC – subspace. The search database provides the memorised projections of sample windows, so the best match can be found. The result is a subset from the image database, the relevant windows are indicated.

techniques by Turk and Pentland [9] and later Moghaddam and Pentland [18], or in the visual servoing application of Nayar et al. [21].

The local PCA – approach used in ColoSeek has been applied in previous work to classification of facial features [22], object- and texture recognition [23], hand posture recognition [24] and visual judging of grasp stability in robotics [25]. This variety shows the major benefit of the approach: It is inherently scale- and domain independent. The resulting filters can specialise to colour shape as well as to shapes of mere grey value contrast, to large scale structures as well as to texture.

The combination of a local PCA – representation with data driven attentional mechanisms for salient point selection is aimed to make progress compared to previous research in three aspects:

**Combining colour and shape as a single feature channel:** In most approaches, colour is still treated as a feature *separate* from shape, i.e., no information on the spatial colour distribution is exploited. Examples are the numerous methods based on colour histograms [26, 27], colour sets [28] or colour moments

[29]. Though systems using colour histograms often evaluate also shape, nevertheless colour and shape are regarded as different feature channels. Approaches which deal with the spatial distribution of colour are usually limited to small scales, i.e. colour texture [30]. Another direction of research is encoding very coarse information about spatial colour distribution in a colour index [31].

In contrast, detecting the shape of objects is usually based on grey values. Approaches for grey value based shape detection are numerous, examples are edge based methods [32], elastic deformation matching [33] or local grey value invariants [34, 35]. An overview on the use of shape in CBIR is given by Mehtre et al. [36], for a more condensed summary see Smeulders et al. [37].

One of the rare approaches to exploit the local spatial colour structure was proposed by Gevers and Smeulders [38], who evaluate pairs of hue values to both sides of colour edges. The method favours, however, objects with many colour edges, in contrast, objects with only few different colours or continuously changing colours lead to less specific representations.

**Indexation based on local descriptors:** The majority of works on CBIR is based on global feature distributions: Either features are used which are global in themselves or distributions of local features spread out over the whole image. This applies to both the indexed images in the database and to query images given by the user. But since natural images are highly heterogeneous, evaluating *whole* images is inappropriate when only objects are searched for. If the user gives a query image of a car under a tree, then what is wanted? The car or the tree? So, if the user wants the car, she or he should indicate this in the query image and features should be extracted in this area only. Similarly, if the query features can be found in the database, the image should appear in the result list — independent of the surroundings of the car. In contrast, when an overall image structure is desired — e.g. a landscape under blue sky — it should be possible to overlook details deliberately. This problem can be solved if the user can chose among differently sized windows and position them arbitrarily on query images.

Tian et al. [39] realise the idea of using local features to a certain extent. They use Haar-wavelets (see e.g. [40]) at different resolutions to obtain salient points. Features are extracted only in the neighbourhood of the points, so the image is not scanned completely. Schmid and Mohr [34] propose a related approach based on grey value invariants extracted at interest points. But still a complete query image is searched for in a database of images, while ColoSeek searches for windows in a database of windows (associated to images). Concerning this strategy, it has to be noted that ColoSeek is not primarily designed as a stand-alone tool, but rather as a search facility a retrieval system should offer.

**System feedback:** A frequently heard complaint of users of image retrieval systems is the following: "Why does he do that now?!" The user gets usually no insight of (a) what the system is searching for, (b) why a certain result was chosen, (c) how the system was caused to search for this result and finally (d) how to express the aim better in terms of query images.

*System feedback* is aimed to overcome such shortcomings. Smeulders et al. [37] give a good overview of existing techniques. Solutions to the problems/questions mentioned above can be sorted into two groups: (1) System feedback accompanying the result images, indicating why they were chosen, and (2) system feedback during the search, e.g., display of the search space.

Using a window based approach, system feedback can be easily given by displaying the best match windows in the result images (problem (b)). Also, problem (d) becomes less critical because a query can be narrowed down by using "tight" windows around the searched objects. Other approaches to give feedback in the result images range from indicating regions that contribute most to the result ranking [41] to explicit labelling of recognised image components [42].

In contrast, visualisation of the search space is more difficult. When both the images of the database and the query are expressed as feature vectors, the search space can be visualised as the "neighbourhood" of the query vector in feature space. Here, two directions can be distinguished: Visualising proximity in feature space using the *real images* as, e.g., in [43, 44], or generating artificial images of the region around the query vector in feature space. The latter is done in ColoSeek, as described in section 4.2. Though the method of back-projection from feature space to pixel space is simple and yields only a rough impression, it is nevertheless helpful, as outlined in section 5.

### 1.3 Organisation of the paper

In the following two sections, 2 and 3, the phase of acquiring a search database from a set of images will be described: Reduction of the image database to a set of windows, approximating the manifold of windows by local PCA and memorizing the windows as feature vectors. Section 4 describes the application of this system for image queries. In section 5, results are presented: Query examples, tests by users and an evaluation of system performance against imaging conditions. Finally, conclusions and future research topics arising from this work are discussed in section 6.

## 2 Automatic window selection

Selecting appropriate image patches or "windows" from which features can be extracted plays a key role in image retrieval. Windows may comprise small but important details like a face in the crowd, but also larger entities or even the entire image. In ColoSeek, two different strategies for window selection complement each other:

1. **Global windows:** From each image $I_i$ one *global window* $\vec{W}_G(i)$ is cut out around the center such that it covers the major part of the image. This window is strongly sub-sampled because it serves to give an overview of the global image structure. Selection of the global window is described in section 2.1.

2. **Local windows:** To capture details at various scales, *interest points* are generated for all images of the database. Depending on the scale on which the algorithm for interest point detection works, several *local windows* $\vec{W}_L(i,j)$ are cut out for each image $I_i$, where $j = 1 \ldots N_L(i)$ is the index of the windows. The methods for interest point detection are described in sections 2.2 and 2.3.

By this means, there are one fixed global window and several local windows at different locations for each stored image. Table 1 gives an overview of all used types of windows.

Local windows should characterise prominent local image features to give the user the possibility to search, e.g., for objects within an image. Therefore, local windows will be centered at salient points or

Table 1: Overview of the window sizes. The global window $W_G$ is sized to 90% of the smaller one of the image dimensions, hence the subsampling factor is adaptive.

| Window type | Size in image | Sub-samp. | Wid. size | Location determined by |
|---|---|---|---|---|
| Global window $\vec{W}_G$ | $0.9 \cdot \min(w, h)$ | (adaptive) | $32 \times 32$ | Center |
| Local window $\vec{W}_L^1$ | $128 \times 128$ | 4 | $32 \times 32$ | SYM(R=50), HARRIS |
| Local window $\vec{W}_L^2$ | $64 \times 64$ | 2 | $32 \times 32$ | SYM(R=25), HARRIS |
| Local window $\vec{W}_L^3$ | $32 \times 32$ | — | $32 \times 32$ | SYM(R=12), HARRIS |

"interest points" generated from domain independent saliency features. Such features, that can be found in almost all natural and artificial images, are *symmetry* and *edges and corners*. Algorithms which can detect these basic features will be described in sections 2.2 and 2.3, respectively. Feature extraction from both global and local windows will be described in section 3.

## 2.1 Selection of the global window $\vec{W}_G$

To give a uniform characterisation of the central part, a global window $\vec{W}_G$ of square shape is cut out from each image. $\vec{W}_G$ is slightly smaller than the largest square fitting in the image to avoid border effects, the side length of $\vec{W}_G$ is equal to 90% of the smaller one of the image dimension. The center of $\vec{W}_G$ is at the center of $I_i$ (Fig. 4). Since $\vec{W}_G$ should give only a rough approximation of the image, it is smoothed and sub-sampled to resolution $32 \times 32$.[1]

## 2.2 Selection of local windows using symmetry

In the following, first the symmetry based algorithm SYM for interest point detection will be described (section 2.2.1), which relies on an approach of Reisfeld et al. [45]. Then the way local windows are selected from the database using SYM will be outlined in section 2.2.2.

The use of symmetry as a general saliency feature is motivated by psychophysical findings [46, 47, 48], which indicate that human attention is drawn towards symmetric image regions. In particular, it could be shown by Privitera and Stark [49] that a high correlation exists between interest points predicted by the algorithm of [45] and human eye fixations. Furthermore, previous studies showed that SYM yields interest points which are robust against object rotations, camera noise and changes of lighting [50].

### 2.2.1 Saliency maps from local symmetries

Reisfeld et al. [45] have proposed an algorithm that creates a saliency map $M_{Sym}$ from local symmetries. $M_{Sym}$ is a continuous valued, local symmetry judgement based on grey value edges. Since the complete

[1] For the database used in the current work (section 5.1), for all images the smaller side length ranges between 392 and 480 pixels. Therefore, a common size of $384 \times 384$ was chosen for $\vec{W}_G$ and sub-sampled by a factor 12.


Original | Global window $W_G$

HARRIS saliency map | Best maxima of HARRIS saliency map

Figure 4: Selection of windows for subsequent feature extraction: One global window $W_G$ is cut out around the center of each image. Local windows: Using the HARRIS algorithm, a saliency map $M_{Harris}$ is generated. The highest maxima of this map are used as the centers for the local windows $W_L^1, W_L^2$ and $W_L^3$.

algorithm is beyond the scope of this contribution, it can be outlined here only in short, for details see the original work [45].

To find local symmetries, first the local grey value gradients are calculated. Then for each pixel $p$ a set $\Gamma(p)$ of index pairs $(i, j)$ of surrounding pixel pairs $(p_i, p_j)$ is defined:

$$\Gamma(p) = \left\{ (i,j) \;\middle|\; \frac{p_i + p_j}{2} = p \right\}. \tag{1}$$

The symmetry value $M_{Sym}(p)$ of pixel $p$ is a sum over all index pairs of $\Gamma(p)$:

$$M_{Sym}(p) = \sum_{(i,j)\in\Gamma(p)} PWF(i,j) \cdot GWF(i,j) \cdot DWF_\sigma(i,j). \tag{2}$$

The most important factor in this sum is the so called *phase weight function* $PWF(i, j)$. It evaluates the gradient directions at pixel pairs $(p_i, p_j)$ around $p$ for the probability that $(p_i, p_j)$ are part of a symmetric object with symmetry center at $p$. The second factor, the *gradient weight function* $GWF$, is a logarithmic

SYM(R=50) saliency map | Best maxima of SYM(R=50) with $W_L^1$

SYM(R=25) saliency map | Best maxima of SYM(R=25) with $W_L^2$

SYM(R=12) saliency map | Best maxima of SYM(R=12) with $W_L^3$

Figure 5: The symmetry algorithm SYM is applied to each image with three different symmetry radii $R$. The left column shows the resulting saliency maps. Since $R$ defines the "scale" on which symmetry is detected, the local windows are assigned accordingly: Windows $W_L^1$ are centered at the highest maxima of SYM(R=50), $W_L^2$ for SYM(R=25) and $W_L^3$ for SYM(R=12) (right column).

weighting function which gives larger weight to symmetry contributions from pixel pairs with high gradient magnitudes, but attenuates the influence of very strong edges.

The last factor of Eq. (2) is the *distance weight function* $DWF_\sigma$, which makes $M_{Sym}$ a local measure:

$$DWF_\sigma(i,j) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{\|p_i - p_j\|^2}{2\sigma^2}\right). \tag{3}$$

The width $\sigma$ of the Gaussian defines the scale on which symmetries are detected. In practice, however, the factor $DWF_\sigma$ can be omitted if the set $\Gamma(p)$ is replaced by a restricted set $\Gamma^*(p,R)$, as proposed by Nattkemper [51]:

$$\Gamma^*(p,R) = \left\{(i,j) \;\middle|\; \frac{p_i + p_j}{2} = p \;\wedge\; \|p_i - p_j\| \le 2R\right\}. \tag{4}$$

Using $\Gamma^*(p,R)$ instead of $\Gamma(p)$, only a circular neighbourhood of $p$ contributes to the symmetry value $M_{Sym}(p)$. The *symmetry radius* $R > 0$ takes the role of the parameter $\sigma$ to restrict contributions to $M_{Sym}(p)$ to the surroundings of $p$. Thus, the simplified version of $M_{Sym}(p)$, which will be used throughout this paper, is

$$M_{SYM}(p,R) = \sum_{(i,j)\in\Gamma^*(p,R)} PWF(i,j)\cdot GWF(i,j). \tag{5}$$

$M_{SYM}$ shows no substantial difference from the original measure (Eq. (2)) but is computationally much more efficient.

