Binary Principal Component Analysis

Feng Tang and Hai Tao
University of California, Santa Cruz
tang|tao@soe.ucsc.edu

Abstract

Efficient and compact representation of images is a fundamental problem in computer vision. Principal Component Analysis (PCA) has been widely used for image representation and has been successfully applied to many computer vision algorithms. In this paper, we propose a method that uses Haar-like binary box functions to span a subspace which approximates the PCA subspace. The proposed method can perform vector dot product very efficiently using integral image. We also show that B-PCA base vectors are nearly orthogonal to each other. As a result, in the non-orthogonal vector decomposition process, the expensive pseudo-inverse projection operator can be approximated by the direct dot product without causing significant distance distortion. Experiments on real image datasets show that B-PCA has comparable performance to PCA in image reconstruction and recognition tasks with significant speed improvement.

1 Introduction

Principal component analysis(PCA) [3] is widely used in applications ranging from neuroscience to computer vision [9, 1, 2, 4] to extract relevant information from high dimensional data sets. It reduces the dimension of a data set to reveal its essential characteristics and the subspace captures the main structure of the data. The main computation in PCA is the dot products of a data vector with the PCA base vectors. This can be computationally expensive especially when the original data dimension is high because it involves element-by-element floating point multiplications. This paper investigates the possibility of finding a subspace representation that is similar to PCA in terms capturing the essential data structure while avoiding the expensive floating point dot product operations.

1.1 Related work

In recent years, Haar-like box functions became a popular choice as image features due to the seminal face detection work of Viola and Jones [11]. Examples of such features used in our method are shown in Figure 1. The main advantage of such features is that the inner product of a data vector with a box function can be performed by three or seven integer additions, instead of d floating point multiplications, where d is the dimension of the base vectors. In a more recent paper [7], the authors proposed a method to represent images using these simple one- or two-box base vectors. Since the binary box base vectors are highly non-orthogonal to each other, the representation is called non-orthogonal binary subspace (NBS). The base vectors of NBS are selected using a greedy algorithm

BMVC 2006 doi:10.5244/C.20.39

- OOMP (optimized orthogonal matching pursuit) [5]. NBS has shown to be an efficient representation for the task of template matching, image reconstruction and recognition. However, NBS has the following drawbacks, 1) the NBS base vector lacks a meaningful interpretation which is important in both image representation and recognition. 2) the NBS base vectors are usually highly non-orthogonal to each other, which makes the L-2 norm in NBS significantly deviate from the Euclidean distance. Therefore, it leads to numerical problems in recognition. These problems with NBS have motivated us to find a better subspace representation that is computationally efficient as NBS but with enhanced representation power for the recognition task.

PCA is an orthogonal subspace and has been proved to have good performance in image representation and recognition. Its capability to capture image structure information, orthogonal base vectors are complement to NBS. This makes it natural to combine NBS and PCA to the proposed binary PCA representation. Various extensions of PCA have been proposed in computer vision including Probabilistic PCA(P-PCA) [8], kernel PCA(K-PCA) [6], generalized PCA(G-PCA) [10], but to our knowledge, our paper is the first that approximates the PCA using non-orthogonal binary box functions for the purpose of efficient computation.

Figure 1: Three typical one- and two-box functions.

1.2 Contributions

In this paper, we combine the advantage of PCA and Haar-like binary box functions into a novel non-orthogonal subspace representation called Binary PCA (B-PCA) that can be computed very efficiently and at the same time capture the structure information of the data. B-PCA spans a subspace that approximates the PCA subspace. Yet each B-PCA base vector is a linear combination of a small number of Haar-like box functions. As the result, the dot product with each B-PCA base vector can be computed with significantly reduced number of operations using the integral image trick. Main contributions of this paper are:

- A novel non-orthogonal image subspace representation with each base vector as a linear combination of Haar-like box functions.
- A fast PCA-embedded optimized orthogonal matching pursuit (OOMP) algorithm to find the B-PCA base vectors.
- Applications of the proposed representation in image reconstruction and object recognition tasks.

