Image Modeling based on Kernel Principal Component Analysis

Kwang In Kim, Matthias O. Franz and Bernhard Schölkopf

Max-Planck-Institut für Biologische Kybernetik, Tübingen, Germany.
E-mail:kimki;mof;bs@tuebingen.pg.de

Abstract - This article presents a method for estimating a generative image model based on *Kernel Principal Component Analysis* (KPCA). In contrast to other patch-based modeling approaches such as PCA, ICA or sparse coding, KPCA is capable of capturing nonlinear interactions between the basis elements of the image. The original form of KPCA, however, can be only applied to strongly restricted image classes due to the limited number of training examples that can be processed. We therefore propose a new iterative method for performing KPCA, the *Kernel Hebbian Algorithm*. By kernelizing the Generalized Hebbian Algorithm, one can iteratively estimate the Kernel Principal Components with only linear order memory complexity. We demonstrate the generalization capabilities of the resulting image model in single-frame super-resolution and denoising applications.

Index terms–Principal component analysis, Kernel methods, Image models, Image enhancement.

1 Introduction

Prior knowledge about the statistics of specific image classes affords numerous applications in image processing such as super-resolution [1, 2], denoising [3, 4, 5], segmentation [6], or compression [7]. The prior can be coded either *implicitly* by directly learning the mapping between input and desired output (as in [1, 2]), or *explicitly* by finding a suitable image model. In image modeling, we can roughly distinguish between approaches that try to estimate aspects of the underlying probability distribution using a fixed set of basis elements such as wavelets [7, 4], projected profiles of objects [8] or geometrical primitives [9, 10], and approaches that try to find basis sets with certain optimality properties ranging from Principal Component Analysis (PCA) [11], Independent Component Analysis (ICA) [12, 13] to sparse coding [14].

Interestingly, all of the latter approaches model images as linear combinations of transparent basis images. Many researchers have pointed out, however, that this does not reflect the generation process of natural images (e.g., [9]). Here, one of the main contributing factors is *occlusion* which is highly nonlinear. This suggests the use of techniques that can cope with nonlinear combinations of basis images. One of these techniques is *Kernel Principal Component Analysis* (KPCA) [15]. In contrast to linear PCA, KPCA is capable of capturing part of the higher-order statistics which are particularly important for encoding image structure [16].

Capturing these higher-order statistics can require a large number of training examples, particularly for larger image sizes and complex image classes such as patches taken from natural images. This causes problems for KPCA, since KPCA requires to store and manipulate the *kernel matrix* the size of which is the square of the number of examples. To overcome this problem, a new iterative algorithm for KPCA, the *Kernel Hebbian Algorithm* (KHA) is introduced. It is based on the generalized Hebbian algorithm (GHA) which was introduced as an online algorithm for linear PCA [17, 11].

The resulting algorithm estimates the kernel principal components with linear order memory complexity, making it applicable to large problems.

In this article, we focus on the computational aspects of estimating a KPCA image model, and on its application in single-frame super-resolution and denoising. The remainder of this paper is organized as follows: Section 2 briefly introduces PCA, GHA, and KPCA. Section 3 formulates the KHA. Experimental results are presented in Section 4 and conclusions are drawn in Section 5.

2 Principal component models

2.1 Linear principal component analysis and the Generalized Hebbian Algorithm

Given a set of l centered observations $\mathbf{x}_k = \mathbb{R}^N$, $k = 1, \ldots, l$, and $\sum_{k=1}^{l} \mathbf{x}_k = 0$, PCA diagonalizes the covariance matrix[1]

$$\overline{C} = \frac{1}{l} \sum_{j=1}^{l} \mathbf{x}_j \mathbf{x}_j^\top . \tag{1}$$

For lower-dimensional data, this is readily performed by solving the eigenvalue equation $\lambda \mathbf{v} = \overline{C}\mathbf{v}$ for eigenvalues $\lambda \geq 0$ and eigenvectors $\mathbf{v}_i \in \mathbb{R}^N \setminus 0$ (cf., e.g. [18]). The resulting set of mutually orthogonal eigenvectors defines a new basis along the directions of maximum variance in the data. The pairwise decorrelated expansion coefficients in this new basis are called the *principal components* of the dataset. From the point of view of image modeling, the PCA basis has the interesting property that, among all basis expansions, it minimizes the reconstruction error when the expansion is truncated to a smaller number of basis vectors. Thus, a class of high-dimensional images can be described by a low-dimensional model containing only a few principal components.

Computationally, it can be advantageous to solve the eigenvalue problem by iterative methods which do not need to compute and store $\overline{C}$ directly. This is particulary useful when the size of $\overline{C}$ is large such that the memory complexity becomes prohibitive. Among the existing iterative methods for PCA, the Generalized Hebbian Algorithm (GHA) is of particular interest, since it does not only provide a memory-efficient implementation but also has the inherent capability to adapt to time-varying distributions. Let us define a matrix $\mathbf{W}(t) = (\mathbf{w}_1(t)^\top, \ldots, \mathbf{w}_r(t)^\top)^\top$, where r is the number of eigenvectors considered and $\mathbf{w}_i(t) \in \mathbb{R}^N$. Given a random initialization of $\mathbf{W}(0)$, the GHA applies the following recursive rule

$$\mathbf{W}(t+1) = \mathbf{W}(t) + \eta(t)(\mathbf{y}(t)\mathbf{x}(t)^\top - \mathrm{LT}[\mathbf{y}(t)\mathbf{y}(t)^\top]\mathbf{W}(t)), \tag{2}$$

where $\mathbf{x}(t)$ is a randomly selected pattern from the l input examples, presented at time t, $\mathbf{y}(t) = \mathbf{W}(t)\mathbf{x}(t)$. $\mathrm{LT}[\cdot]$ sets all elements above the diagonal of its matrix argument to zero, thereby making it lower triangular. It was shown in [17] for $r = 1$ and in [11] for $r > 1$ that $\mathbf{W}(t) \rightarrow \mathbf{V}_r = (\mathbf{v}_1(t)^\top, \ldots, \mathbf{v}_r(t)^\top)^\top$ as $t \rightarrow \infty$.[2] For a detailed discussion of the GHA, readers are referred to [11].

[1] More precisely, the covariance matrix is defined as the expectation $E[\mathbf{x}\mathbf{x}^\top]$; $\overline{C}$ is an estimate based on a finite set of examples.

[2] Originally it has been shown that $\mathbf{w}_i$ converges to the i-th eigenvector of $E[\mathbf{x}\mathbf{x}^\top]$, given an infinite sequence of examples. By replacing each $\mathbf{x}(t)$ with a random selection $\mathbf{x}_i$ from a finite training set, we obtain the above statement.

The GHA has been applied in several studies to compute the principal components of natural images [11, 19, 14]. PCA image models have been used, for instance, for image coding and texture segmentation [11], and for explaining psychophysically derived orientation tuning curves [20].