### 2.2.2 Local windows from symmetry maps

The symmetry radius $R$ plays a key role for $M_{SYM}(p,R)$, because it restricts the scale on which symmetries can be detected. ColoSeek uses three different radii to evaluate a large, medium and small scale. Examples for the resulting symmetry maps $M_{SYM}$ are shown in Fig. 5, left column.

Interest points are the maxima of $M_{SYM}$. However, if only a single image is considered, it is difficult to judge whether the best (i.e. the highest) maxima really correspond to "strong" symmetries — even an image of pure noise would have its maxima somewhere. Therefore, first the symmetry values $M_{SYM}$ of the ten best maxima of each image of the entire database are collected in a histogram. The histogram helps to choose a symmetry threshold $\theta_{SYM}$ that defines which maxima will be "accepted" as the locations of interest points, however, the actual value of $\theta_{SYM}$ still has to be chosen by the designer. For the setup evaluated in this contribution, $\theta_{SYM}$ was chosen such that 60% of all images obtain at least ten interest points from symmetry, thus making a compromise between over-representation of images with multiple strong symmetries and under-representation of others.

For each image an interest point is set at each maximum of $M_{SYM}$ that exceeds $\theta_{SYM}$. However, to reduce the influence of $\theta_{SYM}$ and thus to avoid crowds of interest points on regular symmetric patterns like a wallpaper, the maximum number of interest points which can be generated by SYM for a single image is restricted to 30. Similarly, a minimum number of three interest points is required to avoid that an image remains without feature vectors. So the three best maxima are always used regardless of their absolute height.

The size of the extracted local windows is coupled to the scale on which their interest points were detected. In an image database of realistic complexity as used in section 5, symmetries of all scales are

represented, so an adaptation to the database is impossible. Therefore, a possibly large range of scales should be covered. In ColoSeek, the size ratios $\vec{W}_L^1$ / $\vec{W}_L^2$ / $\vec{W}_L^3$ are chosen $4\,/\,2\,/\,1$. As the size of the subsampled global window $\vec{W}_G$ is $32 \times 32$, all local windows are bound to be subsampled to the same size for feature extraction. Therefore, the smallest possible size is $32 \times 32$ for $\vec{W}_L^3$, which leads to dimensions $128 \times 128$ and $64 \times 64$ for $\vec{W}_L^1$ and $\vec{W}_L^2$, respectively. It is provided that $128 \times 128$ fits into the images, which is the case for common formats. Table 1 gives an overview of the windows. For feature extraction, only windows completely inside the image are used, so interest points too close to the borders are discarded.

Symmetry radii were chosen such that the evaluated region is slightly smaller than the window: $R = 50, 25$ and 12, for $\vec{W}_L^1$, $\vec{W}_L^2$ and $\vec{W}_L^3$, respectively. The idea is that features should be extracted not only in the symmetric area itself, but also in its proximity to achieve a greater variability. Fig. 5, right column, shows examples of local windows. For $R = 12$, small details like the door of the lighthouse are comprised in windows $\vec{W}_L^3$. The largest scale $R = 50$ yields windows $\vec{W}_L^1$ containing an entire building.

## 2.3 Selection of local windows using edges and corners

Harris and Stephens [52] have proposed a method for corner and edge detection which will be used to supplement the symmetry based interest point detection. While SYM tends to generate interest points in the center of objects, corner and edge detection yields more interest points on the border of objects or on prominent region borders in an image, like, e.g., the horizon.

There is a large variety of edge and corner detectors. Many of them rely on discrete approximations of the auto-correlation function of the signal [52, 53, 54], as proposed originally by Moravec [55]. Others search for points of high curvature on extracted contours (e.g. [56]) or intersection points [57, 58].

The approach of Harris and Stephens [52] — which will be referred to as "HARRIS" — was chosen because Schmid et al. [59] could show that it is more stable than other, related methods. In particular, it could be shown that interest points obtained by HARRIS are robust against 2D image rotations, scaling, lighting variation, viewpoint change and camera noise. Most important from the point of view of image retrieval is that windows centered at interest points chosen by HARRIS have a higher information content than other, randomly selected windows. Schmid et al. measured the information content in terms of the entropy of local descriptors [59].

HARRIS is based on first derivatives of the signal $I_x, I_y$

$$I_x(p) = \frac{\partial I(p)}{\partial x}, \quad I_y(p) = \frac{\partial I(p)}{\partial y}, \tag{6}$$

which are approximated using $5 \times 5$-Sobel operators in ColoSeek. From the derivatives a matrix $A$ is calculated

$$A(p) \;=\; \begin{pmatrix} \langle I_x^2 \rangle_{W^*(p)} & \langle I_x I_y \rangle_{W^*(p)} \\ \langle I_x I_y \rangle_{W^*(p)} & \langle I_y^2 \rangle_{W^*(p)} \end{pmatrix}, \tag{7}$$

where $\langle\,.\,\rangle_{W^*(p)}$ denotes a weighted averaging over the components of $A$ within a window $W^*(p)$ centered at $p$. The weighting function is a Gaussian of width $\sigma = 2$, as proposed in [59]. $A$ is an approximation of the auto-correlation function of the signal. Interest points are detected at locations where both eigenvalues of $A$ are large. To reduce the computational effort, a saliency map $M_{Harris}$ is calculated as

$$M_{Harris}(p) \;=\; \det(A) - \alpha \cdot (\mathrm{Trace}(A))^2\,, \tag{8}$$

where a value of 0.06 is used for the constant $\alpha$ as suggested in [59]. An example for $M_{Harris}$ is shown in Fig. 4.

The procedure to derive interest points from the maxima of $M_{Harris}(p)$ is the same as outlined in section 2.2.2 for SYM, so there are in total at least six and at most 60 interest points per image. Since HARRIS is aimed to detect corners and edges, it is applied only with one set of parameters. So in contrast to SYM, interest points from HARRIS refer only to one scale. As a consequence, for HARRIS all three local windows $\vec{W}_L^1, \vec{W}_L^2$ and $\vec{W}_L^3$ are centered at the same points as indicated in Fig. 4. Again, only windows completely inside the image are used.

# 3 Local PCA of colour windows

Let the database $D$ be given as a set of $N_I$ images: $D = \{I_1 \dots I_{N_I}\}$. For each image $I_i \in D$ one global window $\vec{W}_G(i)$ and $N_L(i)$ local windows $\vec{W}_L(i,j), j = 1 \dots N_L(i)$, have been extracted. Since all windows have the same dimensions ($32 \times 32$), they can be considered as a single data distribution in a $d = 3 \cdot 32 \cdot 32 = 3072$ – dimensional data space. By means of local PCA, compression is achieved in a way that redundancy is reduced and similar images become "neighbours" in the approximation, so that queries for certain types of images can be answered.

Choice of the colour space is nontrivial for the local PCA – representation. It is well known that colour spaces providing invariance are better suited for techniques like histograms than the RGB-space (e.g. [60]). Despite these advantages, however, it is not yet clear whether the same applies to local PCA. The problem is that local PCA is based on distance calculations, therefore, calculation of PCs e.g. in the HSI colour space is not straight forward due to its special topology. For this contribution, experiments have been made with the perceptually uniform CIE–$L^*a^*b^*$ space and were compared to RGB-space. The CIE–$L^*a^*b^*$ representation appears particularly well-suited because its metric is constructed such that distances correspond better to colour differences perceived by humans. Moreover, the distance calculation keeps the same euclidian form as in RGB-space. The results showed that the local PCs resulting from calculation in CIE–$L^*a^*b^*$ lead to a better visualisation in the SSV (more colour contrast) than in RGB space. However, retrieval results are better for RGB than CIE–$L^*a^*b^*$. This unexpected effect is not yet understood, but since retrieval performance is more important than the visualisation, the RGB-representation will be used in the following.

Local PCA is performed in two steps: First the data set is divided into clusters (section 3.1), than normal PCA is performed for each cluster separately (section 3.2). Once the data has been approximated, the projections of all image windows are memorised in the search database (section 3.3). An overview of the entire acquisition phase is given in Fig. 2. The method chosen here is relatively simple, since it decouples clustering and PCA. There are two main directions in which other approaches improve this procedure: One way is to keep the principle of hard clustering and conventional PCA, but to carry out both steps alternating in an iterative procedure as in [12], which allows to achieve better representations in the sense of least square reconstruction error. A different approach is the probabilistic method presented in [19].

Nevertheless, the decoupled approach is used in the present context for basically three reasons: (1) Loosely speaking, searching the "product space" of finding suitable cluster centers and PCs simultaneously

can be avoided by the decoupled approach though perhaps not the optimal solution can be found. (2) The probabilistic approach of [19] is particularly successful for highly disjoint data – it is not clear whether the image data processed here have this feature. (3) The major algorithmic requirements are efficiency for large data sets with high dimensionality and independence of sophisticated parameter settings. Due to the computational effort and the complexity of the domain, extensive search for suitable parameters is impossible, instead, algorithms have to work as "black boxes". With the AEV-algorithm for clustering and Sanger's method for PCA, as presented in the next two sections, two reliable tools with the desired properties are available, which have proven successful in several vision applications [22, 23, 25, 24].

### 3.1 Data clustering

Let the set of all extracted windows (global and local) of all images be denoted by $\mathcal{W} = \{\vec{W}_1 \dots \vec{W}_{N_W}\}, \vec{W}_i \in \mathbb{R}^d$. To find clusters of the data in $\mathbb{R}^d$, vector quantisation (VQ) will be performed. VQ is a standard method for data compression and data mining [61, 62]. VQ approximates a data distribution by a set of $N_{VQ}$ *reference vectors* (or *codewords*) $\vec{r}_1 \dots \vec{r}_{N_{VQ}} \in \mathbb{R}^d$. A data vector $\vec{W}_i \in \mathcal{W}$ can be approximated by the *best match* reference vector $\vec{r}_{\mathcal{BM}(\vec{W}_i)}$. The mapping $\mathcal{BM} : \mathbb{R}^d \to \mathbb{N}$ assigns to every vector $\vec{x} \in \mathbb{R}^d$ the number of the "closest" reference vector by minimising the distance

$$\mathcal{BM}(\vec{x}) \;=\; \arg \min_{k=1\dots N_{VQ}} \|\vec{x} - \vec{r}_k\| \qquad \text{with} \quad \vec{x} \in \mathbb{R}^d. \tag{9}$$

$\|.\|$ denotes a distance measure (here the euclidian).

To choose an appropriate VQ-algorithm for the current task, it has to be considered that most VQ-algorithms are aimed to minimise the mean square error $E$ when the data set $\mathcal{W}$ is reconstructed:

$$E \;=\; \frac{1}{N_W} \sum_{\vec{W}_i \in \mathcal{W}} \|\vec{W}_i - \vec{r}_{\mathcal{BM}(\vec{W}_i)}\|^2. \tag{10}$$

Minimisation of $E$ — by whatever means — usually leads to a good approximation of $\mathcal{W}$. However, the problem of *codeword under-utilisation* remains, especially if the dimensionality is high [63]: Since the huge volume of a high dimensional space cannot be "filled" by the provided data, some reference vectors do not "find" clusters and remain outliers. The reason is that $E$ has numerous local minima, some of which correspond to situations where the data distribution is well approximated by the majority of the reference vectors, but some remain unused. Considering $E$ alone, outliers may remain undetected, because they do not necessarily lead to a substantial increase of $E$.