The rest of the paper is organized as follows: Some background about PCA is briefly discussed in section 2. In section 3, we present the B-PCA algorithm. Experimental results are demonstrated in section 4. We conclude the paper and highlight future work in section 5.

2 Background - principal component analysis

The intuition behind the PCA method is to find a set of base vectors $\Theta = [\mathbf{e}_1, \mathbf{e}_2, ..., \mathbf{e}_n]$ so that they explain the maximum amount of variance of the data. PCA can be defined with an incremental formulation. Suppose we have determined the first $k-1$ principal components, the *k-th* principal component $\mathbf{e}_k$ is determined as the principal vector of the residual:

$$\arg\max_{\mathbf{e}} E\{[\mathbf{e}^T(\mathbf{X} - R_{\Theta_{k-1}}(\mathbf{X}))]^2\} \tag{1}$$

$$subject\ to: ||\mathbf{e}|| = 1$$

where $\mathbf{x} \in R^N$ denotes the data vector, $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n]$ denotes a set of data vectors with zero mean. $\mathbf{e}_i$ is of the same dimension of the data vector $\mathbf{x}_i$. $R_{\Theta_{k-1}}(\mathbf{X}) = \Theta_{k-1}\Theta_{k-1}^T\mathbf{X}$ is the reconstruction of the data using the first $k-1$ principal components. It can be proved that $\mathbf{e}_i$'s are the eigenvectors of the data covariance matrix that correspond to the *i-th* largest eigenvalues.

3 Binary principal component analysis

3.1 The problem formulation

Similar to PCA, B-PCA tries to find a set of base vectors that encompass most of the energy in the dataset. However, an additional requirement for B-PCA is that each base vector has to be a linear combination of a small number of Haar-like box functions. Similar to the PCA, we formulate the B-PCA in an incremental way: having determined the first $(k-1)$-*th* B-PCA bases, the *k-th* base vector ψ_k is determined as:

$$\arg\max_{\psi} E\{[\psi^T(\mathbf{X} - R_{\Psi_{k-1}}(\mathbf{X}))]^2 - \beta cost(\psi)\} \tag{2}$$

$$subject\ to: ||\psi|| = 1; \psi = \textstyle\sum_{j=1}^{N_k} c_{j,k}\mathbf{b}_{j,k}; \mathbf{b}_{j,k} \in D$$

where $\Psi_{k-1} = [\psi_1, ..., \psi_{k-1}]$ is the set of B-PCA base vectors selected up to iteration $k-1$ and $R_{\Psi_{k-1}}(\mathbf{X})$ is the reconstruction of the data $\mathbf{X}$ using the bases Ψ_{k-1}. Since B-PCA base vectors are linear combination of a small number of binary box functions, they may not be orthogonal to each other. As a result, the reconstruction process becomes $R_{\Psi_{k-1}}(\mathbf{X}) = \Psi_{k-1}(\Psi_{k-1}^T\Psi_{k-1})^{-1}\Psi_{k-1}^T\mathbf{X}$. The term $\mathbf{X} - R_{\Psi_{k-1}}(\mathbf{X})$ is the reconstruction residual. $cost(\psi)$ is the cost function associated with computing the dot product between a data vector and ψ. It is roughly proportional to the number of binary box functions N_k that are used to represent ψ. A simplified cost term used in our implementation will be discussed in the next subsection. The fact that each B-PCA base vector is represented in NBS is reflected in the constraint $\psi_k = \sum_{j=1}^{N_k} c_{j,k}\mathbf{b}_{j,k}$ and $\mathbf{b}_{j,k} \in D$. Note the $c_{j,k}$ is different from the direct NBS approximation coefficients, they are normalized to enforce the constraint $||\psi|| = 1$.

The intuition behind this objective function is that, the new base vector ψ_i needs to capture most of the data variance (the first term) and at the same time is a linear combination of a few binary box functions $\mathbf{b}_{j,k}$ which makes it computationally efficient.