2.2 Kernel principal component analysis

Linear PCA is an appropriate model for data that are generated by a Gaussian distribution, or data that are best described by second-order correlations. In fact, PCA is based only on second-order correlations (cf. Eq. 1) such that no higher-order statistics can influence its result. It is well known, however, that the distribution of natural images is highly non-gaussian, and that all the "interesting" structures in images such as edges or corners cannot be described by second-order correlations [16]. This motivates the use of a nonlinear analysis technique that can capture higher-order dependencies in the data.

As a nonlinear extension of PCA, KPCA computes the principal components (PCs) in a possibly high-dimensional *Reproducing Kernel Hilbert Space* (RKHS) F which is related to the input space by a nonlinear map $\Phi : \mathbb{R}^N \rightarrow F$ [21]. An important property of a RKHS is that the inner product of two points mapped by Φ can be evaluated using *kernel functions*

$$k(\mathbf{x}, \mathbf{y}) = \Phi(\mathbf{x}) \cdot \Phi(\mathbf{y}), \tag{3}$$

which allows us to compute the value of the inner product without having to carry out the map Φ explicitly. Since PCA can be formulated in terms of inner products, we can compute it also implicitly in a RKHS. Assuming that the data are centered in F (i.e., $\sum_{k=1}^{l} \Phi(\mathbf{x}_k) = 0$)[3] the covariance matrix takes the form

$$C = \frac{1}{l} \mathbf{\Phi}^\top \mathbf{\Phi}, \tag{4}$$

where $\mathbf{\Phi} = \left(\Phi(\mathbf{x}_1)^\top, \ldots, \Phi(\mathbf{x}_l)^\top\right)^\top$. We now have to find the eigenvalues $\lambda \geq 0$ and eigenvectors $\mathbf{v} \in F \setminus 0$ satisfying

$$\lambda \mathbf{v} = C\mathbf{v}. \tag{5}$$

For $\lambda \neq 0$, all solutions $\mathbf{v}$ lie within the span of $\{\Phi(\mathbf{x}_1), \ldots, \Phi(\mathbf{x}_l)\}$ [15], and we may consider the following equivalent problem

$$\lambda \mathbf{\Phi} \mathbf{v} = \mathbf{\Phi} C \mathbf{v}, \tag{6}$$

and represent $\mathbf{v}$ in terms of an l-dimensional vector $\mathbf{q}$ as $\mathbf{v} = \mathbf{\Phi}^\top \mathbf{q}$. Combining this with (4) and (6) and defining an $l \times l$ kernel matrix $\mathbf{K}$ by $\mathbf{K} = \mathbf{\Phi}\mathbf{\Phi}^\top$ leads to $l\lambda \mathbf{K}\mathbf{q} = \mathbf{K}^2\mathbf{q}$. The solution can be obtained by solving the *kernel eigenvalue problem* [15]

$$l\lambda \mathbf{q} = \mathbf{K}\mathbf{q}. \tag{7}$$

The resulting kernel principal components are linear combinations of inner products of the data points, i.e., there is no need to compute Φ explicitly since everything can be expressed in terms of kernel functions. In contrast to PCA, ICA or sparse coding, kernel principal components consist of nonlinear interactions between the data points.

[3]The centering issue will be dealt with later.

In terms of image modeling, this means that images are modeled as nonlinear combinations of the input images by using the kernel function.

Note that the modeling capability of PCA is retained by KPCA: It allows for a truncated expansion in only a few kernel principal components. However, the truncated expansion minimizes the reconstruction error in the RKHS, not in the input space as in linear PCA. This seems like an odd optimization principle, but it is not clear from the outset whether the Euclidian error norm is a better error measure for such complex objects as images. In fact, numerous applications have shown that KPCA often leads to better models than PCA.

3 KPCA Image Model

3.1 Kernel Hebbian Algorithm

The size of the kernel matrix scales with the square of the number of examples. Thus, it becomes computationally infeasible to directly solve the kernel eigenvalue problem for a large number of examples. As noted in the introduction, a similar problem occurs with linear PCA when the covariance matrix becomes large. This motivated the introduction of the GHA which does not require the storage and inversion of the covariance matrix. Here, we propose a similar approach by reformulating the GHA in a RKHS to obtain a memory-efficient approximation of KPCA.

The GHA update rule of Eq. (2) is represented in the RKHS F as

$$\mathbf{W}(t+1) = \mathbf{W}(t) + \eta(t)(\mathbf{y}(t)\Phi(\mathbf{x}(t))^\top - \mathrm{LT}[\mathbf{y}(t)\mathbf{y}(t)^\top]\mathbf{W}(t)), \tag{8}$$

where the rows of $\mathbf{W}(t)$ are now vectors in F and $\mathbf{y}(t) = \mathbf{W}(t)\Phi(\mathbf{x}(t))$. $\Phi(\mathbf{x}(t))$ is a pattern presented at time t which is randomly selected from the mapped data points $\{\Phi(\mathbf{x}_1), \dots, \Phi(\mathbf{x}_l)\}$. For notational convenience we assume that there is a function $J(t)$ which maps t to $i \in \{1, \dots, l\}$ ensuring $\Phi(\mathbf{x}(J(t))) = \Phi(\mathbf{x}_i)$ and denote $\Phi(\mathbf{x}(J(t)))$ simply by $\Phi(\mathbf{x}(t))$. From the direct KPCA solution, it is known that $\mathbf{w}(t)$ can be expanded in the mapped data points $\Phi(\mathbf{x}_i)$. This restricts the search space to linear combinations of the $\Phi(\mathbf{x}_i)$ such that $\mathbf{W}(t)$ can be expressed as

$$\mathbf{W}(t) = \mathbf{A}(t)\mathbf{\Phi} \tag{9}$$

with an $r \times l$ matrix $\mathbf{A}(t) = (\mathbf{a}_1(t)^\top, \dots, \mathbf{a}_r(t)^\top)^\top$ of expansion coefficients. The ith row $\mathbf{a}_i = (a_{i1}, \dots, a_{il})$ of $\mathbf{A}(t)$ corresponds to the expansion coefficients of the ith eigenvector of $\mathbf{K}$ in the $\Phi(\mathbf{x}_i)$, i.e., $\mathbf{w}_i(t) = \mathbf{\Phi}^\top \mathbf{a}_i(t)$. Using this representation, the update rule becomes

$$\mathbf{A}(t+1)\mathbf{\Phi} = \mathbf{A}(t)\mathbf{\Phi} + \eta(t)\left(\mathbf{y}(t)\Phi(\mathbf{x}(t))^\top - \mathrm{LT}[\mathbf{y}(t)\mathbf{y}(t)^\top]\mathbf{A}(t)\mathbf{\Phi}\right). \tag{10}$$