For the purpose of reconstruction (section 4) it is essential that *all* reference vectors of the local PCA are really "inside" the image data distribution and not in "empty" regions of the image space. Fig. 6 illustrates the problem: While reference vectors within the clusters correspond to real images, the reference vector in the middle is just a linear combination of the two others — with no meaning on the semantic level. Reference vectors even farther away from the data would correspond to mere noise. Consequently, under-utilised reference vectors must by all means be avoided.

Several solutions have been proposed to this problem like the Neural Gas [64] or the Representation-Burden Conservation Network [65]. But since they rely on interactions between the reference vectors during the adaptation phase, certain parameters have to be chosen appropriately, which is extremely difficult for

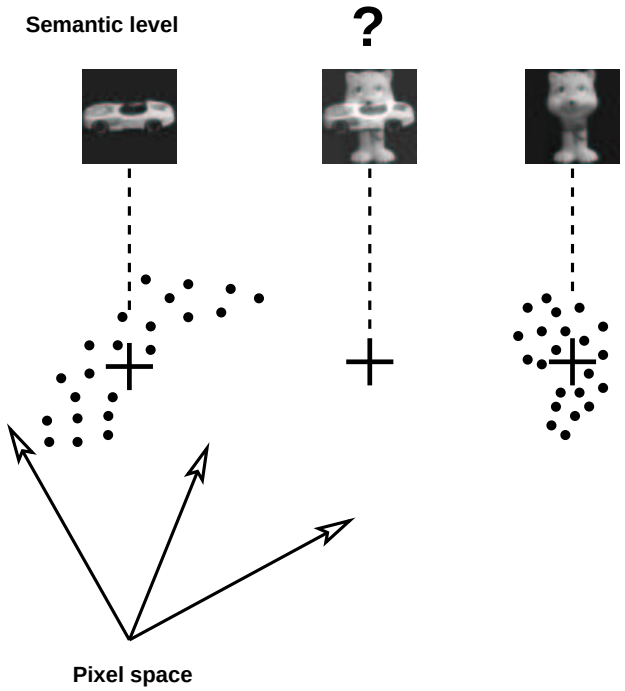


Figure 6: Reference vectors outside the actual data distribution (like the one in the middle) do not correspond to real images. Hence, they must be avoided for the purpose of image reconstruction. As a consequence, the AEV-algorithm is used for vector quantisation, because it removes under-utilised reference vectors by counting codeword access frequencies.

non-toy problems. Only algorithms which explicitly count the codeword access frequencies to avoid under-utilisation like "Frequency Sensitive Competitive Learning" [66, 67] or "Activity Equalisation VQ" (AEV) [68] solve the problem without extensive parameter tuning. For ColoSeek, AEV is used because it was successfully applied to object recognition tasks earlier [23, 25].

AEV is a competitive learning VQ algorithm based on the *winner-takes-all* rule. During adaptation to a data set, the codeword access frequencies are counted, i.e. the number of updates of each reference vector. Reference vectors with particularly low access frequencies are likely to be in regions with few data points. Such reference vectors are re-positioned to the neighbourhoods of others with high access frequencies so that the data distribution can be "explored from the inside". AEV proved to be more efficient than well known VQ algorithms like K-means [69], Maximum Entropy Clustering [70], Representation-Burden Conservation [65], Frequency Sensitive Competitive Learning (FSCL) [66, 67] and the Neural Gas [64]. For a detailed description of AEV see [68].

## 3.2 PCA using neural networks

Once the reference vectors are positioned by VQ, the local principal components (local PCs) have to be calculated from the data vectors within each Voronoi-tessellation cell. Calculation of the PCs from the covariance matrix of the data is impossible because of the dimensionality (d=3072) and the number of data points ($\approx 2 \cdot 10^5$). Therefore, the exact calculation of the PCs is replaced by an iterative neural method.

To each reference vector $\vec{r}_i$ a single layer feed forward network is associated for the successive calculation of the PCs after the learning rule proposed by Sanger ([71], for a detailed description see also [72]).

The associated PCA-net of each reference vector $\vec{r}_m$ is trained stepwise using samples selected randomly from its Voronoi-tessellation cell. The PCA-net has one node for each PC with an input weight vector $\vec{w}_i^{(m)} \in \mathbb{R}^d$ with $i = 1 \ldots N_{PC}$ for $N_{PC}$ different PCs. The index $(m)$ of the weight vector refers to its associated reference vector $\vec{r}_m$.

For a training sample $\vec{W}_l$, the activation $V_i$ of the nodes is calculated using the linear function

$$V_i^{(m)} = \sum_{j=1}^{d} (\vec{w}_i^{(m)})_j \left( (\vec{W}_l)_j - (\vec{M}^{(m)})_j \right), \quad i = 1 \ldots N_{PC}, \tag{11}$$

where $\vec{M}^{(k)} \in \mathbb{R}^d$ denotes the mean value of Voronoi tessellation cell number $k$. The weight vectors are adapted after Sanger's rule

$$\Delta(\vec{w}_i^{(m)})_j = \epsilon V_i^{(m)} \left[ \left( (\vec{W}_l)_j - (\vec{M}^{(m)})_j - \sum_{k=1}^{i-1} V_k^{(m)} (\vec{w}_k^{(m)})_j \right) - V_i^{(m)} (\vec{w}_i^{(m)})_j \right] \tag{12}$$

with $i = 1 \ldots N_{PC}$, $j = 1 \ldots d$. The adaptation step-size $\epsilon$ decreases exponentially from 0.5 to 0.001 during training. After the training, the weight vectors represent the PCs in the order of the corresponding eigenvalues, beginning with the largest. To improve ortho-normality of the obtained weight vectors, Gram-Schmidt ortho-normalisation [73] is carried out after the training. An estimation of the eigenvalues can be obtained by projecting the sample windows onto the weight vectors. The eigenvalues can be used to find a suitable value for $N_{PC}$ which is large enough to capture most of the local data variance, but still small compared to the original dimensionality $d$.

For calculation of PCs often the sample windows are multiplied (not convolved) pointwise with a Gaussian to avoid border effects [71]. In a first version of ColoSeek, all local PCs were computed from samples multiplied with a Gaussian of $\sigma = 6$ (in pixels). However, using the back-projection described in section 4.2, subjects of the pretest (section 5.4.1) disapproved with the "keyhole-effect" caused by the circular overall structure the reconstruction takes on. Though the visible area was not much reduced, most of them said the image structure was difficult to make out. Therefore the raw samples were used for training in further experiments.

## 3.3 Memorizing feature vectors of images

After the approximation phase, all windows are projected onto the local PCA – representation and are memorised in the search database. For each window $\vec{W}(i,j)$ (i.e. number $j$ of image $I_i$) first the best match reference vector $\vec{r}_k$ has to be found after Eq. (9). Then $\vec{W}(i,j)$ is projected onto the local PCs $\vec{w}_1^{(k)} \ldots \vec{w}_{N_{PC}}^{(k)}$:

$$\left( \vec{P}(i,j) \right)_l = \sum_{n=1}^{d} \left( (\vec{W}(i,j))_n - (\vec{M}^{(k)})_n \right) (\vec{w}_l^{(k)})_n \qquad \text{for} \quad l = 1 \ldots N_{PC}, \tag{13}$$

$$\text{where} \quad k = \mathcal{BM}(\vec{W}(i,j)). \tag{14}$$

Again, $\vec{M}^{(k)}$ denotes the mean value of Voronoi tessellation cell number $k$. The projection $\vec{P}(i,j) \in \mathbb{R}^{N_{PC}}$ has a much lower dimension than the original window $\vec{W}(i,j) \in \mathbb{R}^d$, if only the PCs with large eigenvalues

are used. $\vec{P}(i,j)$ is memorised together with the number of the best match reference vector, $k$, for the corresponding image. See also Fig. 2.

Hence, an entry for image number $i$ in the search database has the following components:

1. Image identifier $i$ (a number).
2. Number of local windows found by SYM and HARRIS: $N_L(i)$.
3. Information for each window $\vec{W}(i,j)$, $j \in [0, N_L(i)]$, where $j = 0$ denotes the global window:
   (a) Center position $(x, y)$
   (which is the image center for the global window).
   (b) Window dimension in the *original* image (before sub-sampling):
   $0.9 \cdot \min(w,h)$, $128 \times 128$, $64 \times 64$ or $32 \times 32$.
   (c) Number $k = \mathcal{BM}(\vec{W}(i,j)) \in [1, N_{VQ}]$ of the best match reference vector.
   (d) Projections onto the local PCs of $\vec{r}_k$: $\vec{P}(i,j) \in \mathbb{R}^{N_{PC}}$.

With this information, unknown windows can be searched in the database, as described in the next section.

# 4 Image retrieval

First, an image retrieval session with ColoSeek will be outlined from the point of view of a user. Then it will be described how the system works internally (section 4.2).

## 4.1 A search session

A search using ColoSeek starts by providing a set of query images. Then the user defines the object to be searched for by one or several windows. "Object" means not only physical objects like a cup or a face, but also scenery like a beach, the sea and blue sky. Four different window sizes are available: *Huge, Large, Medium* and *Small*, which correspond (internally) to the windows defined in Table 1. The user can position one of the available windows to cut out a certain image patch. This can be repeated arbitrarily often and with varying window sizes.

To show the user how the machine "understands" the selection of windows, the search state visualisation (SSV) is displayed as a system feedback. It shows a superposition of the selected window projections to give an impression of the constructed scenery (see next section). Weight of relevance can be assigned to each window, the effect of this weighting becomes visible in the SSV. Moreover, windows already selected may be removed.

The system does not accept all specified windows, but may reject a new one because of too great dissimilarity with previously defined windows. Rejected windows may, however, be reused in a separate search in *complex queries* as described in section 4.4.

The user can start the search at any time. In this case a pre-defined number of result images is displayed in order of similarity. In the images the window which was the best match to the query is indicated. Now, the result images can be treated equally as the initial query images: Windows can be selected, weighted using the SSV or discarded. This process can be repeated until satisfactory images are found.

## 4.2 Search- and visualisation algorithm

The algorithm will be first described qualitatively, then in more detail.

### 4.2.1 Qualitative description

A query consists of a set $\Lambda$ of $N_Q$ windows: $\Lambda = \{\vec{Q}_1 \dots \vec{Q}_{N_Q}\}$. The notation $\vec{Q}$ is used as a distinction from the memorised windows. All query windows are first transformed to the coordinates of the local PCA – representation. Therefore, for all $\vec{Q}_i$ the best match reference vectors are searched. If the query set $\Lambda$ is "homogeneous" in terms of the local PCA – representation, all $\vec{Q}_i$ have the same best match reference vector $\vec{r}_{k^*}$. This situation, however, is not likely in practice. Since the local PCA – coordinates of different reference vectors are not comparable, a decision has to be made which PCs are to be used for projection. Here, the PCs of the reference vector $\vec{r}_{k^*}$ which is best match to most of the $\vec{Q}_i$ is chosen. The rest of the windows in the query set $\Lambda$, i.e. the $\vec{Q}_i$ which are best match to reference vectors other than $\vec{r}_{k^*}$, are discarded for this search. The user is informed about this and may reuse the rejected windows in an additional search as outlined in section 4.4.

The set of the remaining query windows which match to $\vec{r}_{k^*}$ is called $\Lambda^+$. All query windows $\vec{Q}_i \in \Lambda^+$ are now projected onto the local PCs of $\vec{r}_{k^*}$, the resulting $N_{PC}$ – dimensional projections are denoted $\vec{\mathcal{P}}_i$. As depicted in Fig. 7, from the $\vec{\mathcal{P}}_i$ now a weighted average $\vec{\bar{\mathcal{P}}}$ is computed. $\vec{\bar{\mathcal{P}}}$ is the "average query window" for which the database will be searched.