As can be observed, the main difference between PCA and B-PCA is the additional constraint that B-PCA base vectors need to be represented in NBS. These B-PCA base

vectors are in general non-orthogonal, but they span roughly the same subspace as PCA. Fig.2 illustrates the relation between a PCA subspace and a B-PCA subspace.

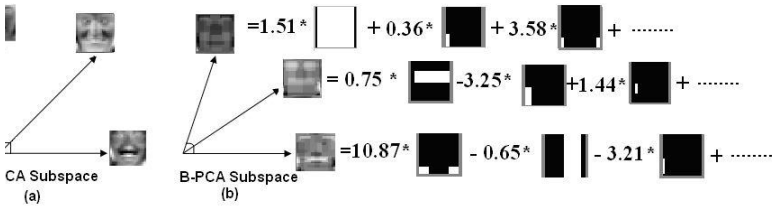

Figure 2: Relation of the PCA subspace and the B-PCA subspace. (a) is the PCA base vectors which are orthogonal to each other. (b) is the B-PCA base vectors which are approximation of the PCA bases using NBS. We can observe block effect in the B-PCA base vectors, which is the consequence of such an approximation.

3.2 The solution - PCA guided NBS

The search space for the optimization problem in Eq.2 is extremely large because the solution can be any base vector that is a linear combination of any box functions from the binary dictionary D. Even for a small image of size 24×24 used in our experiments, there are 134998 box functions in D. Suppose each B-PCA base vector is represented by 10 box functions, the number of possible choices of box functions for a single B-PCA base vectors is C^{10}_{134998}. This makes it impractical to find the global optimal solution.

One possible solution is to apply the PCA on the training data to obtain k principal components $[\mathbf{e}_1, ..., \mathbf{e}_k]$, then employ NBS to approximate each of these principal components with a given precision, and use the approximated vectors as the B-PCA base vectors $[\psi_1, ..., \psi_k]$. But the problem with this solution is that the approximation errors $(\mathbf{e}_i - \psi_i)$ are generally not represented by any of the B-PCA base vectors, this leads to an inaccurate subspace.

To overcome this problem, we propose a PCA guided NBS method to find a sub-optimal solution efficiently. In the PCA guided NBS, we denote the selected B-PCA base vectors up to iteration k as $\Psi_k = [\psi_1, \psi_2, ..., \psi_k]$. This set is empty at the beginning. We start from the original PCA procedure to obtain the first principal component that captures the majority of the data variance. We call the first principal component the *Pre-BPCA* vector, denoted as ψ_1^-. NBS is then applied to approximate this vector as $\psi_1 = \sum_{j=0}^{N_1} c_{j,1}\mathbf{b}_{j,1}$. Then, in iteration k, the data $\mathbf{X}$ is projected to the subspace spanned by the already selected B-PCA bases Ψ_{k-1}, and PCA is applied on the residual of the data $\mathbf{X} - R_{\Psi_{k-1}}(\mathbf{X})$ to obtain the next *Pre-BPCA* ψ_k^- which is again approximated using NBS. The approximation of *Pre-BPCA* at iteration k is called the *k-th B-PCA base vector*. This procedure iterates until the desired number of B-PCA bases have been obtained or an error threshold is reached. The procedure of this optimization is shown in Algorithm 3.1.

Algorithm 3.1: [BPCAs, BOXFUNCTIONS] = BPCA($X, N, thresh$)

comment: Compute the BPCA subspace bases.

comment: X is the training data, N is the number of B-PCA base vectors to be computed.

comment: *thresh* is the Pre-BPCA approximation threshold.

comment: *BPCAs* is the array of BPCA base vectors.

comment: *NBS* is the binary box base vectors to represent each *BPCAs*.