The mapped data points $\Phi(\mathbf{x}(t))$ can be represented as $\Phi(\mathbf{x}(t)) = \mathbf{\Phi}^\top \mathbf{b}(t)$ with a canonical unit vector $\mathbf{b}(t) = (0, \dots, 1, \dots, 0)^\top$ in $\mathbb{R}^l$ (only the $J(t)$-th element is 1). Using this notation, the update rule can be written solely in terms of the expansion coefficients as

$$\mathbf{A}(t+1) = \mathbf{A}(t) + \eta(t)\left(\mathbf{y}(t)\mathbf{b}(t)^\top - \mathrm{LT}[\mathbf{y}(t)\mathbf{y}(t)^\top]\mathbf{A}(t)\right). \tag{11}$$

Representing (11) in component-wise form gives

$$a_{ij}(t+1) = \begin{cases} a_{ij}(t) + \eta y_i(t) - \eta y_i(t) \sum_{k=1}^{i} a_{kj}(t) y_k(t) & \text{if} \quad J(t) = j \\ a_{ij}(t) - \eta y_i(t) \sum_{k=1}^{i} a_{kj}(t) y_k(t) & \text{otherwise,} \end{cases} \tag{12}$$

where

$$y_i(t) = \sum_{k=1}^{l} a_{ik}(t)\Phi(\mathbf{x}_k) \cdot \Phi(\mathbf{x}(t)) = \sum_{k=1}^{l} a_{ik}(t)k(\mathbf{x}_k, \mathbf{x}(t)). \qquad (13)$$

This does not require $\Phi(\mathbf{x})$ in explicit form, thus providing a practical implementation of the GHA in F.

During the derivation of (11), it was assumed that the data are centered in F which is not true in general unless explicit centering is performed. Centering can be done by subtracting the mean of the data from each pattern. Then each pattern $\Phi(\mathbf{x}(t))$ is replaced by $\widetilde{\Phi}(\mathbf{x}(t)) := \Phi(\mathbf{x}(t)) - \overline{\Phi}(\mathbf{x})$, where $\overline{\Phi}(\mathbf{x})$ is the sample mean $\overline{\Phi}(\mathbf{x}) = \frac{1}{l}\sum_{k=1}^{l}\Phi(\mathbf{x}_k)$. The centered algorithm remains the same as in (12) except that Eq. (13) has to be replaced by the more complicated expression

$$y_i(t) = \sum_{k=1}^{l} a_{ik}(t)(k(\mathbf{x}(t), \mathbf{x}_k) - \bar{k}(\mathbf{x}_k)) - \overline{a}_i(t)\sum_{k=1}^{l}(k(\mathbf{x}(t), \mathbf{x}_k) - \bar{k}(\mathbf{x}_k)). \qquad (14)$$

with $\bar{k}(\mathbf{x}_k) = \frac{1}{l}\sum_{m=1}^{l} k(\mathbf{x}_m, \mathbf{x}_k)$ and $\overline{a}_i(t) = \frac{1}{l}\sum_{m=1}^{l} a_{im}(t)$.[4] This is directly applicable in a batch setting (i.e., the patterns are fixed and known in advance); in an online setting, one should instead use a sliding mean, in order to be able to adapt to changes in the distribution. For the details of the online algorithm, readers are referred to [22]. It should be noted that not only in training but also in testing, each pattern should be centered using the training mean.

The time and memory complexity for each iteration of the KHA is $\mathbf{O}(r \times l \times N)$ and $\mathbf{O}(r \times l + l \times N)$, respectively, where r, l, and N are the number of principal components to be computed, the number of examples, and the dimensionality of input space, respectively.[5] This rather high time complexity can be lowered by precomputing and storing the whole or part of the kernel matrix. When we store the entire kernel matrix, as KPCA does, the time complexity reduces to $\mathbf{O}(r \times l)$.

Now we state the convergence properties of the KHA (Eq. 8) as a theorem:

Theorem 1 *For a finite set of centered data (presented infinitely often) and* $\mathbf{A}$ *initially in general position,*[6] *(8) (and equivalently (11)) will converge with probability 1,*[7] *and the rows of* $\mathbf{W}$ *will approach the first r normalized eigenvectors of the correlation matrix* C *in the RKHS, ordered by decreasing eigenvalue.*

The proof of theorem 1 is straightforward if we note that for a finite set of data $\{\mathbf{x}_1, \ldots, \mathbf{x}_l\}$, we can induce from a given kernel k, an *kernel PCA map* [21])

$$\Phi_l : \mathbf{x} \rightarrow \mathbf{K}^{-\frac{1}{2}}(k(\mathbf{x}, \mathbf{x}_1), \ldots, k(\mathbf{x}, \mathbf{x}_l))$$

satisfying

$$\Phi_l(\mathbf{x}_i) \cdot \Phi_l(\mathbf{x}_j) = k(\mathbf{x}_i, \mathbf{x}_j).$$

By applying the GHA in the space spanned by the kernel PCA map, (i.e., replacing each occurrence of $\Phi(\mathbf{x})$ with $\Phi_l(\mathbf{x})$ in Eq. 8, and noting that this time, $\mathbf{W}$ lies in $\mathbb{R}^l$ rather than in F), we obtain an algorithm in $\mathbb{R}^l$ which is exactly equivalent to the KHA in F. The convergence of the KHA then follows from the convergence of the GHA in $\mathbb{R}^l$. It should be noted that, in practice, this approach cannot be taken to construct an iterative algorithm since it involves the computation of $\mathbf{K}^{-\frac{1}{2}}$.

[4] Matlab example implementation can be downloaded at http://www.kyb.tuebingen.mpg.de/prjs/comp_vision_robotics/nat_image/kha/kha.htm.

[5] $\bar{k}(\mathbf{x}_k)$ and $\overline{a}_i(t)$ in (14) for each $k, i = 1, \ldots, l$ are calculated only once at the beginning of the whole procedure and at the beginning of each iteration, respectively.

[6] i.e., $\mathbf{A}$ is neither the zero vector nor orthogonal to the eigenvectors.

[7] Assuming that the input data is not always orthogonal to the initialization of $\mathbf{A}$.

3.2 Applications

As a generic image model, KPCA can be applied to a broad range of applications. We will demonstrate the potential of the KPCA model in two image reconstruction tasks, single-frame super-resolution and denoising.

Single-frame super-resolution refers to the task of constructing a high-resolution enlargement of a *single*[8] low-resolution, pixel-based image. In contrast to the usual interpolation and sharpening (e.g. [24]), new high-resolution details are added to the reconstruction. This can only be done by relying on prior knowledge about the image class to be processed.

In previous work, single-frame super-resolution was mainly done in a *supervised* learning setting [2, 25]: During the training phase, pairs of low-resolution patches and the corresponding high-resolution patches are collected. In the super-resolution phase, each low-resolution patch of the input image is compared to the stored low-resolution patches and the high-resolution patch corresponding to the nearest low-resolution patch is selected.