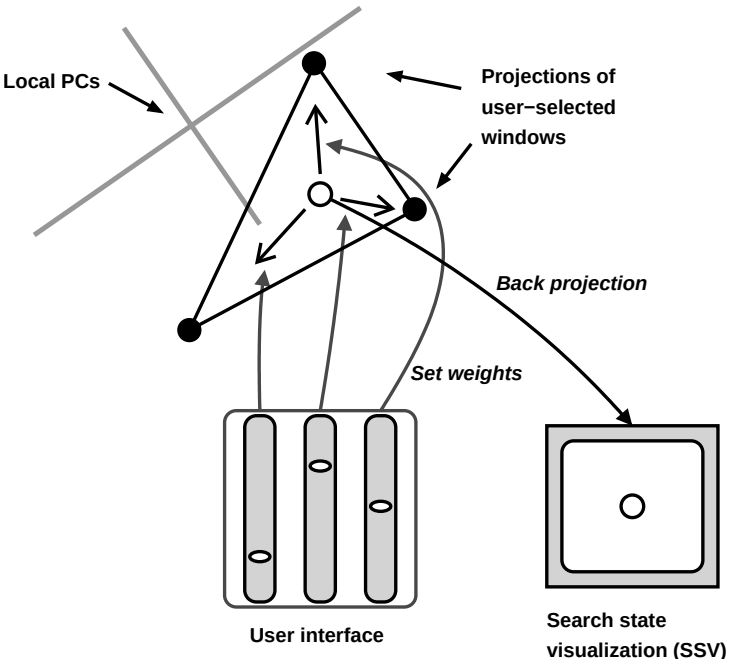


Figure 7: The user interface allows an interactive relevance weighting of the chosen query windows $\vec{Q}_i$ using sliders. The weights $\xi_i$ are the coefficients in the linear combination of the query window projections $\vec{\mathcal{P}}_i$ onto the local PCs. The resulting weighted average of the projections, $\vec{\bar{\mathcal{P}}}$, is back-projected to the pixel space as a — still rather coarse — system feedback.

As $\vec{\vec{\mathcal{P}}}$ is the "concentrate" of the user's input, it has an outstanding function in the search process. Therefore, it is visualised in the SSV as a back-projection to the pixel space. This can be simply done if the components of $\vec{\vec{\mathcal{P}}}$ are regarded as coefficients in a linear combination of the PCs $\vec{w}^{(k^*)}$ of $\vec{r}_{k^*}$. Visualisation then requires only normalisation to the common 24-bit format. To give more or less weight to single query windows $\vec{Q}_i$ (more precise: their projections $\vec{\mathcal{P}}_i$), the user can assign weights $\xi_i$ between 0 and 1. The resulting $\vec{\vec{\mathcal{P}}}$ is again visualised.

When the actual search is activated, the distances between the current $\vec{\vec{\mathcal{P}}}$ and the memorised projections $\vec{P}(i,j)$ in the search database are calculated. The best result image is the one that has the window $\vec{W}(i^*,j^*)$ with projection $\vec{P}(i^*,j^*)$ closest to $\vec{\vec{\mathcal{P}}}$. This image $I_{i^*}$ is shown first and the window is indicated, then the second best image and so on.

### 4.2.2 Detailed description

1. The user selects a set $\Lambda$ of $N_Q$ windows: $\Lambda = \{\vec{Q}_1 \dots \vec{Q}_{N_Q}\}$.

2. Find the best match reference vector $\vec{r}_{\mathcal{BM}(\vec{Q}_i)}$ for each $\vec{Q}_i \in \Lambda$.

3. Find the reference vector $\vec{r}_{k^*}$ which is best match for **most** of the $\vec{Q}_i \in \Lambda$,
   i.e. the "most frequently accessed" one for this query:

$$k^* \;=\; \arg \max_{l=1\dots N_{VQ}} \left| \left\{ \vec{Q}_i \in \Lambda \;\mid\; \mathcal{BM}(\vec{Q}_i) = l \right\} \right| . \tag{15}$$

4. Decompose $\Lambda$ into two subsets $\Lambda^+$ and $\Lambda^-$ with $\Lambda^+ \cup \Lambda^- = \Lambda \;\wedge\; \Lambda^+ \cap \Lambda^- = \emptyset$:

   - $\Lambda^+$ contains all $\vec{Q}_i$ for which $\vec{r}^*$ **is** best match:
     $\Lambda^+ \;=\; \{\vec{Q}_i \in \Lambda \;\mid\; k^* = \mathcal{BM}(\vec{Q}_i)\}$.
   - $\Lambda^-$ contains all $\vec{Q}_i \in \Lambda$ for which $\vec{r}^*$ is **not** best match:
     $\Lambda^- \;=\; \{\vec{Q}_i \in \Lambda \;\mid\; k^* \neq \mathcal{BM}(\vec{Q}_i)\}$.

5. Tell the user for all $\vec{Q}_i \in \Lambda^-$ that they were rejected, but could be reused in a separate query (using a *complex query*).

6. Calculate the projections $\vec{\mathcal{P}}_i \in \mathbb{R}^{N_{PC}}$ of all $\vec{Q}_i \in \Lambda^+$

$$\left(\vec{\mathcal{P}}_i\right)_l \;=\; \sum_{n=1}^{d} \left( (\vec{Q}_i)_n - (\vec{M}^{(k^*)})_n \right) \, (\vec{w}_l^{(k^*)})_n \qquad \text{for} \quad l = 1 \dots N_{PC}, \tag{16}$$

   in the same way as the projections of the memorised windows, see Eq. (13).

7. Get the weighting coefficients $\xi_i$ for each $\vec{Q}_i \in \Lambda^+$ from the user interface (weighting-sliders). Then calculate the weighted average projection $\vec{\vec{\mathcal{P}}}$ of the query window projections:

$$\vec{\vec{\mathcal{P}}} \;=\; \frac{1}{|\Lambda^+|} \sum_{\{i \,\mid\, \vec{Q}_i \in \Lambda^+\}} \xi_i \, \vec{\mathcal{P}}_i \, , \tag{17}$$

   where $0 \leq \xi_i \leq 1$.

8. Search state visualisation (SSV):

   (8a) Calculate the back-projection $\vec{B} \in \mathbb{R}^d$ of $\vec{\tilde{\mathcal{P}}}$ to the pixel space:

$$\vec{B} \;=\; \sum_{i=1}^{N_{PC}} (\vec{\tilde{\mathcal{P}}})_i \, \vec{w}_i^{(k^*)}. \tag{18}$$

   (8b) Transform $\vec{B}$ linearly to $0 \ldots 255$ for the three colour channels and visualise as an image in the SSV.

   (8c) If the user changes the weights, `goto` 7.

9. Find and display result:

   (9a) Find the best image $I_{i^*}$ with the best match window $j^*$ by minimising the distance between $\vec{\tilde{\mathcal{P}}}$ and the memorised projections $\vec{P}(i,j)$:

$$i^* \;=\; \arg \min_{i=1\ldots N_I} \left( \min_{j=0\ldots N_L(i)} \|\vec{\tilde{\mathcal{P}}} - \vec{P}(i,j)\| \right), \tag{19}$$

$$j^* \;=\; \arg \min_{j=0\ldots N_L(i^*)} \|\vec{\tilde{\mathcal{P}}} - \vec{P}(i^*,j)\|, \tag{20}$$

   with $\vec{P}(i,j)$ from Eq. (13).

   (9b) Get the position and size of window $j^*$ from the search database.

   (9c) Find in the same way the "second best", "third best" images etc.

   (9d) Display all images and their windows.

10. `Goto` 1.

## 4.3 Speedup of the search

For simplicity, so far the search database was described and used in the form of section 3.3. But obviously a considerable speedup can be achieved when the database is not indexed by image numbers but by reference vector numbers. In this case, searching for a weighted query projection $\vec{\tilde{\mathcal{P}}}$, only the memorised projections $\vec{P}$ of the reference vector $\vec{r}^*$ have to be evaluated. Additional speedup can be achieved when the projections $\vec{P}$ are stored in a way such that a tree-structured search is facilitated. However, details of the implementation aimed to computational speed are not in the scope of this contribution.

## 4.4 Complex queries

So far, only a search for a single object or scenery was described. It is easy to combine several searches using the boolean operators AND, OR and NOT. In this case, it is possible to search, e.g., for a scene like a beach and combine it using AND with a search for a palm tree. Then the system searches for an image which has windows of both kinds.

Rejected windows in $\Lambda^-$ may be used as a second query and can be combined with the original query ($\Lambda^+$) using OR. Another useful operator for complex queries is ATLEAST($n$). An example of this type is shown in Fig. 11.

The last operator is MIRROR($\alpha$). If MIRROR($\alpha$) is applied to a query window one or several times, ColoSeek makes internal copies of the window which are mirrored at an axis through the center with angle $\alpha$ to the horizon. These new windows are added to the query. By this means, it is sufficient to supply one image of an asymmetric object — e.g. a head looking to one side — when also the mirrored version should be searched for. MIRROR(90°) was applied to all windows in Fig. 10 (not depicted).

### 4.5 Discussion of the retrieval algorithm

Two features of ColoSeek give rise to further discussion: Query window rejection and the use of $\vec{\mathcal{P}}$ as a starting point for the search.

Rejection is part of the system feedback to show the user what kind of windows fit together. In addition to the SSV, the rejection makes clear what the system "thinks" to be similar. Since rejection is not final, but refers only to a particular set of query windows, it can be considered as a "suggestion" of the system to structure the query by use of the operators described in section 4.4.

A problem is that in the rare case where a set of query windows is close to the border of two Voronoi tessellation cells, it might be divided by the rejection mechanism in a counterintuitive way. Refinements of the algorithm might minimise this risk, e.g., query windows could be divided into clusters instead of being assigned to reference vectors, but the problem cannot be completely avoided as long as a hard assignment is used. A soft assignment could overcome this limitation, however, it is doubtful that the user would profit from such detailed information about window similarities.

The weighted averaging of windows to obtain $\vec{\mathcal{P}}$ leads to a loss of information. As an alternative, for each query window an individual set of results could be searched for. The motivation to use $\vec{\mathcal{P}}$ is to concentrate the search, to facilitate the use of relevance weights and to obtain only one manageable set of ranked results. However, in a future version an additional option for experienced users may be helpful that returns separate sets of ranked results for single query windows or clusters of query windows.

## 5 Results

In this section ColoSeek will be applied to a database of 8000 images and evaluated for efficiency and robustness.

### 5.1 Image database

As a database, part of the "photos" section of the "Art Explosion® 600000 Images" by Nova Development Corp. was used. This section contains about 90000 natural images covering a wide range of common topics like *cityscapes, people, sports, landscapes, animals, flowers* or *industry*, but also highly special topics like *firefighting* or *balloons*. From the common topics, a subset of 8000 images was selected, the special topics were omitted. This subset was used for the following experiments. The automatic window selection described in section 2 yields 222302 windows for this database.

### 5.2 Calculating the local PCs

The key parameters of local PCA are the number of reference vectors $N_{VQ}$ and the number of local PCs, $N_{PC}$. For the purpose of object recognition, it could be shown e.g. in [22] that the classification performance is "well-behaved" in both parameters, i.e. the classification rate rises smoothly with $N_{VQ}$ and $N_{PC}$ until saturation is reached. However, for image retrieval the situation is more difficult: On the one hand, $N_{VQ}$ should be large to approximate a complex data distribution as good as possible. On the other hand, windows of a query will be spread out the more over the reference vectors the larger $N_{VQ}$, which means even the reference vector with the most hits has relatively few. Up to now, this problem has to be solved by trial and error. The only hint to a reasonable maximum number of reference vectors is obtained from the AEV algorithm: If AEV is applied with too many initial reference vectors, they are automatically reduced until a homogeneous distribution can be reached. For the current database $N_{VQ} = 16$ proved to be a suitable value.