X←X−*mean*(X);
for $k \leftarrow 1$ **to** N
do
Pre-BPCA← PCA(X)
(BPCAs[k],NBS[k]) ← OOMP(Pre-BPCA,thresh)
X←X−RECONSTRUCTION(X,BPCAs)
return (BPCAs,NBS)

The function $PCA(\mathbf{X})$ returns the first PCA base vector for the given data $\mathbf{X}$. The function $OOMP(\mathbf{X}, thresh)$ returns the approximation of the given signal $\mathbf{X}$ with pre-defined precision threshold *thresh* and the binary box functions used. $BPCAs[k]$ is the B-PCA base vector at iteration k, $NBS[k]$ is the set of binary box functions selected to approximate $BPCAs[k]$. The function $Reconstruction(\mathbf{X}, \Psi)$ returns the representation of the signal $\mathbf{X}$ in the non-orthogonal subspace spanned by Ψ, i.e., it returns $\Psi(\Psi^T\Psi)^{-1}\Psi^T\mathbf{X}$.

Generally, it takes a large number of box functions to represent each *Pre-BPCA* perfectly. The computation cost term in the objective function prefers a solution with fewer box functions. To make the optimization simpler, we enforce the computation cost constraint by finding the minimum number of box functions that satisfy:

$$\frac{1}{N}\sum_{i=1}^{N}\left|\frac{(\psi_1^- - \overline{\psi_1^-})_i}{(\psi_1^-)_i}\right| \leq \zeta \tag{3}$$

where $\overline{\psi_1^-}$ is the reconstruction of ψ_1^- using binary box functions. $\zeta \in [0,1]$ is the approximation error threshold that controls the precision. N is the dimension of the base vector. $(\cdot)_i$ means the $i-th$ element of a vector. A smaller value of ζ tends to produce a more accurate approximation. This means the element-wise approximation error ratio should be smaller than a threshold. This constraint is more strict than the norm constraint which requires the difference in norm to be smaller than a threshold. The comparison of PCA, pre-BPCA and B-PCA base vectors are shown in Fig.3.

4 Experiments

We applied the proposed B-PCA method to solving modelling and recognition problems of faces and vehicles. Promising results have been achieved and are demonstrated in this section.

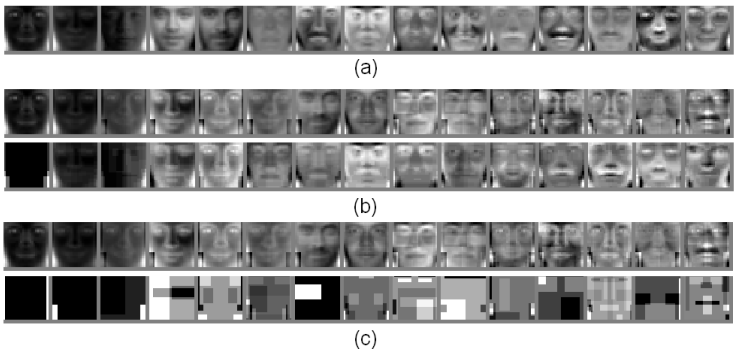

Figure 3: Approximations of the first 15 Pre-BPCA base vectors using different thresholds ζ. (a) the PCA base vectors. (b)B-PCA base vectors with $\zeta = 0.2$ Pre-PCA base vectors(first row) and B-PCA base vectors(second row). (c)B-PCA base vectors with $\zeta = 0.85$ which has obvious block effect.

4.1 Effectiveness of B-PCA for reconstruction

The PCA guided NBS method described in section 3 is implemented to obtain the B-PCA face subspace. To model face subspace, 500 frontal view images from the FERET database were used. These images were spatially aligned using an affine transform based on the facial feature points. Each of these aligned images is then cropped and scaled down to 24×24 pixels. Using 500 training samples, the B-PCA base vectors are computed. The first 15 of these vectors are shown in Fig. 3. It can be observed that, like PCA, B-PCA base vectors can capture the face structure. Each individual base vector resembles some face shape. However, there is some block effect in the B-PCA base vectors due to the approximation error using box functions.

