Clustered Blockwise PCA for Representing Visual Data

Ko Nishino, *Member*, *IEEE*,
Shree K. Nayar, *Member*, *IEEE*, and
Tony Jebara, *Member*, *IEEE*

Abstract—Principal Component Analysis (PCA) is extensively used in computer vision and image processing. Since it provides the optimal linear subspace in a least-square sense, it has been used for dimensionality reduction and subspace analysis in various domains. However, its scalability is very limited because of its inherent computational complexity. We introduce a new framework for applying PCA to visual data which takes advantage of the spatio-temporal correlation and localized frequency variations that are typically found in such data. Instead of applying PCA to the whole volume of data (complete set of images), we partition the volume into a set of blocks and apply PCA to each block. Then, we group the subspaces corresponding to the blocks and merge them together. As a result, we not only achieve greater efficiency in the resulting representation of the visual data, but also successfully scale PCA to handle large data sets. We present a thorough analysis of the computational complexity and storage benefits of our approach. We apply our algorithm to several types of videos. We show that, in addition to its storage and speed benefits, the algorithm results in a useful representation of the visual data.

Index Terms—Principal component analysis, singular value decomposition, eigenvalues and eigenvectors, natural image statistics, clustering, region growing/partitioning.

1 Scaling PCA

THERE are many instances in computer vision and image processing where we want to reduce the redundancy within a collection of images. These situations frequently occur when many observations need to be handled, for example, in object or face recognition, object tracking, and image-based rendering. Given a data set that presumably spans a manifold much smaller in dimension than its raw embedding dimensionality, it is traditional to apply Principal Component Analysis (PCA) [12], [14] to extract the subspace. PCA handles a Euclidean vector space and from several independent identically distributed vectors (or multidimensional point observations) recovers subspaces that are optimal in the least squared-error reconstruction sense [22].

However, the inherent computational complexity of PCA can make it hard to scale. As a data set becomes large, the corresponding covariance/data matrix grows and direct eigen decomposition/SVD can become intractable. Also, as data increases in size, the memory required to even store the data matrix also grows and may push the storage limit of the machine. Furthermore, since eigen decomposition on an $n \times n$ covariance matrix requires $O(n^3)$ computation, the runtime of PCA rapidly increases with the length of an image sequence.

Note that PCA can be applied to any given data set by breaking the data into pieces and vectorizing them. PCA can then be applied to each piece of vector treating them as if they were independent identically distributed. Fig. 1 illustrates the memory required to store an image sequence after the application of PCA, as a function of the spatial extent x (span) of the pieces the data is broken into. In this illustration, we are assuming the same reconstruction error for each of the cases along this axis. The bar on the right represents the commonly used approach where the data is not broken down in any way—the span is the size of the image (we call this global PCA). On the left, we show the memory needed to store the raw data, which is equivalent to applying PCA to each pixel (referred to as pixelwise PCA). In this paper, we are interested in characterizing the shaded region in between the two bars.

One might imagine that the amount of storage in Fig. 1 gradually drops from the left bar to the right one and that we cannot achieve better compression than global PCA (curve (a) in Fig. 1). This is the case when the raw visual variation of the data already lies in an extremely low-dimensional subspace; for instance, when the image sequence is of objects with pure Lambertian reflection. Another extreme case is when the whole data volume is filled with spatially and temporary high-frequency variations; for instance, if the image sequence is of the appearance of a mirror under varying complex illumination environment. In such cases, PCA would not achieve compression of the data and, hence, partitioning it further would simply add storage overhead associated with multiple subspaces (curve (b) in Fig. 1).

Fortunately, research on natural image statistics [24] reveals that for almost any visual data, high-frequency variations in the data tend to be localized in space and time in regions that are distributed over the data. In addition, the spatial and temporal variations within a local region may be correlated with those of other regions [24]. These phenomena almost always ensure there exists a minimum within the shaded region in Fig. 1 that is significantly more efficient in storage and runtime than global PCA (curve (c) in Fig. 1).