Here, we propose an alternative approach to super-resolution based on KPCA which is an *unsupervised* learning method. Instead of encoding a fixed relationship between pairs of high- and low-resolution image patches, we rely on the generic model of the high-resolution images obtained from KPCA. To reconstruct a super-resolution image from a low-resolution image which was *not* contained in the training set, we first scale up the image to the same size as the high-resolution training images, then map the image (call it $\mathbf{x}$) into the RKHS F using Φ, and project it into the KPCA subspace corresponding to a limited number of PCs to get $P\Phi(\mathbf{x})$. Via the projection P, the image is mapped to an image which is consistent with the statistics of the high-resolution training images. However, at that point the projection still lives in F which can be infinite-dimensional. We thus need to find a corresponding point in $\mathbb{R}^N$ — this is a preimage problem. To solve it, we minimize $\|P\Phi(\mathbf{x}) - \Phi(\mathbf{z})\|^2$ over $\mathbf{z} \in \mathbb{R}^N$. Note that this objective function can be computed in terms of inner products and thus in terms of the kernel (3). For the minimization, we use gradient descent [26] with starting points obtained using the method of [27].

Image denoising refers to the task of constructing a noise-free image from a noisy input image. Since image denoising is a standard problem in the image processing community, the readers are referred to [28] for a brief survey. From the point of view of a KPCA model, image denoising can be regarded as the same problem as image super-resolution: the projection method from the super-resolution task can be applied to image denoising as well. The only difference is that the scaling of input image is omitted which is not necessary for denoising. KPCA has already been applied to digit image denoising and demonstrated promising results [5]. This was possible because the class of digit images is small enough for the direct computation of KPCA. Presently, we focus on more complex image classes such as faces that require a larger number of training examples. In these cases, the direct computation of KPCA is no more feasible but requires the use of the KHA.

[8]This should not be confused with *aggregation from multiple frames* where a single high-resolution frame is extracted from a sequence of low resolution images (cf., e.g. [23]).

4 Experiments

From a machine learning point of view, an image is simply a point in an image space whose dimensionality is equal to the number of pixels in the image. If the class of images is moderately constrained (e.g. face images) or if the image size is small enough, the learning of a KPCA model poses no serious problems since the image space can be sampled densely enough. However, for rather large images with arbitrarily high complexity (e.g., natural images) the necessarily limited amount of training data leads to *overfitting*, where one obtains a model which explains the training data perfectly but fails to generalize to unknown data. In these cases, we adopt a *patch-based* approach where a large image is regarded as a composition of patches (small sub-images). Accordingly, this section describes two distinct sets of experiments according to the classes of images considered: the *single-patch* case regards a small image as a single pattern and the *multi-patch* case regards a large image as a set of small patches.

The number of iterations for the convergence of the KHA depends on the data set. The iteration finished when the squared distance between two solutions from consecutive iterations is smaller than a given threshold. It took around 40 and 120 iterations for face and natural images, respectively.

Single-patch case: Super-resolution and de-noising of face images. Here we consider a database of face images. The Yale Face Database B contains 5,760 images of 10 persons [29]. 5,000 images were used for training while 10 randomly selected images which are disjoint from the training set were used to test the method (note, however, as there are only 10 persons in the database, the same person, in different views, is likely to occur in training and test set). Since the direct computation of KPCA for this dataset is not practical on standard hardware, the KHA was utilized. In training, (60×60)-sized face images were fed into the KHA using a Gaussian kernel $k(\mathbf{x}, \mathbf{y}) = \exp(-\|\mathbf{x} - \mathbf{y}\|^2/2\sigma^2)$ with $\sigma = 1$.

For the super-resolution experiments, the test images were blurred and subsampled to a 20×20 grid and scaled up to the original scale (60×60) by turning each pixel into a 3×3 square of identical pixels, before doing the reconstruction. Fig. 1 shows reconstruction examples obtained using different numbers of components. For comparison, reconstructions obtained by linear PCA are also displayed. While the images obtained from linear PCA look like somewhat uncontrolled superpositions of different face images, the images obtained from its nonlinear counterpart (KHA) are more face-like. In spite of its less realistic results, linear PCA was slightly better than the KHA in terms of the mean squared error (average 9.20 and 8.48 for KHA and PCA, respectively for 100 PCs). This stems from the characteristics of PCA which is constructed to minimize the MSE while KHA is not concerned with the MSE in the input space. Instead, it seems to force the images to be contained in the manifold of face images. Similar observations have been reported in [30]. Interestingly, a small number of examples and a sparse sampling of this manifold can have the consequence that the KPCA (or KHA) reconstruction looks like the face of person different from the one used to generate the test image. In a sense, this means that the errors performed by KPCA are errors *along* the manifold of faces. Fig. 2 demonstrates this effect by comparing results from KPCA on 1,000 example images (corresponding to a sparse sampling of the face manifold) and KHA on 5,000 training images (denser sampling). As the examples show, some of the misreconstructions that are made by KPCA due to the lack of training examples were corrected by the KHA using a larger training set.

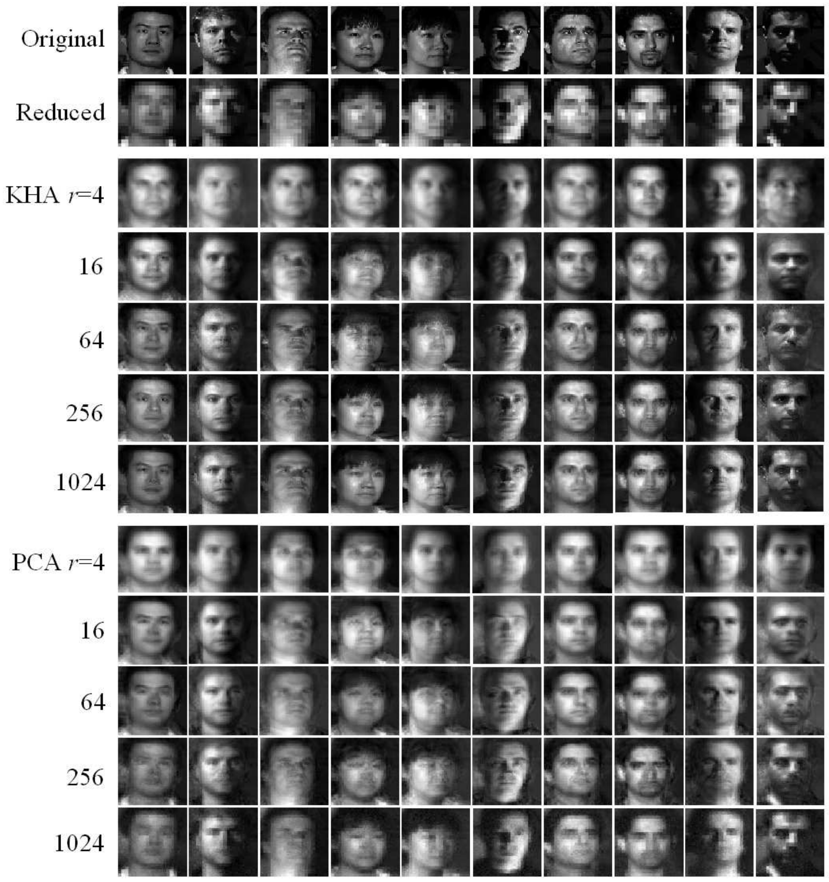

Figure 1: Face reconstruction based on PCA and KHA for varying number of principal components.