The choice of $N_{PC}$ is less difficult, because the eigenvalues help to estimate how much of the total variance is represented for a certain value of $N_{PC}$. Moreover, too large values for $N_{PC}$ only consume memory and computational speed but do not have much effect on the results. This is because PCs with low eigenvalues have little influence on the structures in the projection space. For this reason, a common, sufficiently large value of $N_{PC} = 40$ could be used for all 16 PCA-nets. Training the PCA-nets according to the rule given by Eqs. (11,12) over 3000000 steps takes about three days on a standard Pentium III – 1 GHz – machine in a non-optimised implementation.

Part of the resulting local PCs is shown in Fig. 8. A semantic interpretation of the local PCs is difficult, specialisation to certain types of objects is hardly visible. An exception are the first PCs, some of which are specialised to frequent overall structures of images. For example, the fifth reference vector in Fig. 8 corresponds to a landscape-like scenery with dark ground and blue sky. It is the best match for the first query in Fig. 9. A feature common to all reference vectors is that with decreasing eigenvalues (i.e. increasing PC-number) the spatial frequencies represented by the local PCs increase. Hence, taking into account small-eigenvalued PCs captures more details of a window.

For comparison, also a data approximation by normal PCA (special case $N_{VQ} = 1$) was computed. The resulting PCs lead to a better and more detailed reconstruction $\vec{B}$, which seems to be an advantage at first glance. But a precise reconstruction of superposed windows is not the aim of the SSV — this would be done better without any projection. Rather, the local structure of the windows of the database *which can actually be searched for* should be displayed. Since the search results using normal PCA are much worse, this approach was not pursued any further.

### 5.3 Examples for search results

Fig. 9 shows three queries and the ten best result images. The first task is quite easy, because the sample image structure is clear (grey or brown ground, clear horizon, blue or very light blue sky). Moreover, there are lots of images in the database which have approximately this overall structure. All results were chosen because of the global window $W_G$.

The second query — a sliced orange — yields only one true positive, but there is only one more sliced

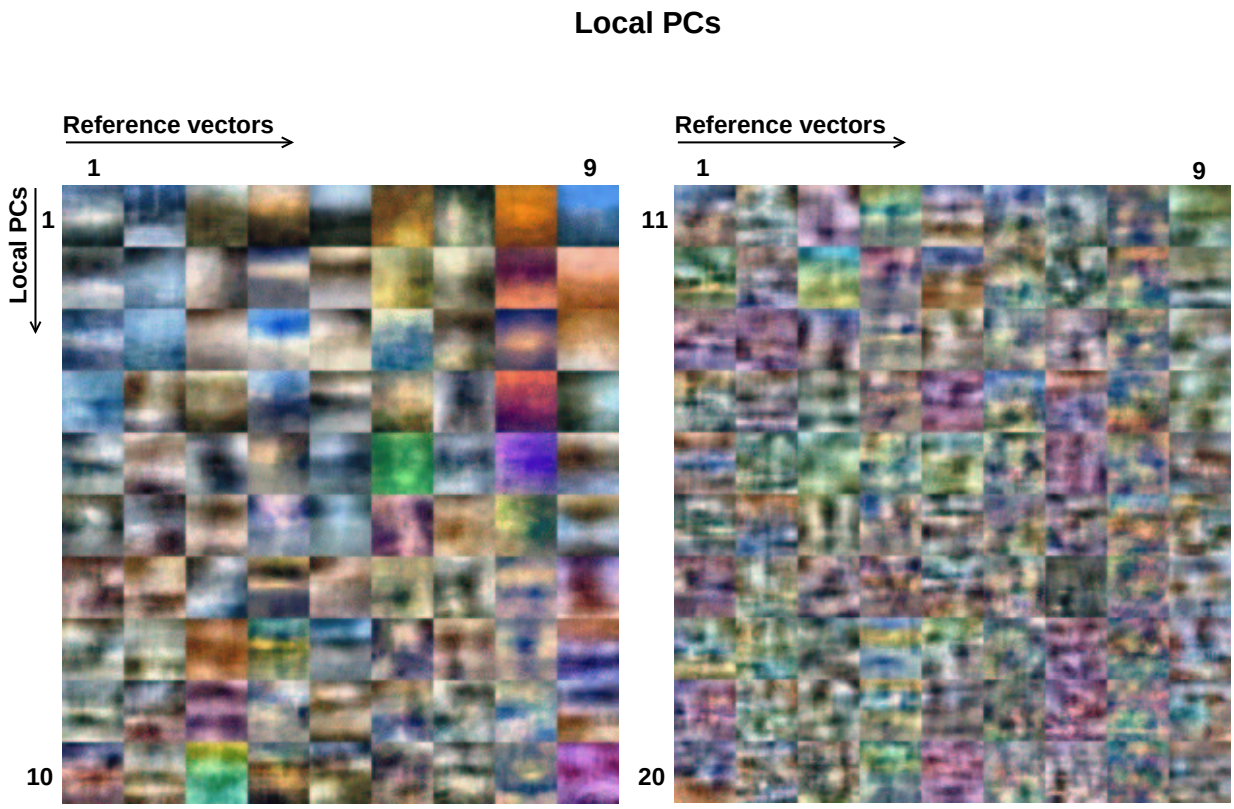


Figure 8: Part of the local PCs which serve as feature detectors for the database outlined in section 5.1. The weight vectors $\vec{w}_i^{(m)}$ $(m = 1 \ldots 9,\ i = 1 \ldots 20)$ were linearly transformed to 24 bit RGB values for visualisation. Note spatial frequencies increase with decreasing eigenvalues.

orange in the database, while there are lots of "distractors" like the helmet. This example demonstrates that false positives are understandable for the user because the colour features capture local colour structures. The last query is the most complex because the blue flowers have different colours and a high shape variance. Considering the difficulty, results are quite good.

Fig. 10 illustrates the use of a complex query. Since the object has a great appearance variety, five categories of query windows are selected and coupled by ATLEAST(3), which means that only result images are accepted for which the best match windows are spread out over at least three of the categories. In the result images the windows are indicated together with the perceived categories.

At first glance it seems that combining several windows to formulate a query requires much effort. However, the required time is small compared to the time saved by the window-based approach: Actually, once the user is familiar with the interface, window selection and combination takes less than two minutes for the above example. By comparison, searching an unknown database for suitable query images may take much longer, especially for systems that evaluate the *complete* query images and thus require possibly good matches. In contrast, the window based approach saves much time because only query images comprising the *components* of the target have to be found (e.g., to find a sports car in front of a villa, a window of a sports car plus a window of a villa is sufficient). The time saved in this process is large compared to the time to cut out and combine windows.

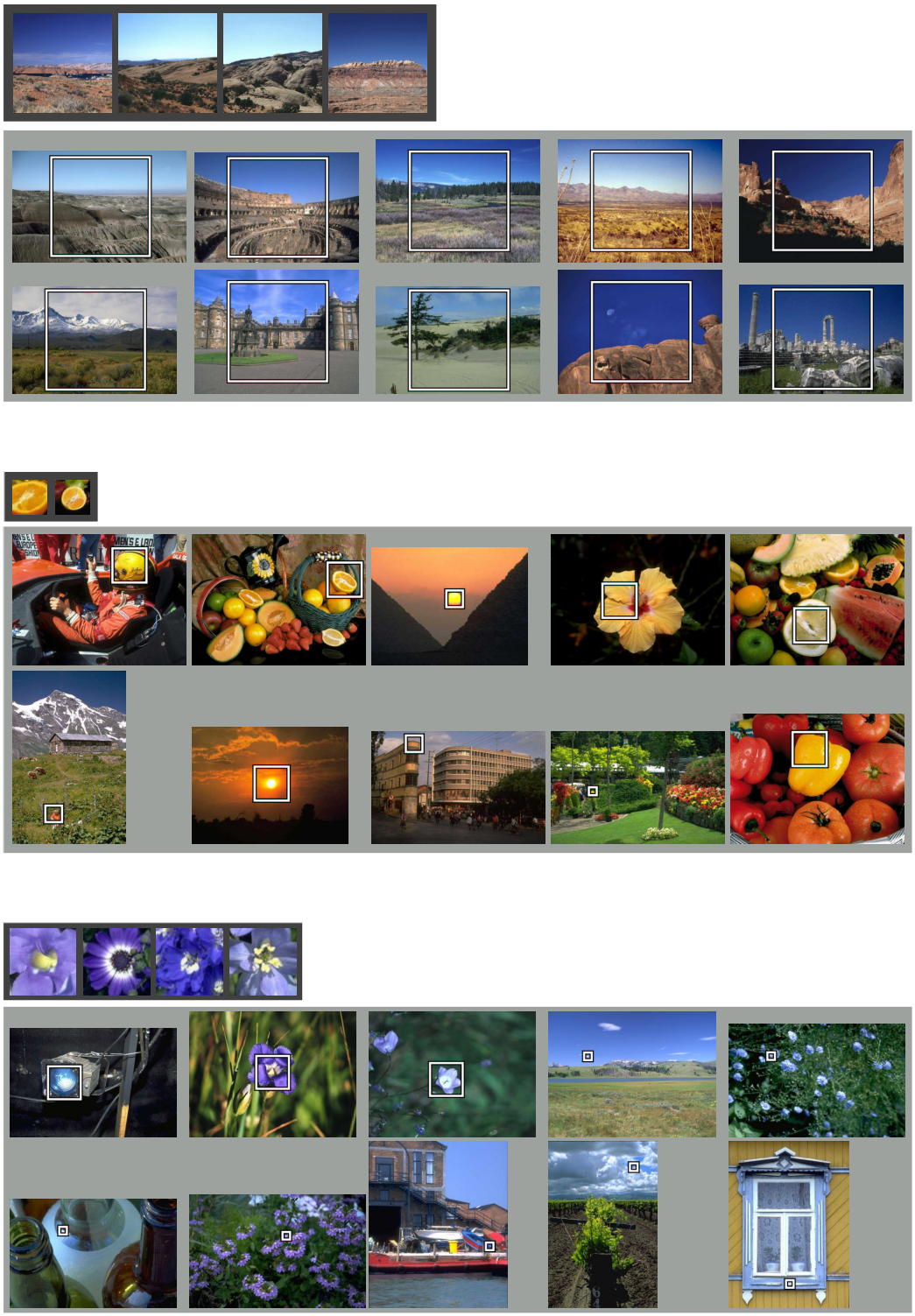

Figure 9: Results of three queries. The query windows are the ones on dark grey background, they are printed equal in size though resolutions differ. For an interpretation see section 5.3. It becomes clear in almost every case *why* a certain image was chosen.