In this paper, we present an algorithm that enables a user to explore the shaded region in Fig. 1. The algorithm has two key attributes:

- **Partitioning:** It partitions the data volume into smaller blocks and applies PCA to the blocks individually. We refer to this approach as **blockwise PCA**. With blockwise PCA, we are able to take full advantage of the local linearity in the data.
- **Grouping:** It subsequently clusters the subspaces obtained from blockwise PCA, merging them into clusters each with a single subspace. We refer to this as **clustered blockwise PCA**. This merging of subspaces enables us to tie together blocks with similar local subspaces or statistics to achieve greater efficiency in representation by avoiding redundant encoding of subspaces.

In practice, these simple ideas work surprisingly well. We can not only scale PCA to handle large data sets, but also simultaneously achieve greater efficiency in the representation of the visual data. Note that, since we do not rely on any specific PCA algorithm, any matrix decomposition technique can be used to obtain the local subspaces. We have conducted a detailed complexity analysis (storage and time) of our algorithm and its benefits over global PCA. We present experimental results that verify our complexity claims.

2 Related Work

Extensive work has been done to make PCA applicable to large data sets. In particular, incremental methods have drawn wide attention because of their memory efficiency. Several methods have been proposed that update an already computed subspace as we add observations in an online setting time [3], [4], [20], [10]. However, these incremental methods are known to be prone to errors which may accumulate over time. While variants have been proposed to address this problem [9], there exists an inherent tradeoff between error and computational efficiency. While all these incremental methods are computationally efficient, none of them exploit the local linearity or the spatio-temporal correlations within the image sequence.

However, several nonincremental PCA-based algorithms that take account of spatial or temporal correlation in the data have been proposed [15], [16], [1], [19], [27]. These methods exploit the local spatial correlation within fixed and connected regions obtained manually or automatically. These spatially local PCA methods can be considered as special cases of our framework: We also take

• *The authors are with the Department of Computer Science, Columbia University, MC 0401, 1214 Amsterdam Avenue, New York, NY 10027. E-mail: {kon, nayar, jebara}@cs.columbia.edu.*

Manuscript received 4 Aug. 2003; revised 16 Dec. 2004; accepted 4 Feb. 2005; published online 11 Aug. 2005.
Recommended for acceptance by J. Weng.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-0206-0803.

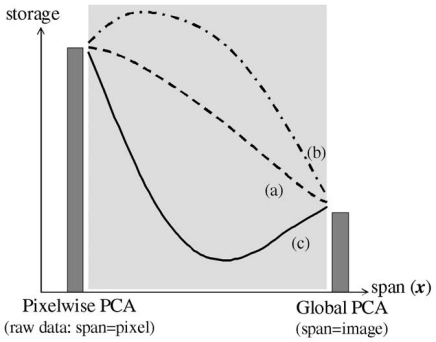

Fig. 1. PCA can be applied to a volume of image data by breaking into vector pieces with span x. The bar on the left represents the storage needed for the raw data, which is equivalent to applying PCA at a pixel level. The bar on the right corresponds to the traditional approach of applying PCA at an image level (global PCA). We are assuming the reconstruction error is the same for each of the cases. The nature of visual data is such that there is almost always a minimum in the shaded region in between. This span size can be significantly more efficient for storage and runtime than either extreme (curve (c)).

temporal correlation into account and also allow the local regions to be unconnected spatially and temporarlly.

Murase and Lindenbaum [21] utilize DCT coefficients to group small image regions along the temporal axis into four different classes to which PCA is applied individually. Jebara et al. [13] extended the method in [19] to group pixels into a fixed number of subspaces through an EM-based algorithm. These methods, which exploit both the spatial and temporal correlation in the data, can also be considered as special cases of our framework: We allow each of the resulting subspace corresponding to several spatio-temporal regions have different dimensionality which is determined adaptively.

From a statistical point of view, the distribution of the variation of visual data can be far from a Gaussian distribution, while PCA assumes the data it is applied to is Gaussian. However, variations within local regions can often be modeled with Gaussian distributions and regions in different spatio-temporal locations may have identical Gaussian distributions. A recent Bayesian analysis of PCA [2] suggests that there is a trade-off between the number of subspaces, the dimensionality of each subspace, and the dimensionality of the data. Our clustered blockwise PCA framework seeks to exploit this tradeoff.