To see the effect of varying input image resolution on the reconstruction result, another set of experiments has been performed with different resolutions of 10×10, 20×20, and 40×40 as shown in Fig. 3. A graceful degradation of reconstruction performance was observed from both PCA and KPCA models as the input image resolution decreases. However, KPCA results look constantly better than that of PCA, especially when the input image is very small (10×10).

For image denoising, uniform random noise was added to the test images of Fig. 1 with a SNR around 14.13dB. We applied the KPCA model used for face image super-resolution with no additional training and no modification of the experimental setting, except for the omission of the smoothing and resizing steps. The results (Fig. 4) indicate that the proposed model is general enough to be applied to more than one specific application, but still has acceptable performance in each of them. Compared to PCA, the mean squared error of KPCA was again slightly worse. However, visual inspection shows that the KPCA solutions are far more realistic.

By moving along the principal axes in the RKHS and computing the corresponding preimages one can directly visualize what the KPCA model has learned. As an example, Fig. 5 shows the preimages along the principal axes corresponding to the three

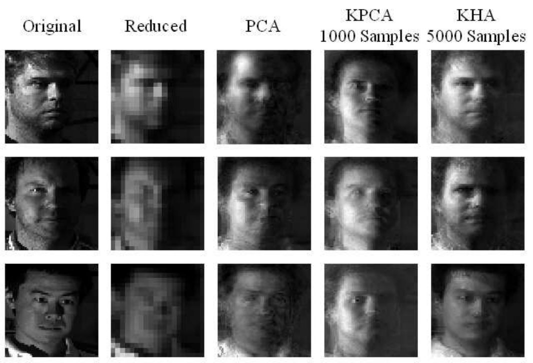

Figure 2: Face reconstruction examples obtained from KPCA and KHA trained on 1,000 and 5,000 examples, respectively. Occasional erroneous reconstruction of images indicates that KPCA requires a large amount of data to properly sample the underlying structure.

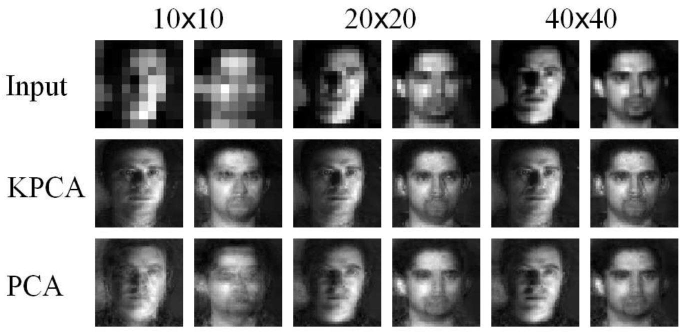

Figure 3: Face reconstruction based on PCA and KPCA with different input image resolutions.

largest eigenvalues. In contrast to linear eigenfaces (i.e., eigenvectors of face images) [31], the nonlinear eigenfaces obtained from KPCA are more face-like. This can be explained if we assume that the data have a cluster structure and that the Gaussian kernel parameter σ is small compared to the distances between the clusters, but large compared to the distances within the clusters. Then KPCA becomes similar to linear PCA performed on each cluster of similar (close in terms of the distance in input space) images: in this case, the kernel matrix is almost block diagonal, and accordingly, the eigenvectors of the kernel matrix become similar to the eigenvectors of each cluster. As a result, the eigenvectors are superpositions of similar images (cf. Eq. (6)) such that their preimages do not show the superposition artifacts usually encountered in linear eigenfaces. On the other hand, the distance structure within a block (a set of patterns close to each other) is similar to the Euclidean distance in the input space if the block size is small enough.[9]

In addition, within the projection interval defined by the standard deviation of sam-

[9]This becomes evident when the function $f(z) = e^z$ is expanded in the Taylor series about zero. When the absolute values of z approaches to zero, the high-order terms in the series tend to vanish such that f becomes linear. Thus the kernel is approximately $1 - \|\mathbf{x} - \mathbf{y}\|^2$, a conditionally positive definite kernel which for KPCA is equal to $\mathbf{x} \cdot \mathbf{y}$ [21].

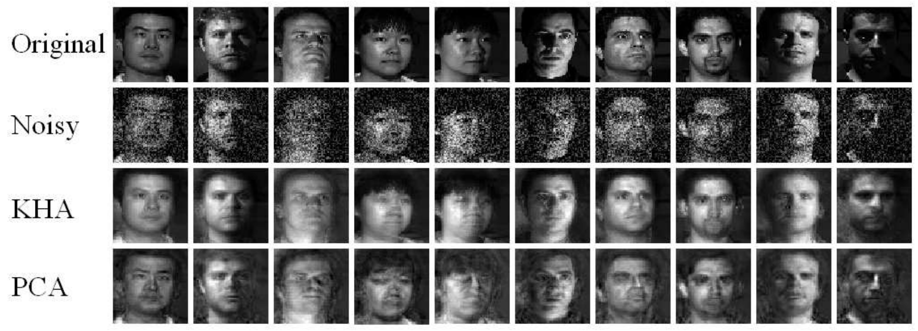

Figure 4: Face image denoising based on PCA and KPCA with 256 principal components.

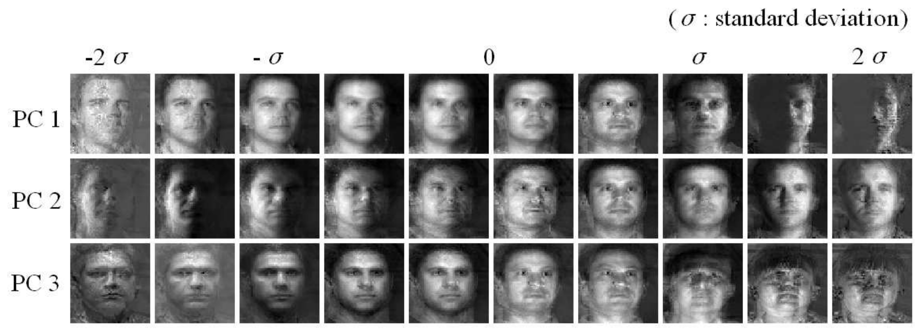

Figure 5: Preimages of the first three eigenvectors with RKHS projections varying from -2σ to 2σ (where σ is the standard deviation along each principal axis). Note that moving from $-\sigma$ to σ generates a morph between two faces.

pling points along the eigenvector (Fig. 5), the eigenfaces show a morphing behavior from one face class to another as we move along the principal axis. The corresponding trajectory is not entirely contained in the training set, but is actually *learned* from the training samples.[10] Outside this region the sampling of patterns becomes very sparse as indicated by the increasing distance to the nearest training pattern in Fig. 6. Here, we observe a rather uncontrolled superposition of face images similar to linear PCA. This supports the previously mentioned manifold interpretation: assuming that the data are sampled densely enough, the KPCA image model reconstructs the manifold defined by the image class. Note that it has recently been pointed out that for certain kernels, KPCA corresponds to several known manifold dimensionality reduction algorithms [33].