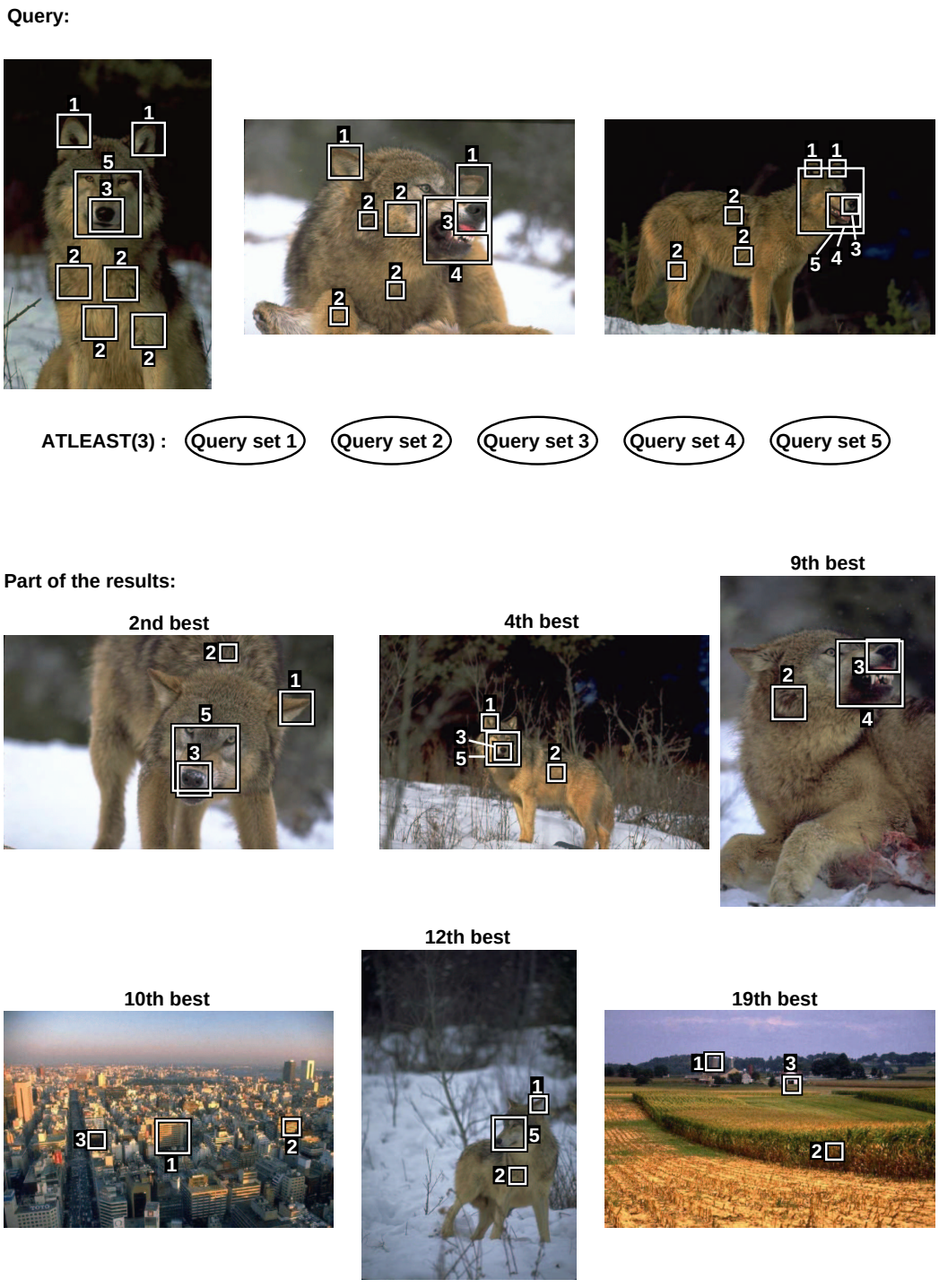


Figure 10: A complex query: Of three query images five types of windows are cut out to form query sets: 1 – ears, 2 – fur, 3 – nose, 4 – snout, 5 – face. These sets are connected using the ATLEAST(3) operator, which means that at least three types of windows must be found in a result image. By this means even a complex object can be searched for by selection of typical details.

## 5.4 Experiences of users

ColoSeek was tested qualitatively by ten subjects in two sessions. All users were non-experts in image retrieval but had at least once before tried out a retrieval system available on the Internet. The first session was a pretest to find out whether the basic man-machine interaction works. From this experience, the system design was refined. The main test took place in a second session, where user interaction with ColoSeek was logged.

### 5.4.1 Session 1 (Pretest)

In the first session, the aim and functionality of the system was explained to the subjects, a sample query was demonstrated and eight given search tasks had to be solved, with assistance if necessary. For the first four of the search tasks, the sets of query images were supplied, for the other four the subjects had to search by themselves using the textural categorisation provided in the Art Explosion© Gallery.

An important item was to test how much colour contributes to the search results. Therefore, at the end of the session four additional search tasks had to be solved using a grey value based version of the system. For this experiment, the local PCA had been trained only on the grey values. The query images and windows were still displayed in colour. At the end of the session the subjects were interviewed.

**Results of the pretest:**

- Finding suitable query images is often difficult, but this is a general drawback of sample based queries and not in the scope of this paper.
- The rejection of certain windows at first confused the subjects. It is often not intuitively understandable, why a certain window does not fit in the scheme established by the others. However, when explained, rejection helped the users to forget their "semantical point of view" and to become familiar with the low level similarity measures of the machine — a change of mind which must be considered helpful as long as no real semantical understanding can be realised in a machine.
- Eight subjects expressed they were disillusioned about what a machine can search for. They had expected much more understanding on the symbolic level. They had assumed when an image of a bicycle is shown the system searches — somehow — for bicycles, not *features*. However, all of them expressed that the disillusion brought about by the visual feedback was helpful. Though the machine couldn't do all they wanted, they were now enabled to exploit at least what it *could* do.
- The training windows for the local PCA cannot be masked by a Gaussian, because this leads to a "looking through a keyhole" – effect (section 3.2).
- Use of the SSV is helpful only for about up to eight sample windows, then the visualisation becomes too much blurred. For the main test, this restriction was automated.
- Results for the grey value based representation were extremely bad, so this item was discarded in the main test.

Table 2: Average user results in the main test.

| Subjects use complex search: | Yes | No |
|---|---|---|
| Success rate | 81% | 52% |
| Avg. time until success: | 6.6 min | 8.2 min |
| Avg. time until failure: | 10.2 min | 9.1 min |
| Add. time to find query images: | + 5.2 min | + 6.1 min |

#### 5.4.2 Session 2 (main test)

For the second session, ColoSeek was changed according to the pretest results. Eight given search tasks had to be carried out, for half of the tasks the query images were supplied. For the other half suitable query images were "hidden" within a set of 1000 images separate from the actual database, whereas in the pretest the subjects had been allowed to use the whole database. After a short repetition of the introduction, no further help was given. A search was considered completed when three images of the object could be found. The subjects were told when they could stop searching. There were no time constraints, so the subjects had to decide by themselves when to give up a search.

**Results of the main test:**

- One subject did not manage to use the system by himself.
- The other subjects could be clearly divided into two groups: Those who used complex search and those who didn't.
  - Subjects who don't use complex queries are less successful.
  - Subjects who do use complex queries use this facility more with growing experience and decompose tasks into several queries as illustrated in Fig. 10.
- See Table 2 for success rates and times.
- The feedback by the SSV is very useful in the beginning, since the users learn that the machine searches for colour patterns instead of semantics, but it is too coarse to be helpful for more complex tasks.

The last point is illustrated in Fig. 11, which was derived from the protocol of an actual user search: At first, various windows of red peppers are chosen, which show the red peppers from the side and with much background. Since the SSV shows an unstructured red surface, the user tries to capture only the most typical part of the red peppers, which leads to more structure in the SSV. Remarkably, even the poor feedback has led the subject to the right idea.

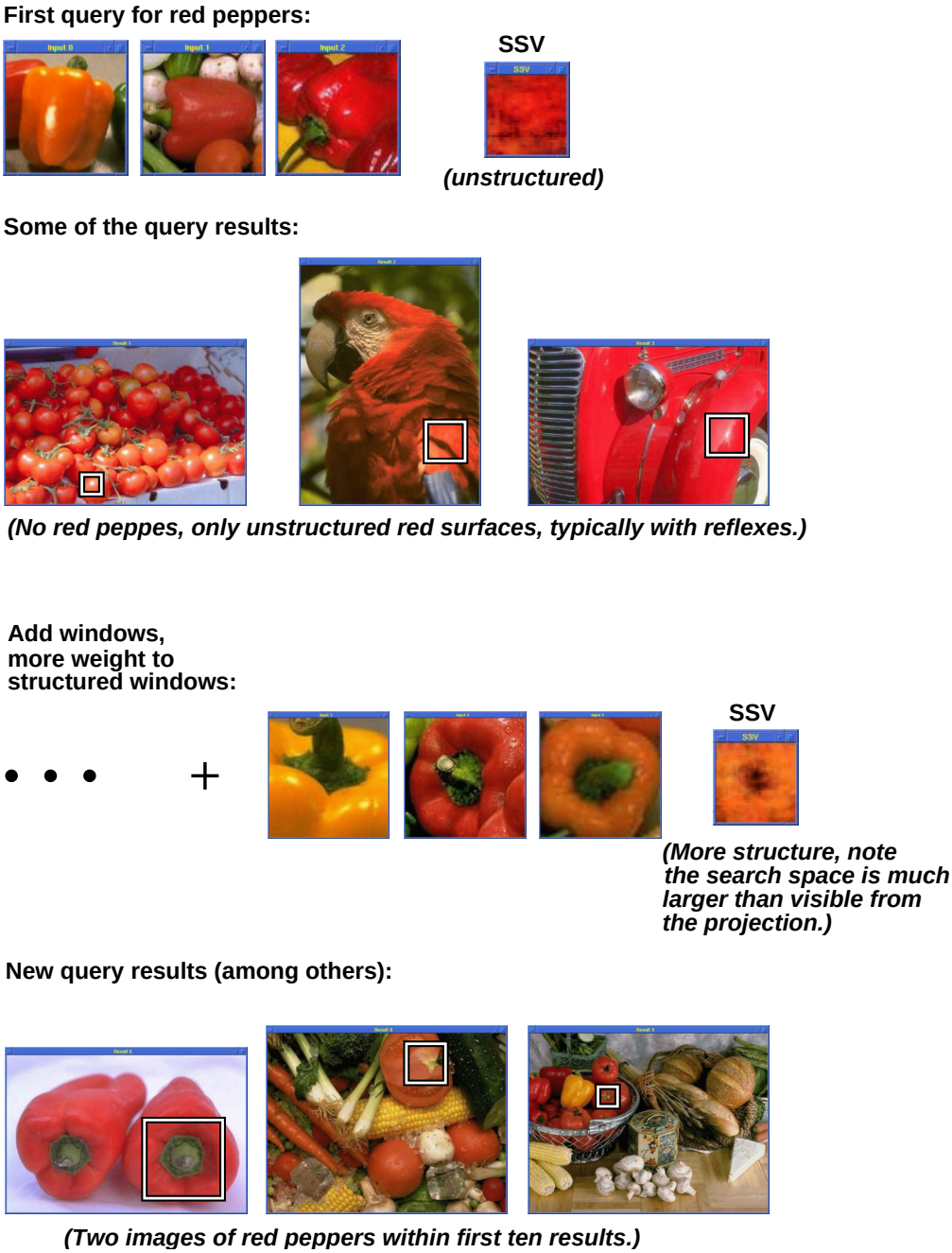


Figure 11: After a protocol of a retrieval session. The user is asked to find red peppers. The first query images are not very useful as they lead only to an unstructured red surface, as visible in the SSV. The user refines the search until two images with red peppers are found. Remarkably, even the poor feedback of the SSV leads the user to the right idea.

#### 5.4.3 Robustness against viewpoint variations: User test

In a third experiment, retrieval performance was tested for robustness against different viewpoints. Though in natural image galleries like Art Explosion® several images of the same object can be found, such a database is not suitable for this experiment because viewing conditions (angles, lighting etc.) are unknown and un-reproducible. Therefore, as an image database offering controlled conditions, the Columbia Object Image Library COIL-100 [74] was used. The library consists of RGB-images of 100 objects presented on a turntable, which is rotated by 360° in steps of 5°, so there are 7200 images in total. Since the resolution of the COIL-100 images is $128 \times 128$ and matches the size of $\vec{W}_L^3$ (section 2.2.2), $\vec{W}_L^3$ was omitted in this experiment.