3 Blockwise PCA

We first consider blockwise PCA by exploring various partitionings of the visual data volume (of size $W \times H \times F = \text{width} \times \text{height} \times \text{frames}$) and discuss its computational complexity, its storage requirements, and compare it to global PCA.

3.1 Computational Complexity

When we apply global PCA, we first vectorize each image in a raster scan manner and stack the vectors as columns to obtain a $WH \times F$ data matrix. Then, either eigen decomposition can be applied to the covariance matrix of the data matrix or SVD can be directly applied to the data matrix to obtain the eigenvectors and eigenvalues (singular values are square roots of eigenvalues). In either case, the mean image can be subtracted from the raw data matrix. As the data matrix grows in size, even computing the covariance matrix becomes intractable. Hence, it is more practical to directly apply SVD to the data matrix. Also, SVD is known to be computationally fast and reliable compared to eigen decomposition.

The computational complexity of the most efficient SVD algorithm, known as R-bidiagonalization-based SVD, for a $M \times N$ matrix is $C_{SVD} = O(k_1 M^2 N + k_2 N^3)$ [8]. In vision applications, we often use PCA to obtain basis images; we compute eigenvectors of length WH. This corresponds to computing only the left singular vectors U and the singular values Σ of $A = U\Sigma V$, where A is the data matrix. For this case, $k_1 = 4$ and $k_2 = 13$ in C_{SVD}. Therefore, the computational cost for global PCA becomes:

$$C_G = O\Big(4(WH)^2 F + 13F^3\Big). \tag{1}$$

Now, let us consider applying blockwise PCA with the block sizes set to (B_w, B_h, B_f). The complexity of computing the subspace corresponding to each block is $O(4(B_w B_h)^2 B_f + 13B_f^3)$. Since we have $\frac{WHF}{B_w B_h B_f}$ blocks, the total cost is:

$$C_B = O\left(4WHFB_w B_h + \frac{13WHFB_f^2}{B_w B_h}\right). \tag{2}$$

Hence,

$$C_G - C_B \approx 4WHF(WH - B_w B_h) + 13F\left(F^2 - \frac{WH}{B_w B_h} B_f^2\right). \tag{3}$$

As we have $B_w \le W, B_h \le H, B_f \le F$, the first term of (3) is greater than 0. Additionally, when $(\frac{F}{B_f})^2 \ge \frac{WH}{B_w B_h}$, the second term of (3) also becomes greater than 0 and blockwise PCA is guaranteed to run faster than global PCA. If we use a cubic block, the speed of blockwise PCA compared to global PCA increases with the number of frames. When runtime is critical to us, we can always set the block size in the temporal domain (B_f) to be smaller.

3.2 Storage

If we use global PCA to represent the whole data set in a single k_G-dimensional subspace, the total amount of data we need to store will be

$$S_G = (k_G + 1)WH + k_G F, \tag{4}$$

where the first term is for the eigenvectors plus one mean vector and the second term is for the coefficients. The coefficients are the projections of each image in the subspace.

Now, if we apply blockwise PCA with (B_w, B_h, B_f) and construct subspaces for each block with the same number of dimensions, k_G, the total amount of storage needed is

$$S_B = (k_G + 1)WH\frac{F}{B_f} + k_G F\frac{WH}{B_w B_h}. \tag{5}$$

It is easy to see that, if the dimensionality of each block subspace is the same as the dimensionality of the subspace computed by global PCA, the total amount of storage will increase as the block size decreases. However, it is safe to expect that the blocks will have very low dimensionality compared to the complete data set; the power of using blocks lies in the fact that, even when the data has global nonlinearities, it can be well-approximated by linear models at a local level. Therefore, in general, the dimensionality of blocks is expected to be much lower than that of the complete data set.