Multi-patch case: Super-resolution of natural images. For a realistic natural image super-resolution, we adopt the method of [25], where the large image is decomposed into its low-frequency components and a set of small patches containing the local high-frequency information. Whereas Freeman et. al. [25] use a nearest neighbor classifier to select appropriate high-frequency patches in the super-resolution phase,

[10]Similar observations have been reported in [32] where KPCA was used to model 3D objects from 2D views.

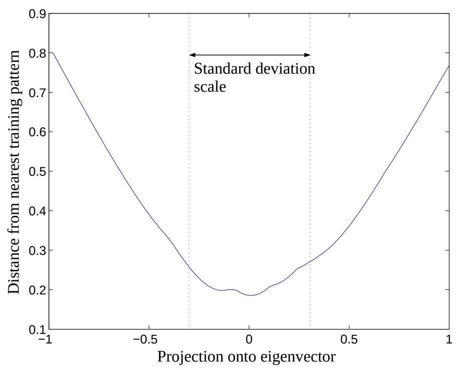

Figure 6: RKHS distance to the nearest training pattern when moving along the third principal axis.

we replace this classifier by the projection step described above. During the training stage, images are high-pass filtered and a set of image patches are collected from the resulting high-frequency images. These image patches are contrast-normalized [25] and then fed into the KHA. In the super-resolution phase, the input image is rescaled to the original resolution using bicubic interpolation and band-pass filtered to remove the low-frequency components. Then, the resulting high-frequency component image is divided into a set of small image patches each of which is reconstructed in the same way as in single patch super-resolution. The resulting image contains only high-frequency components which are then superimposed on the bicubic interpolation to give the final reconstruction.

The KHA was trained on a set of 10,000 (12×12)-sized image patches obtained from the images in Fig. 7. As above, the σ parameter was set to a rather small value (1) to capture the nonlinear structure of the images. The reconstruction of the high-frequency image is then obtained based on the first 200 KPCs. When applied to non-overlapping patches, the resulting image as a whole shows a block structure since each patch is reconstructed independently of its neighborhood. To reduce this effect, the patches are chosen to slightly overlap into their neighbors such that the overlapping regions can be averaged.

A (396×528)-size image not contained in the training set was used for testing. The (198×264)-sized low-resolution image was obtained by blurring and subsampling. Fig. 8 shows the super-resolution result. The final reconstruction was post-processed using high-boost filtering [28] to enhance the edges that become slightly blurred since only the first 200 KPCAs are used in the reconstruction. It should be noted that the original KHA reconstruction of the high-frequency components still contains blocking artifacts even with the use of overlapping patches. This, however, does not severely degrade the final result since the overall structure is contained in the low frequency input image and the KHA reconstruction only adds the missing high-frequency information. Regarding more advanced techniques for the removal of blocking artifacts, readers are referred to [25] where the spatial relationship between patches is modeled based on Markov random fields.

Fig. 9 shows more super-resolution results. The low resolution image is obtained in the same way as in Fig. 8. For comparison, bicubic interpolation and the nearest neighbor technique also have been applied. Again, for all the methods the final recon-

Figure 7: Training images of size 396×528. The training patterns are obtained by sampling 2,500 points at random from each image.

structions are high-boost filtered. In comparison to image stretching (Fig. 9.b), bicubic interpolation (Fig. 9.c) produces far better results. However simple edge enhancement without any priori knowledge failed to completely remove the blurring effect. The two learning-based methods show a better capability in recovering the complex local structure, especially in the leaves.

To get a better understanding of the generalization performance of both methods, we applied the nearest neighbor-based method to the face image super-resolution problem (i.e., single patch application) (Fig. 10). In the simple nearest neighbor reconstruction which replaces the input with the nearest stored pattern based on the Euclidean distance in the input image space, three faces were erroneously reconstructed while the other reconstructions are far better than those of KPCA as they happen to be near to one of the stored patterns. The high-contrast restoration approach used for the natural images failed to get any details. This mainly stems from the high dimensionality of the input images (60×60), since in this case the stored patterns are not dense enough in the input space. As a result, high- and low-frequency components from different images are mixed together in the reconstruction. Overall, the results indicate a better generalization capability of KPCA as compared to nearest neighbor methods.

5 Conclusion

In this article, we proposed a generative image model based on KPCA. In contrast to other patch-based modeling approaches, KPCA allows for nonlinear interactions between its basis images. Moreover, KPCA is capable of capturing part of the higher-

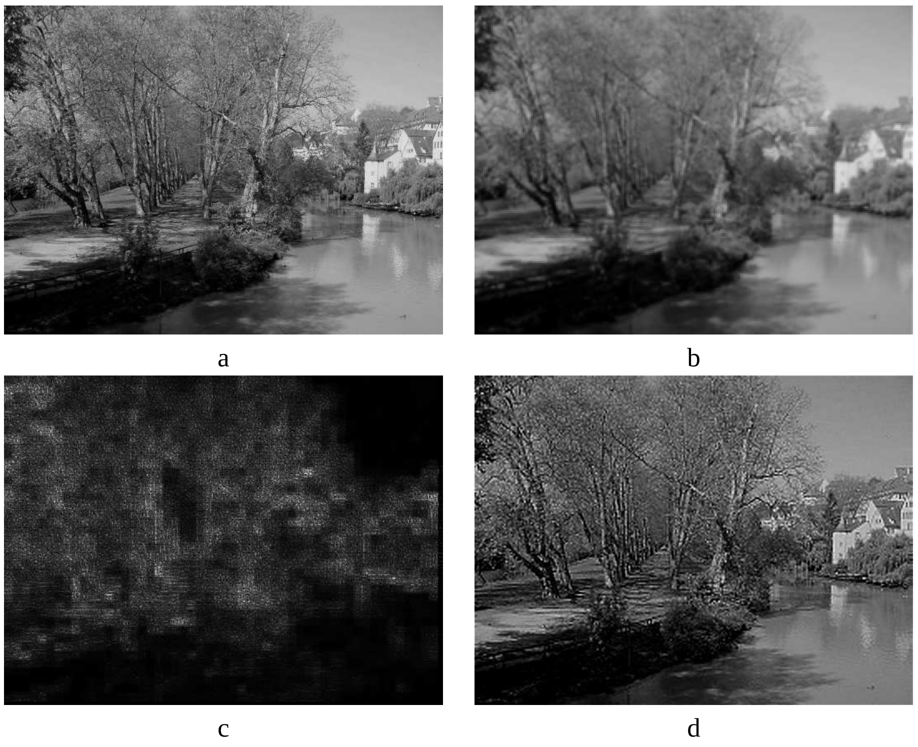

Figure 8: Example of natural image super-resolution: a. original image of resolution 396×528, b. low resolution image (264×198) stretched to the original scale, c. reconstruction of the high-frequency component (contrast enhanced for better visibility), and d. final KHA reconstruction.

order statistics which are particularly important for encoding image structure. To overcome the memory complexity of KPCA, the KHA was proposed as a method for the efficient estimation of kernel principal components. As a kernelization of the GHA, the KHA allows for computing KPCA without storing the kernel matrix, such that large datasets of high dimensionality can be processed. The presented experiments suggest that the generalization capabilities of the KPCA model exceed those of the previously used nearest-neighbor methods.