The B-PCA bases reconstruction performance are directly influenced by the approximation threshold ζ in the PCA guided OOMP. With a higher threshold, which means the approximation is less accurate, the bases become less orthogonal. When the base vectors are more orthogonal (smaller ζ), the reconstruction error will be smaller, because B-PCA base vectors become more similar to PCA base vectors. We have listed the angle between the B-PCA base vectors with different approximation errors in Table-I. The reconstruction performace using different approximation thresholds are plotted in Fig. 4.

To demonstrate the generality of the B-PCA subspace, we make another experiment to model the rear view vehicles. The dataset contains 1200 rear view cars, some of them are shown in Fig. 6. Then we train the B-PCA subspace using 800 images and use the other 400 images as testing for the reconstruction error. Fig. 8 shows that the first 3 B-PCA base vectors and the corresponding box functions used to represent them. The reconstruction performance comparison between PCA and B-PCA with different approximation errors is illustrated in Fig. 5.

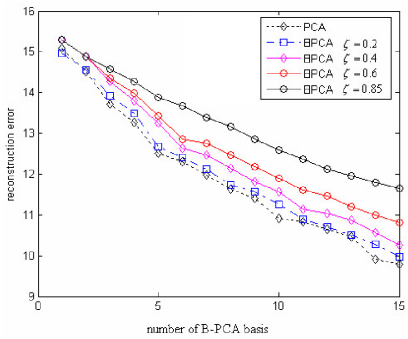

Figure 4: Reconstruction error using different approximation errors for faces.

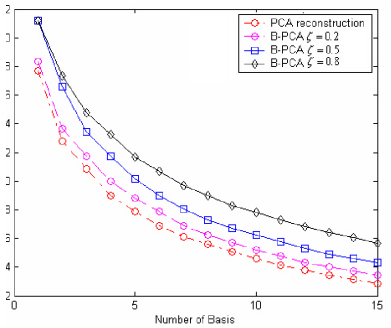

Figure 5: Reconstruction error using different approximation errors for rear view vehicles.

Figure 6: "Rear view vehicle" dataset with 1200 images, each image is of size 20×30.

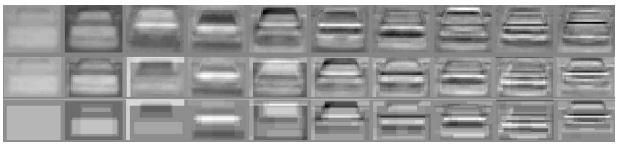

Figure 7: Comparison of different bases (PCA, Pre-BPCA and B-PCA) for rear views of vehicles. the left column is the PCA base vectors of the original data; the middle column is the Pre-BPCA base vectors; the right column is the B-PCA base vectors (approximation threshold is $\zeta = 0.4$).

4.2 B-PCA for Face Recognition

To demonstrate the effectiveness of the proposed binary PCA method, we applied it to face recognition. The result is compared with that of the "Eigenface" method. Although "Eigenface" is not among the latest algorithms in face recognition, it provides a performance base line. Comparison with other more sophisticated eigen-face based methods

Table 1: B-PCA base vectors orthogonality with different approximation thresholds.

Approximation threshold ζ	0.2	0.5	0.8
Min-angle between bases(in degree)	89.6424	84.9034	68.4516

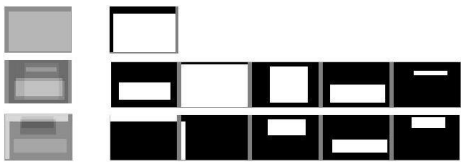

Figure 8: B-PCA base vectors and its corresponding box functions for vehicle modeling.