As PCA extracts the optimal subspace of the data set in a least-square sense, we apply it to reduce the redundancy while preserving the total RMS error between the original data and the data reconstructed using the extracted subspace. As each eigenvalue represents the variance along its corresponding eigenvector, a good measure often used to predict the RMS error is the eigenratio defined as

$$R_k = \sum_i^k \frac{\lambda_i}{\sum_j^M \lambda_j},$$

where λ_i is the ith largest eigenvalue [7]. Now, if we determine each block's subspace dimensionality by thresholding this eigenratio allowing each block subspace to have a different dimensionality, the total amount of storage becomes:

$$\hat{S}_B = (\bar{k}_B + 1)WH\frac{F}{B_f} + \bar{k}_B F\frac{WH}{B_w B_h}, \tag{6}$$

where $\bar{k}_B = (\frac{WHF}{B_w B_h B_f})^{-1} \sum_i^{\frac{WHF}{B_w B_h B_f}} k_B^i$ is the average number of dimensions of all the block subspaces. Note that different measures, such

as eigengaps [5], can be used to adaptively determine the dimensionality of the subspace corresponding to each subspace.

From (4) and (6), we see that we can achieve better compression ($S_B \leq S_G$) when

$$\frac{\bar{k}_B}{k_G} \leq \left(1 + \frac{F}{WH}\right)\left(\frac{F}{B_f} + \frac{1}{B_w B_h}\right)^{-1}. \quad (7)$$

Here, we have neglected the storage of the mean vectors for simplicity.

Loosely speaking, we expect to obtain smaller storage with blockwise PCA compared to global PCA when the average number of dimensions is below some fraction of the dimensionality of the global subspace. In other words, when global PCA does a "great" job and produces a very compact subspace, it becomes hard for blockwise PCA to satisfy (7). A rare case where this happens is when the data is made of images of perfectly Lambertian objects lit from different directions without cast shadows. In this case, it is well-known that the underlying subspace is three-dimensional [11], [23]. This is, however, a very special and rare case where we don't benefit from applying blockwise PCA. Generally, visual data exhibits more complex (nonlinear) appearance variations over space and time. Blockwise PCA exploits the local linearity in such overall nonlinear data and, thus, is able to provide a better compression.

4 Clustered Blockwise PCA

Blockwise PCA enables us to apply PCA to a large data set faster and more importantly, with higher compression compared to global PCA. In addition, it gives us a convenient representation of the data as a set of subspaces, which can be used to exploit self-similarities that commonly occur within visual data. We can combine the subspaces of "similar" block which results in a further reduction in the total amount of storage. Note that such correlations in appearance can occur not only in the spatial domain but also in the temporal domain.

4.1 Distance Metric

We first need a metric to measure the proximity/distance between different subspaces. Linear subspaces can be represented by their orthogonal projection matrix, $P = VV^T$, where V is a matrix whose columns are the eigenvectors of the subspace. The distance between two subspaces can be measured by the difference between their corresponding projection matrices [8]:

$$\rho(\mathcal{X}, \mathcal{Y}) = \| P_{\mathcal{X}} - P_{\mathcal{Y}} \|_2, \quad (8)$$

where $P_{\mathcal{X}}$ and $P_{\mathcal{Y}}$ are the projection matrices of subspaces $\mathcal{X}$ and $\mathcal{Y}$, respectively. It can be shown that [26]:

$$\rho(\mathcal{X}, \mathcal{Y}) = \sin\theta_1, \quad (9)$$

where θ_1 is the first (largest) canonical angle. Canonical angles are the angles between all pairs of orthonormal vectors of the QR-decomposed eigenvector matrices of the two subspaces. Therefore, the largest canonical angle provides a rotation invariant distance between two subspaces. Thus, without computing the projection matrices, we are able to compute the distance ρ.

We threshold this distance measure to decide whether the two subspaces of interest should be merged. Note that we do not want to merge subspace $\mathcal{X}$ with $\mathcal{Y}$ when the dimensionality of $\mathcal{X}$ is smaller than $\mathcal{Y}$ since it will not result in a reduction in the total amount of storage.