After ColoSeek was adapted to COIL-100, the subjects of the preceding experiment — who had now some experience — got the task to find ten of the objects in the database. They had only one try and a fixed set of three query images of each object, showing the object under $0°, 120°$ and $240°$. They were asked to put four windows on each query image, so the search comprised twelve query windows. Since 69 images of each object remain, the 69 best results were computed. On average, 86% of the result images were correct. So it could be shown that a "real user" can achieve good results, though better retrieval rates are possible when the windows are placed optimally.

### 5.5 Robustness against varying imaging conditions

While in section 5.4 the usability of ColoSeek was evaluated, this section presents user independent tests of the robustness against noise, variations of illumination, partial occlusion and changes of viewpoint.

#### 5.5.1 Overview of the tests

As a test database providing controlled conditions, images of 20 household items and toys (Fig. 12) were used. Each object was placed in isolation on uniform grey background, then images were taken under six different conditions:

1. Reference image: Frontal lamplight.
2. Noise: For 30% of the pixels of the reference image random numbers between $-20$ to 20 were added up for each RGB-channel separately (the range of RGB values being $0 \ldots 255$).
3. Different lighting direction: The same source of lighting, but under 30 degrees from one side.
4. Different source of lighting: Illumination from neon lamps.
5. Partial occlusion: Within the windows centered at the interest points of the reference image, 20% of the window area were covered by randomly chosen patches from a different image domain (see section 5.5.2).
6. Change of viewpoint: The object was rotated by 30 degrees under the reference conditions.

So the database comprises in total 120 images. Since ColoSeek does not search for images but for windows, ground truth cannot be established in the sense of labelled image domains, as e.g. in [6]. Instead, the

reference are the "undisturbed" results under condition 1. Another consequence of the window-based approach is that the influence of imaging conditions on ColoSeek has to be tested for two different aspects:

- Effects on the locations of interest points, which define the possible targets of a search.
- Effects on the retrieval performance for stable interest points that remain search targets.

The following sections address both points.


Figure 12: Set of objects used for testing robustness against noise, variations of illumination, partial occlusion, and changes of viewpoint.

### 5.5.2 Robustness of interest point detection

Let the reference image of an object (condition 1 of the database described in the last section) be denoted by $F_1$. The disturbed version of $F_1$ after, e.g., adding noise or changing lighting will be denoted by $F_2$ (conditions 2 to 6 in the last section). For all images of each object, interest points are generated by the methods described in sections 2.2 and 2.3. A measure for the disturbance of $F_2$ compared to $F_1$ is the *$\varepsilon$-repeatability rate* $r(\varepsilon)$ [59], which denotes the fraction of interest points $p^1$ in $F_1$ that can be found also as points $p^2$ in $F_2$ within an $\varepsilon$-neighbourhood of the original position:

$$r(\varepsilon) \;=\; \frac{\left|\{p_i^1 \text{ found in } F_1 \;\mid\; \exists\, p_j^2 \text{ found in } F_2 \;\wedge\; \|p_i^1 - p_j^2\| \le \varepsilon\}\right|}{N_p}, \tag{21}$$

where $N_p$ is the number of interest points $p^1$ in $F_1$. Interest points appearing in $F_2$ without correspondence in $F_1$ have no influence on $r(\varepsilon)$, because additional target windows are not considered a disadvantage for retrieval.

Table 3 shows the repeatability rates $r(\varepsilon)$ for a tolerance $\varepsilon = 4$ pixels. Noise has only minor influence on the interest points, the second best results are achieved for changing the light source, which is mainly a change of the spectral colour distribution. In principle, the grey-value based interest point detection

Table 3: Repeatability rates for interest points.

| | Noise | Lighting dir. | Light source | Occlusion |
|---|---|---|---|---|
| SYM(R=50) | 96% | 71% | 84% | 64% |
| SYM(R=25) | 95% | 73% | 79% | 59% |
| SYM(R=12) | 93% | 73% | 81% | 65% |
| HARRIS | 93% | 79% | 88% | — |

Table 4: Pseudo-repeatability rates for viewpoint variation.

| | $\varepsilon_P$ |
|---|---|
| SYM(R=50) | 72% |
| SYM(R=25) | 75% |
| SYM(R=12) | 80% |
| HARRIS | 91% |

should be undisturbed by spectral changes, but in the experiment the single-bulb lamp (reference) had to be replaced by several neon lamps, leading to a more diffuse illumination with less shadows which causes the major part of the effect. Otherwise repeatability rates would be significantly better.

The influence of lighting direction is stronger since shadows and highlights are subject to significant changes when a single light source is moved. SYM is more affected than HARRIS, because the symmetry of edge patterns on the scale of $R$ is more likely to be disturbed than localised corner or edge points as a such. Occlusion was brought about artificially by copying randomly chosen rectangular patches from the Art Explosion® gallery into the $F_2$-images. The patches were positioned such that 20% of the window centered at the interest point in $F_1$ is covered. Since partial occlusion is the most severe disturbance, results are worse than for the other imaging conditions. For HARRIS, no repeatability rates were calculated under occlusion, because HARRIS evaluates only a very small image area for a particular interest point. Hence, partial occlusion is highly unlikely in reality — the area is either occluded or not.

The $\varepsilon$-repeatability makes sense only as long as the scene geometry is unaffected. To test the influence of viewpoint variation, the objects were rotated by 30 degrees. Stability rating of the interest points must in this case be based on human judgement, i.e. a "pseudo-repeatability" $\varepsilon_P$ is calculated as the fraction of interest points of $F_1$ that are found in $F_2$ at approximately the same object location (not the same pixel location). Table 4 shows the results, interest points detected on small scales are the most stable.

The overall result of the robustness test for interest points is satisfactory. Related investigations on

Table 5: Stability of queries for varying imaging conditions.

| Noise | Lighting dir. | Light source | Occlusion | Viewpoint |
|---|---|---|---|---|
| 99% | 87% | 91% | 61% | 76% |

interest point stability were carried out in [59, 50].

#### 5.5.3 Robustness of query results

To test the robustness of the local PCA representation independently from the windows selected by a user, "artificial" queries were carried out. As query windows served the windows detected in the reference images $F_1$, then ColoSeek searched among the windows detected for one of the other imaging conditions. To judge the performance of the local PCA representation independently from the interest point detection, only those windows of $F_1$ were used for the query which proved to be stable in the tests of the previous section.

Table 5 shows the results, i.e. the fraction of $F_1$-windows to which the corresponding $F_2$-windows could be found. Results are good except for occlusion which obviously takes too much of the information. For viewpoint variation, the correspondence between $F_1$ and $F_2$ had again to be established manually.

## 6 Conclusions

The system ColoSeek for image retrieval was presented. It is based on an automatic selection and representation of salient image windows. A user can specify queries in terms of windows selected arbitrarily from sample images. ColoSeek relies on feature extraction by a local PCA – representation which is obtained by unsupervised learning methods from the image database. The major benefit of local PCA is that it exploits spatial colour information automatically on the scale on which most variance can be found, so both large scale spatial colour distributions as well as texture can be captured.

The benefits as well as the shortcomings of the current system lead to many open questions and future research topics:

1. The achievable results of ColoSeek are limited to the windows selected by the saliency detection. This proved to be no serious limitation so far, but more saliency features like homogeneous regions of high colour contrast will be integrated.

2. As ColoSeek searches only for windows, its capabilities as a stand-alone system are limited. Integration into larger search machines is a future goal.

3. A major benefit of ColoSeek is that most results are *understandable*. It is evident why they were chosen. The user is not confronted with — from a semantical point of view — absurd results which were produced by an unknown feature extraction. Mistaking the sun for an orange may be not satisfactory, but subjects don't feel confused.

4. By rejecting windows that do not fit to others in the query, ColoSeek forces the user to limit queries to a realistic complexity. Simultaneously, the user gets a feeling what is judged to be similar by ColoSeek.

5. Still, the visualisation capabilities of ColoSeek are limited, because only one point of the $N_{PC}$-dimensional search volume is back-projected. In an additional experiment trajectories around $\vec{\mathcal{P}}$ through the space spanned by the local PCs were visualised to give an impression of the variety the search involves. But this method is still unsatisfactory, because only a very small part of the volume can be covered, and the sequence of visualisations is counterintuitive.

6. Surprisingly, the system feedback was nevertheless extremely useful: It helped the subjects to forget their expectations concerning a search based on semantical understanding. Instead, they accepted that only colour patterns can be searched for and treated the system adequately.

7. This leads to the question if also other reliable search features could be visualised. For example, the colour information comprised in a histogram could be displayed as an artificial image of appropriately coloured "patches".

8. The number of tested subjects is evidently much too small for generalisations, but the test shows that human-machine interaction must be a major research topic in CBIR and cannot be treated as a minor add-on after a system was designed.

The main difference to other systems is that ColoSeek intentionally destroys the illusion the search would be based on an understanding of semantics. This feature is not an antithesis to attempts to build systems which represent the semantic level and bridge the gap to the level of features. Instead, ColoSeek tries to make the so far "black box" of retrieval systems become a "transparent box". Making behaviour comprehensible is an important step on the way towards intelligent systems: Once retrieval systems become capable of understanding a query, *more* interaction with the user will be necessary to discuss what is searched for, not less. Naturally, with the transition from the present search for features to a search for objects or scenes the level of the system feedback has to rise equally from simple visualisation to the symbolic level. Consequently, user interaction will be a highly relevant future research topic.

# 7 Acknowledgement

I would like to thank Helge Ritter, head of the Neuroinformatics Department at Bielefeld University, for the long term support of my work. Also I wish to thank the reviewers for their detailed and helpful comments.

This work was funded by the Deutsche Forschungsgemeinschaft (DFG) within the collaborative research project SFB 360 "Situated Artificial Communicators".

# References

[1] C. Faloutsos, R. Barber, M. Flickner, J. Hafner, W. Niblack, D. Petkovic, and W. Equitz. Efficient and Effective Querying by Image Content. *J. Intell. Inf. Sys.*, 3(3):231–262, 1994.

[2] J. R. Smith and S.-F. Chang. VisualSEEK: A Fully Automated Content-Based Image Query System. In *Proc. ACM Multimedia 96*, pages 87–98, 1996.

[3] A. Pentland, R. W. Picard, and S. Sclaroff. Photobook: Content-Based Manipulation for Image Databases. *Int'l J. of Computer Vision*, 18:233–254, 1996.

[4] A. Gupta and R. Jain. Visual Information Retrieval. *Commun. ACM*, 40(5):70–79, 1997.

[5] C. Carson, M. Thomas, S. Belongie, J. M. Hellerstein, and J. Malik. Blobworld: A System for Region-Based Image Indexing and Retrieval. In *Proc. Visual Information Systems*, pages 509–516, 1999.

[6] T. Gevers and A. W. M. Smeulders. PicToSeek: Combining Color and Shape Invariant Features for Image Retrieval. *IEEE Trans. on Image Processing*, 9(1):102–119, 2000.

[7] J. Z. Wang, J. Li, and G. Wiederhold. SIMPLIcity: Semantics-Sensitive Integrated Matching for Picture LIbraries. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 23(9):947–963, 2001.

[8] I. Jolliffe. *Principal Component Analysis*. Springer Verlag, New York, 1986.

[9] M. Turk and A. Pentland. Eigenfaces for Recognition. *J. Cognitive Neuroscience*, 3:71–86, 1991.

[10] N. Kambhatla and T. K. Leen. Fast Non-Linear Dimension Reduction. In J. Cowan, G. Tesauro, and J. Alspector, editors, *Advances in Neural Information Processing Systems 1993*, volume 6, pages 152–159. Morgan Kaufmann Publishers, 1994.

[11] C. Bregler and S. M. Omohundro. Surface Learning with Applications to Lipreading. In J. Cowan, G. Tesauro, and J. Alspector, editors, *Advances in Neural Information Processing Systems 1993*, volume 6, pages 43–50. Morgan-Kaufmann Publishers, 1994.

[12] N. Kambhatla and T. K. Leen. Dimension reduction by local principal component analysis. *Neural Computation*, 9(7):1493–1516, 1997.