Compared to existing super-resolution and denoising methods, the experimental results obtained using KPCA are promising. In terms of reconstruction quality, however, it is difficult to compare our approach with the previous ones in [2] and [25] since this largely depends on subjective assessment, not on objective quantities such as the mean squared error (which is - as the results on PCA indicate - a poor quality measure for images). The main difference lies in the applied learning method: the methods proposed by Hertzmann et al. [2] and Freeman et al. [25] are based on *supervised* learning. In machine learning, it is generally believed that when feasible, a supervised learning approach often leads to the best results. However, at the same time, supervised algorithms can have shortcomings in that the data may be more expensive to obtain (since they require inputs *and* outputs), and the solution can be less flexible in that it is only useful for the exact task considered. This means that once trained on

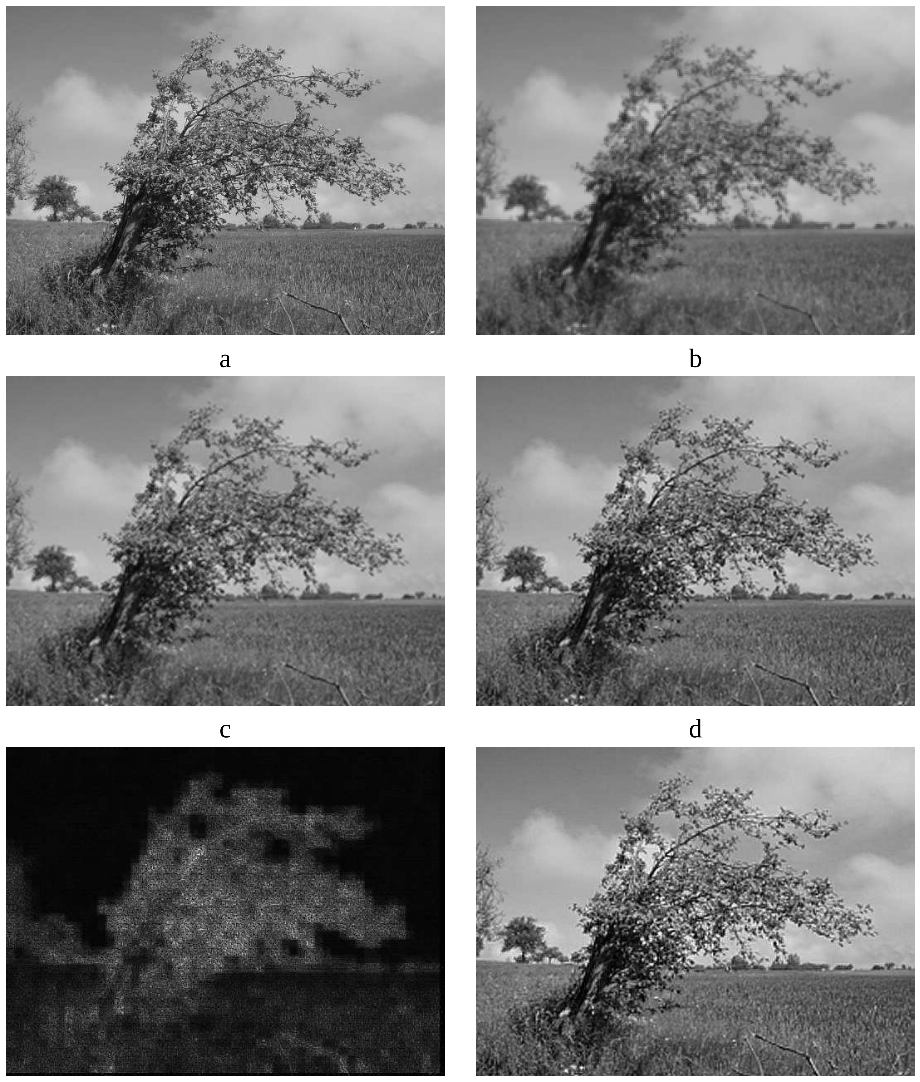

Figure 9: Comparison between different super-resolution methods: a. original image of resolution 396×528 , b. low resolution image (264×198) stretched to the original scale, c. bicubic interpolation, d. supervised example-based learning based on nearest neighbor classifier, e. unsupervised KHA reconstruction of high-frequency component (contrast enhanced for better visibility), and f. unsupervised KHA reconstruction.

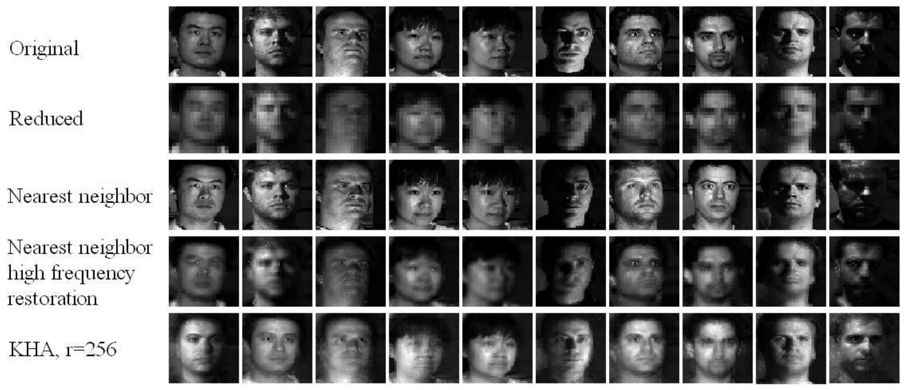

Figure 10: Face image super-resolution based on nearest neighbor methods [25]. First and second rows: original and low-resolution input images, respectively. Third row: reconstructions obtained by replacing the KPCA with a nearest neighbor classifier in the single patch reconstruction experiments. Fourth row: reconstructions obtained by the multi-patch reconstruction method applied to the input image as a single patch. Fifth row: reconstructions obtained by the KPCA.

a training set containing labeled data, the above methods can only be used for the one image super-resolution task it was trained for. In contrast, our image model, which is only trained on high-resolution images, can be directly applied to a variety of image restoration tasks, including denoising and image super-resolution using inputs of various resolutions, without retraining.