4.2 Merging Subspaces

We use the algorithm developed by Hall et al. [10] to merge a pair of subspaces. Readers are referred to [10] for details.[1] Note that we do not need to go back to the raw data to merge subspaces.

1. Unlike the original algorithm, we do not merge the mean vectors. We keep them individually associated with each block.

For each block, we compute ρ between its subspace and all other subspaces and store the block numbers whose subspaces fall below a distance threshold. Although this is a greedy approach with $O(n^2)$, where n is the number of blocks, since we do not evaluate the distance to a subspace that is already marked because it is close to another subspace, its average computational cost is much lower in practice. After listing all the subspaces that are close to each other in terms of distance ρ, we merge each pair one-by-one with the subspace merging algorithm in [10].

Clustering and merging the block subspaces results in a reduction in the number of subspaces, which results in a decrease in the first term of (6). Note, however, that we will introduce a small overhead in storage to keep record of the correspondence between a subspace and a block. This can be kept as n integers where n is the number of blocks.

When a new set of images are added to the data, once the number of these images becomes equal to the temporal size of the block, one can apply PCA to the blocks within this new data set and merge the subspaces that are close to existing subspaces. In this way, the necessary storage will not increase linearly to the size of the added data and correlation of local visual events can be exploited as new data is added. However, if an existing subspace is merged with a subspace computed from a new data block, the projection of the existing data block should be updated with the projection in the newly merged subspace. This will introduce a computational overhead. However, this recomputation of projection is computationally cheap which involves only matrix multiplication.

5 Experiments with Real Data

When we apply blockwise PCA and clustered blockwise PCA to any data, we need to decide the size of the blocks. The block size which produces the highest compression depends on the type of the data. As a result, determining the optimal spatial/temporal block size will involve trying different block sizes and computing the reconstruction error for each of the cases. Note that the reconstruction error cannot be precisely predicted when computing the subspaces; eigen ratio is only a rough indication of the preserved energy. Since the spatial and temporal block size can have an exponential number of combinations, we will need to choose some predetermined block sizes as candidates. There are several facts we can take into consideration when we specify the block size or the candidate block sizes when automatically determining it with this try and reconstruct algorithm.

The spatial block size (B_w and B_h) should be determined by examining the spatial variation within the images. Ideally, we should isolate high-frequency variations in individual blocks, so that they do not creep into the blocks with low-frequency variations. The temporal block size (B_f) mainly effects the subspace clustering stage. If the spatial appearance does not change in the data and other attributes such as lighting vary along the temporal axis, a tube-type blockwise/clustered blockwise PCA (B_f = number of frames) provides the highest compression. Ideally, the temporal block size should be matched with the temporal visual variation frequency. These factors can be used to select a block size and avoid trying all block sizes.

5.1 Example 1: Image-Based Relighting

Image-based relighting [6], [17], [18] aims to synthesize a "relighted" scene image from images of the same scene captured under various lighting conditions. As image-based relighting is based on interpolation, it suffers from the same problem as image-based rendering; the sampling of the acquired images with respect to lighting has to be dense enough to satisfy the Nyquist condition. As a result, the input data set is typically very large and compression becomes crucial.

The top left image in Fig. 2 shows one image from a sequence of 1,440 images of a scene containing various types of object with different material properties, which was lit by a light source from different directions. This data was obtained from the CAVE Laboratory at Columbia University. The image size is 320×240.

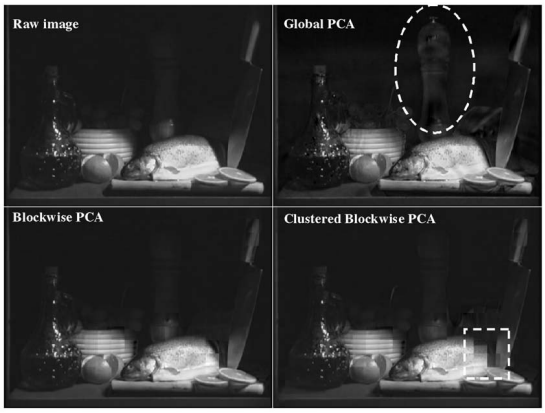

Fig. 2. Comparison between an original image and images reconstructed using subspaces obtained by global PCA, blockwise PCA and clustered blockwise PCA. While blockwise PCA, and clustered blockwise PCA preserve the global appearance with respect to the lighting direction, global PCA fails to do so (see text).