[13] C. Bregler and S. M. Omohundro. Nonlinear Image Interpolation Using Manifold Learning. In G. Tesauro, D. Touretzky, and T. Leen, editors, *Advances in Neural Information Processing Systems 1994*, volume 7. MIT Press, 1995.

[14] R. D. Dony and S. Haykin. Optimally Adaptive Transform Coding. *IEEE Trans. on Image Processing*, 4(10):1358–1370, 1995.

[15] G. E. Hinton, M. Revow, and P. Dayan. Recognizing Handwritten Digits Using Mixtures of Linear Models. In G. Tesauro, D. Touretzky, and T. Leen, editors, *Advances in Neural Information Processing Systems 1994*, volume 7, pages 1015–1022. MIT Press, 1995.

[16] T. Hastie, P. Simard, and E. Sackinger. Learning Prototype Models For Tangent Distance. In G. Tesauro, D. Touretzky, and T. Leen, editors, *Advances in Neural Information Processing Systems 1994*, volume 7, pages 999–1006. MIT Press, 1995.

[17] G. E. Hinton, P. Dayan, and M. Revow. Modelling the Manifolds of Images of Handwritten Digits. *IEEE Trans. on Neural Networks*, 8(1):65–74, 1997.

[18] B. Moghaddam and A. Pentland. Probabilistic Visual Learning for Object Representation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 19(7):696–710, 1997.

[19] M. E. Tipping and C. M. Bishop. Mixtures of probabilistic principal component analyzers. *Neural Computation*, 11(2):443–482, 1999.

[20] H. Murase and S. K. Nayar. Visual Learning and Recognition of 3-D Objects from Appearance. *Int'l J. of Computer Vision*, 14:5–24, 1995.

[21] S. Nayar, H. Murase, and S. Nene. General Learning Algorithm for Robot Vision. In *Neural & Stochastic Methods in Image & Signal Processing*, volume 2304. SPIE, July 1994.

[22] G. Heidemann and H. Ritter. Combining Multiple Neural Nets for Visual Feature Selection and Classification. In *ICANN 99, Ninth Int'l Conf. on Artificial Neural Networks*, pages 365–370. IEE, London, 1999.

[23] G. Heidemann, D. Lücke, and H. Ritter. A System for Various Visual Classification Tasks Based on Neural Networks. In A. Sanfeliu et al., editor, *Proc. 15th Int'l Conf. on Pattern Recognition ICPR 2000, Barcelona*, volume I, pages 9–12. IEEE-CS, 2000.

[24] G. Heidemann and H. Ritter. Combining Gestural and Contact Information for Visual Guidance of Multi-Finger Grasps. In M. Verleysen, editor, *Proc. ESANN 02*, pages 301–306, Bruges, Belgium, 2002. d-side publications.

[25] G. Heidemann and H. Ritter. Visual Checking of Grasping Positions of a Three-Fingered Robot Hand. In G. Dorffner, H. Bischof, and K. Hornik, editors, *Proc. ICANN 2001*, pages 891–898. Springer, 2001.

[26] M. Swain and D. Ballard. Color Indexing. *Int'l J. of Computer Vision*, 7(1):11–32, 1991.

[27] B. V. Funt and G. D. Finlayson. Color Constant Color Indexing. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 17(5):522–529, 1995.

[28] J. R. Smith and S.-F. Chang. Tools and Techniques for Color Image Retrieval. In *Storage and Retrieval for Image and Video Databases (SPIE)*, pages 426–437, 1996.

[29] M. A. Stricker and M. Orengo. Similarity of Color Images. In *Proc. SPIE Storage and Retrieval for Image and Video Databases*, pages 381–392, 1995.

[30] A. Mojsilovic, J. Kovacevic, J. Hu, R. J. Safranek, and S. K. Ganapathy. Matching and Retrieval Based on the Vocabulary and Grammar of Color Patterns. *IEEE Trans. on Image Processing*, 9(1):38–54, 2000.

[31] M. Stricker and A. Dimai. Spectral covariance and fuzzy regions for image indexing. *Machine Vision and Applications*, 10(2):66–73, 1997.

[32] S. X. Zhou, Y. Rui, and T. S. Huang. Water-filling algorithm: A novel way for image feature extraction based on edge maps. In *Proc. IEEE Int'l Conf. on Image Processing*, Japan, 1999.

[33] A. del Bimbo and P. Pala. Visual image retrieval by elastic matching of user sketches. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 19(2):121–132, 1997.

[34] C. Schmid and R. Mohr. Local Grayvalue Invariants for Image Retrieval. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 19(5):530–535, 1997.

[35] T. Tuytelaars and L. van Gool. Content-Based Image Retrieval Based on Local Affinely Invariant Regions. In *3rd Int'l Conf. on Visual Information Systems, Visual'99*, pages 493–500, Amsterdam, The Netherlands, 1999.

[36] B. M. Mehtre, M. S. Kankanhalli, and W. F. Lee. Shape Measures for Content Based Image Retrieval: A Comparison. *Information Processing Management*, 33(3):319–337, 1997.

[37] A. W. M. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain. Content-Based Image Retrieval at the End of the Early Years. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(12):1349–1380, 2000.

[38] T. Gevers and A. W. M. Smeulders. Content-based image retrieval by viewpoint-invariant color indexing. *Image and Vision Computing*, 17(7):475–488, 1999.

[39] Q. Tian, N. Sebe, M. S. Lew, E. Loupias, and T. S. Huang. Image Retrieval Using Wavelet-Based Salient Points. *J. of Electronic Imaging*, 10(4):835–849, 2001.

[40] S. Mallat. A Theory for Multiresolution Signal Decomposition: The Wavelet Representation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 11(7):674–693, 1989.

[41] C. Carson, S. Belongie, H. Greenspan, and J. Malik. Region-Based Image Querying. In *Proc. Int'l Workshop on Content-Based Access of Image and Video Libraries*, pages 42–49, San Juan, Puerto Rico, 1997.

[42] R. W. Picard and T. P. Minka. Vision Texture for Annotation. *Multimedia Systems*, 3:3–14, 1995.

[43] A. Hiroike, Y. Musha, A. Sugimoto, and Y. Mori. Visualization of Information Spaces to Retrieve and Browse Image Data. In *Proc. Visual '99: Information and Information Systems*, pages 155–162, 1999.

[44] J. Tatemura. Browsing Images Based on Social and Content Similarity. In *Proc. IEEE Int'l Conf. on Multimedia and Expo(III)*, pages 1567–1570, 2000.

[45] D. Reisfeld, H. Wolfson, and Y. Yeshurun. Context-Free Attentional Operators: The Generalized Symmetry Transform. *Int'l J. of Computer Vision*, 14:119–130, 1995.

[46] L. Kaufman and W. Richards. Spontaneous Fixation Tendencies for Visual Forms. *Perception and Psychophysics*, 5(2):85–88, 1969.

[47] V. Bruce and M. Morgan. Violation of Symmetry and Repetition in Visual Pathways. *Perception*, 4:239–249, 1975.

[48] P. J. Locher and C. F. Nodine. Symmetry Catches the Eye. In A. Levy-Schoen and J. K. O'Reagan, editors, *Eye Movements: From Physiology to Cognition*, pages 353–361. Elsevier Science Publishers B. V. (North Holland), 1987.

[49] C. M. Privitera and L. W. Stark. Algorithms for Defining Visual Regions-of-Interest: Comparison with Eye Fixations. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(9):970–982, 2000.

[50] G. Heidemann. Focus-of-Attention from Local Color Symmetries. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 26(7):817–830, 2004.

[51] T. W. Nattkemper. Untersuchung und Erweiterung eines Ansatzes zur modellfreien Aufmerksamkeitssteuerung durch lokale Symmetrien in einem Computer Vision System. Master's thesis, Bielefeld Univ., Technische Fakultät, 1997.

[52] C. Harris and M. Stephens. A Combined Corner and Edge Detector. In *Proc. 4th Alvey Vision Conf.*, pages 147–151, 1988.

[53] W. Förstner. A Framework for Low Level Feature Extraction. In *Proc. 3rd European Conf. on Computer Vision*, pages 383–394, Stockholm, Sweden, 1994.

[54] C. Tomasi and T. Kanade. Detection and Tracking of Point Features. Technical Report CMU-CS-91-132, Carnegie Mellon University, Pittsburgh, 1991.

[55] H. P. Moravec. Towards Automatic Visual Obstacle Avoidance. In *Proc. 5th Int'l Joint Conf. on Artificial Intelligence*, pages 584–587, Cambridge, Massachusetts, USA, 1977.

[56] G. Medioni and Y. Yasumoto. Corner Detection and Curve Representation Using Cubic B-Splines. *Computer Vision, Graphics and Image Processing*, 39:267–278, 1987.

[57] R. Horaud, F. Veillon, and T. Skordas. Finding Geometric and Relational Structures in an Image. In *Proc. 1st European Conf. on Computer Vision*, pages 374–384, Antibes, France, 1990.

[58] E. Shilat, M. Werman, and Y. Gdalyahu. Ridge's Corner Detection and Correspondance. In *Proc. Conf. on Computer Vision and Pattern Recognition*, pages 976–981, Puerto Rico, USA, 1997.

[59] C. Schmid, R. Mohr, and C. Bauckhage. Evaluation of Interest Point Detectors. *Int'l J. of Computer Vision*, 37(2):151–172, 2000.

[60] T. Gevers. Color Based Image Retrieval. In M. Lew, editor, *Multimedia Search*. Springer Verlag, 2001.

[61] J. Buhmann and H. Kühnel. Vector quantization with complexity costs. *IEEE Trans. on Information Theory*, 39(4):1133–1145, 1993.

[62] T. Kohonen. Learning Vector Quantization. In M. A. Arbib, editor, *The Handbook of Brain Theory and Neural Networks*, pages 537–540. MIT Press, 1995.

[63] S. Grossberg. Competitive learning: From interactive activation to adaptive resonance. *Cognitive Sci.*, 11:23–63, 1987.

[64] T. Martinetz, S. G. Berkovich, and K. Schulten. "Neural-Gas" Network for Vector Quantization and its Application to Time-Series Prediction. *IEEE Trans. on Neural Networks*, 4(4):558–569, 1993.

[65] J.-H. Wang and C.-P. Hsiao. Representation-burden conservation network applied to learning vq. *Neural Processing Letters*, 5(3):209–217, 1997.

[66] S. C. Ahalt, A. K. Krisnamurthy, P. Chen, and D. E. Melton. Competitive learning algorithms for vector quantization. *Neural Networks*, 3:277–290, 1990.

[67] A. S. Galanopoulos, R. L. Moses, and S. C. Ahalt. Diffusion Approximation of Frequency Sensitive Competitive Learning. *IEEE Trans. on Neural Networks*, 8(5):1026–1030, 1997.

[68] G. Heidemann and H. Ritter. Efficient Vector Quantization Using the WTA-rule with Activity Equalization. *Neural Processing Letters*, 13(1):17–30, 2001.

[69] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proc. 5th Berkeley Symp. Math. Stat. Probab.*, volume 1, pages 281–297, 1965.

[70] K. Rose, F. Gurewitz, and G. Fox. Statistical mechanics and phase transitions in clustering. *Phys. Rev. Lett.*, 65(8):945–948, 1990.

[71] T. D. Sanger. Optimal Unsupervised Learning in a Single-Layer Linear Feedforward Neural Network. *Neural Networks*, 2:459–473, 1989.

[72] S. Haykin. *Neural Networks*. Prentice Hall, New Jersey, 1999.

[73] A. S. Householder. *The Theory of Matrices in Numerical Analysis*. Dover Publications, New York, 1964.

[74] S. A. Nene, S. K. Nayar, and H. Murase. Columbia Object Image Library: COIL-100. Technical Report CUCS-006-96, Dept. Computer Science, Columbia Univ., 1996.