could be similar. The test database comprises 64 aligned face images in the FERET database. Each image is projected to the PCA and B-PCA subspace and the coefficients are used as features. A nearest neighbor classifier is used to find the best match as the recognition result. We compared the PCA with B-PCA using the original projection process which has the pseudo-inverse $(\Psi^T\Psi)^{-1}\Psi^T\mathbf{x}$ and also that of direct dot product (DNP) $\Psi^T\mathbf{x}$ for recognition, the results are shown in Fig. 9 and Fig. 10, in Fig. 9, the approximation threshold ζ is set to 0.9, in Fig. 10, ζ is 0.1. As can be observed, when $\zeta = 0.1$, which means the B-PCA base vectors are more orthogonal, the difference between recognition rates using pseudo-inverse and DNP is very small. The DNP, $\Psi^T\mathbf{x}$, is computationally much cheaper because it only needs several additions. The difference between $(\Psi^T\Psi)^{-1}\Psi^T x$ and $\Psi^T\mathbf{x}$ is small when the B-PCA base vectors are more orthogonal (smaller ζ), and it is large when the base vectors are less orthogonal (larger ζ). As we can observe that when the approximation threshold ζ is small, the difference between B-PCA and PCA base vectors is small, the difference between DNP and pseudo-inverse projection is also small.

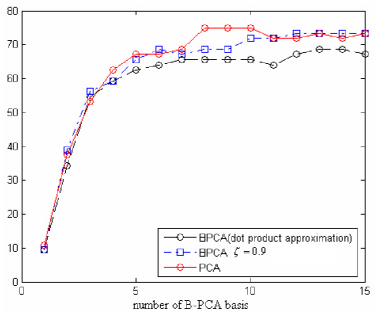

Figure 9: Recognition performance for face dataset with $\zeta = 0.9$.

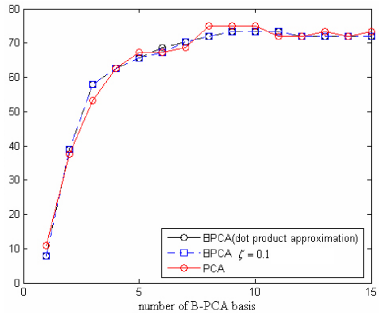

Figure 10: Recognition performance for face dataset with $\zeta = 0.1$.

4.3 Speed Improvement of B-PCA

Suppose the image size is $m \times n$, T_{PCA} denotes the time for computing the PCA subspace projection coefficients. N denotes the number of PCA base vectors. It will take $m \times n \times N$ floating point multiplications and $N \times (m \times n - 1) + (N - 1)$ floating point additions to

perform the projection operation, or

$$T_{PCA} = N \times m \times n \times T_{fm} + [N \times (m \times n - 1) + (N - 1)] \times T_{fa} \quad (4)$$

where T_{fm} is the time for a single floating point multiplication, T_{fa} is the time for a single floating point addition.

For B-PCA, the time for a single projection is denoted as T_{BPCA}. It consists of two parts, one is T_{ii}, the time to construct the integral image, for a $m \times n$ image, it will take $m \times n \times 2$ integer additions with recursive implementation. This is performed only once for each image. The other is the time for the projection operation $P_\Psi(\mathbf{x}) = (\Psi^T\Psi)^{-1}\Psi^T\mathbf{x}$. When the bases are nearly orthogonal to each other, we can approximate the projection coefficient using direct dot-product $\Psi^T\mathbf{x}$. The B-PCA base vector ψ_i ($1 \le i \le N$) is represented as a linear combination of N_i box functions, $\psi_i = \sum_{j=1}^{N_i} c_j \mathbf{b}_j$. The projection of $\mathbf{x}$ to ψ_i can be written as $\langle \psi_i, \mathbf{x} \rangle = \sum_{j=1}^{N_i} c_{i,j} \langle \mathbf{b}_j, \mathbf{x} \rangle$. Each box function $\mathbf{b}_j$ has n_j boxes, where n_j can be one or two. The $\langle \mathbf{b}_j, \mathbf{x} \rangle$ can be performed using $3 \times n_j$ integer additions. Since c_j is floating point, $\langle \psi_i, \mathbf{x} \rangle$ needs N_i floating point multiplications and $N_i - 1$ floating point additions.

$$T_{BPCA} = \sum_{i=1}^{N} \sum_{j=1}^{N_i} (3 \times n_j \times T_{ia} + N_i \times T_{fm} + (N_i - 1) \times T_{fa}) \quad (5)$$

where T_{ia} is the time for one integer addition. As we can observe, T_{BPCA} is only dependent on the number of binary box functions which is often much smaller than the dimension of the image. While for PCA, the time is proportional to the image dimension. Since the number of operations in B-PCA is much smaller than PCA, T_{BPCA} is much faster than T_{PCA}, and the speed up is more dramatic when the dimension of data is higher.