There are various directions for further work. The KHA, as a general iterative algorithm of KPCA applicable to large datasets, can significantly enlarge the application area of KPCA, which as a generic machine learning technique also enjoys some popularity in other fields. With respect to image modeling, the best choice of the kernel remains elusive. We still do not know which higher-order statistics are important for coding image content. It is also unclear to what extent the different available kernels are capable of modeling the occlusion and superposition phenomena that contribute to the generation of an image. A further investigation of these questions could lead to the design of new kernels that specifically incorporate the generation principles of natural images.

Acknowledgments. The authors greatly profited from discussions with A. Gretton, M. Hein, and G. Bakır.

References

[1] W. T. Freeman, E. C. Pasztor, and O. T. Carmichael. Learning low-level vision. *Intl. J. Comp. Vis.*, 40(1):25 – 47, 2000.

[2] A. Hertzmann, C. E. Jacobs, N. Oliver, B. Curless, and D. H. Salesin. Image analogies. In *Computer Graphics (Proc. Siggraph 2001)*, pages 327–340, NY, 2001. ACM Press.

[3] S. Zhu and D. Mumford. Prior learning and Gibbs reaction-diffusion. *IEEE Trans. PAMI*, 19(11):1236 – 1250, 1997.

[4] E. P. Simoncelli. Bayesian denoising of visual images in the wavelet domain. In P. Müller and B. Vidakovic, editors, *Bayesian inference in wavelet based models*, pages 291 – 308. Springer, New York, 1999.

[5] S. Mika, B. Schölkopf, A. J. Smola, K.-R. Müller, M. Scholz, and G. Rätsch. Kernel PCA and de-noising in feature spaces. In M. S. Kearns, S. A. Solla, and D. A. Cohn, editors, *Advances in Neural Information Processing Systems 11*, pages 536–542, Cambridge, MA, 1999. MIT Press.

[6] J. Malik, S. Belongie, T. Leung, and J. Shi. Contour and texture analysis for image segmentation. *Intl. J. Comp. Vis.*, 43(1):7 – 27, 2001.

[7] R. W. Buccigrossi and E. P. Simoncelli. Image compression via joint statistical characterization in the wavelet domain. *IEEE Trans. Image. Proc.*, 8(12):1688 – 1701, 1999.

[8] U. Grenander and A. Srivastava. Probability models for clutter in natural images. *IEEE Trans. PAMI*, 23(4):424 – 429, 2001.

[9] D. L. Ruderman. Origins of scaling in natural images. *Vis. Res.*, 37(23):3385 – 3395, 1997.

[10] A. B. Lee, D. Mumford, and J. Huang. Occlusion models for natural images: A statistical study of a scale-invariant dead leaves model. *Intl. J. Comp. Vis.*, 41(1/2):35 – 59, 2001.

[11] T. D. Sanger. Optimal unsupervised learning in a single-layer linear feedforward neural netowork. *Neural Networks*, 12:459–473, 1989.

[12] A. J. Bell and T. J. Sejnowski. The 'independent components' of natural scenes are edge filters. *Vision Research*, 37:3327–3338, 1997.

[13] J. Hurri, A. Hyvärinen, J. Karhunen, and E. Oja. Image feature extraction using independent component analysis. In *Proc. IEEE 1996 Nordic Conference of Signal Processing (NORSIG'96)*, 1996.

[14] B. A. Olshausen and D. J. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381:607–609, 1996.

[15] B. Schölkopf, A. Smola, and K. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, 1998.

[16] D. J. Field. What is the goal of sensory coding? *Neural Computation*, 6:559–601, 1994.

[17] E. Oja. A simplified neuron model as a principal component analyzer. *Journal of Mathematical Biology*, 15:267–273, 1982.

[18] S. Haykin. *Neural Networks*. Prentice Hall, Upper Saddle River, NJ, 1999.

[19] P. J. B. Hangcock, R. J.Baddely, and L. S Smith. The principal components of natural images. *Network*, 3:61 – 70, 1992.

[20] R. J. Baddeley and P. J. B. Hancock. A statistical analysis of natural images predicts psychophysically derived orientation tuning curves. *Proc. R. Soc. B*, 246(1317):219 – 223, 1991.

[21] B. Schölkopf and A. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

[22] K. I. Kim, M. O. Franz, and B. Schölkopf. Kernel Hebbian algorithm for iterative kernel principal component analysis. Technical Report 109, Max-Planck-Insitut für biologische Kybernetik, Tübingen, June 2003.

[23] S. Baker and T. Kanade. Limits on super-resolution and how to break them. *IEEE Trans. Pattern Analalysis and Machine Intelligence*, 24(9):1167–1183, 2002.

[24] R. Keys. Cubic convolution interpolation for digital image processing. *IEEE Trans. Acoustics, Speech, Signal Processing*, 29(6):1153–1160, 1981.

[25] W. T. Freeman, T. R. Jones, and E. C. Pasztor. Example-based super-resolution. *IEEE Computer Graphics and Applications*, 22(2):56–65, 2002.

[26] C. J. C. Burges. Simplified support vector decision rules. In L. Saitta, editor, *Proceedings of the 13th International Conference on Machine Learning*, pages 71–77, San Mateo, CA, 1996. Morgan Kaufmann.

[27] J. T. Kwok and I. W. Tsang. Finding the pre-images in kernel principal component analysis. *6th Annual Workshop On Kernel Machines*, Whistler, Canada, 2002, poster available at http://www.cs.ust.hk/~jamesk/kernels.html.

[28] R. C. Gonzalez and R. E. Woods. *Digital Image Processing*. Addison Wesley, Massachusetts, 1992.

[29] A. S. Georghiades, P. N. Belhumeur, and D. J. Kriegman. From few to many: illumination cone models for face recognition under variable lighting and pose. *IEEE Trans. Pattern Analalysis and Machine Intelligence*, 23(6):643–660, 2001.

[30] S. Mika, G. Rätsch, J. Weston, B. Schölkopf, and K.-R. Müller. Fisher discriminant analysis with kernels. In Y.-H. Hu, J. Larsen, E. Wilson, and S. Douglas, editors, *Neural Networks for Signal Processing IX*, pages 41–48. IEEE, 1999.

[31] A. Turk and A. P. Pentland. Face recognition using eigenfaces. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 586–591, 1991.

[32] S. Romdhani, S. Gong, and A. Psarrou. A multiview nonlinear active shape model using kernel PCA. In *Proceedings of BMVC*, pages 483–492, Nottingham, UK, 1999.

[33] J. Ham, D. D. Lee, S. Mika, and B. Schölkopf. A kernel view of the dimensionality reduction of manifolds. Technical Report 110, Max-Planck-Insitut für biologische Kybernetik, Tübingen, July 2003.