Table 2: Comparison of the computational cost between PCA and B-PCA projection operation

Approximation threshold: ζ	0.4	0.6	0.85
# binary box functions	409	214	68
T_{PCA} (a single subspace projection (s))	3.15×10^{-4}		
T_{ii} (integral image computation (s))	4.72×10^{-5}		
T_{BPCA} (a single subspace projection (s))	8.11×10^{-6}	4.29×10^{-6}	1.45×10^{-6}
$Speedup(\frac{T_{PCA}}{T_{BPCA}+T_{ii}/N})$	27.97	42.34	68.47

Suppose $m = n = 24$, $N = 15$, T_{PCA} needs $24 \times 24 \times 15 = 8640$ floating point multiplications to compute the projection coefficients. Suppose the total number of NBS base vectors used to represent all the B-PCA base vectors is 200, that is, $\sum_{i=1}^{N} N_i = 200$, the B-PCA needs only $2 \times \sum_{i=1}^{N} N_i - 1 = 399$ floating point operations. The speed up is significant.

The experiment for speed improvement is carried out on a Pentium 4, 3.2GHz, 1G RAM machine, using C++ code. We compute the 15 PCA base vectors and 15 B-PCA base vectors, and evaluate the computation time for an image to project onto the PCA subspace and B-PCA subspace. We tested the PCA projection and B-PCA projection with

1000 images and the time for a single projection is computed as the average. In Table-IV, we can observe, the B-PCA projection process $\Psi^T \mathbf{x}$ is much faster than direct PCA projection operation. The improvement is even more dramatic if more base vectors are used for comparison. Note: 1) for PCA projection test, the image data is put in continuous memory space to avoid unnecessary memory access overhead; 2) the T_{ii} is distributed into each base vector projection operation to make the comparison fair, because integral image is only needed to be computed once.

5 Conclusion and future work

A novel compact and efficient representation called B-PCA is presented in this paper. It can capture the main structure of the dataset while take advantages of the computational efficiency of NBS. A PCA-embedded OOMP method is proposed to obtain the B-PCA basis. We have applied the B-PCA method to the image reconstruction and recognition tasks. Promising results are demonstrated in this paper. Future work may use more sophisticated classification methods to enhance the recognition performance.

References

[1] M.J. Black and A.D. Jepson. Eigentracking: Robust matching and tracking of articulated objects using a view-based representation. In *ECCV*, pages 329–342, 1996.

[2] T.F. Cootes, G.J. Edwards, and C.J. Taylor. Active appearance models. In *ECCV*, pages II 484–498, 1998.

[3] I. Jollife. *Principal Component Analysis*. Springer-Verlag, 1986.

[4] H. Murase and S.K. Nayar. Visual learning and recognition of 3d objects from appearance. *IJCV*, 14(1):5–24, June 1995.

[5] L. Rebollo-Neira and D. Lowe. Optimized orthogonal matching pursuit approach. *IEEE Signal Processing Letters*, 9(4):137–140, April 2002.

[6] B. Scholkopf, A. Smola, and K.R. Muller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, July 1998.

[7] H. Tao, R. Crabb, and F. Tang. Non-orthogonal binary subspace and its applications in computer vision. In *ICCV*, pages 864–870, 2005.

[8] M. Tipping and C. Bishop. Probabilistic principal component analysis. *J. Royal Statistical Soc.*, 61(3):611–622, 1999.

[9] M. Turk and A. Pentland. Face recognition using eigenface. In *CVPR*, pages 586–591, 1991.

[10] R. Vidal, Y. Ma, and S. Sastry. Generalized principal component analysis (gpca). *PAMI*, 27(12):1–15, December 2005.

[11] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *CVPR*, pages I: 511–518, 2001.