Table 1 shows the results of applying global, blockwise, and clustered blockwise PCA to this data set. Because we were not able to directly apply global PCA to this large data set, we applied it to 120 images at a time and applied SVD on the obtained set of eigenvectors scaled by their eigenvalues. Because we could not even apply this approximate version of global PCA to the whole sequence, we did this for the first 720 images and the last 720 images, separately. We did not have enough memory to store the high-dimensional subspace needed to reduce the RMS error of global PCA below the value in Table 1; the dimensionality turned out to be around 500 for an RMS error even as high as three gray levels. For reasons mentioned in the previous section, we applied a tube-type blockwise/clustered blockwise PCA to this sequence (block size = $16 \times 16 \times 1,440$).

The results show that clustered blockwise PCA clearly outperforms global PCA. This is an example where clustered blockwise PCA enables us to apply PCA to a data set that is too large for global PCA to handle. Furthermore, it results in better compression.

We should mention that, although the RMS errors for the different cases are similar, the errors in the reconstructed images have different characteristics. While global PCA achieves compression by cutting off the high frequencies in the image, blockwise and clustered blockwise PCA achieve compression by dropping the high-frequency variations within blocks. Therefore, blockwise and clustered blockwise PCA tend to preserve the global appearance due to the lighting while global PCA tends to compromise it. Fig. 2 shows a side-by-side comparison of reconstructed images. In the dashed circle in Fig. 2, we see that the appearance of the pepper dispenser is not consistent with the lighting and makes the image look odd. On the other hand, within the dashed box, we see that the shadow cast by the knife onto the fish is blurred and thus blocky in the image reconstructed by clustered blockwise PCA. However, this shadow remains consistent with the lighting direction, which is very important for relighting applications.

TABLE 1
Comparison between Global PCA, Blockwise PCA, and Clustered Blockwise PCA Applied on 1,440 Images of a Static Scene with Various Objects Lit by a Light Source from Different Directions

	Raw	G-PCA	B-PCA	CB-PCA
Storage(MB)	437.7	127.2	24.4	14.9
Dimensionality	–	109	12.1	3.8
# of subspaces	–	2	300	155
RMS Error	0.0	7.5	3.2	4.6
Compression Ratio	**1.0**	**3.4**	**18.0**	**29.3**

Blockwise PCA and clustered blockwise PCA outperform global PCA. In particular, clustered blockwise PCA gives an improvement in compression of a factor of nine over global PCA.

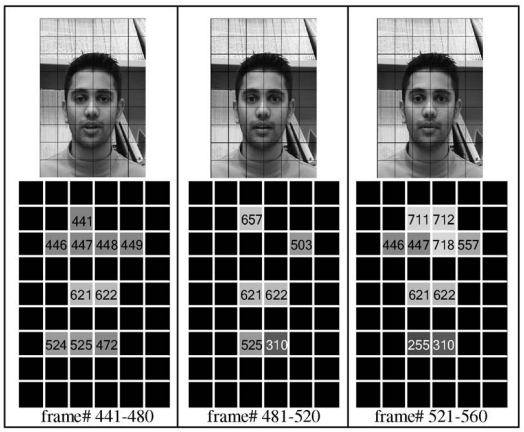

Fig. 3. Results of clustering subspaces corresponding to blocks (blocksize = $40 \times 40 \times 40$) for the talking head data. Three slices of blocks are shown here. The first image in each block slice is shown in the first row (the grid shows the spatial blocks). The second row shows the clustering results; the numbers are those of the subspaces that remain after subspace merging. The black blocks (without block numbers) were detected as static blocks. All static regions in the sequence were successfully detected and the moving regions are efficiently represented with a small number of subspaces, resulting in a high-compression ratio.

5.2 Example 2: Talking Head

Global PCA does not work well on image sequences where objects in the scene are in motion; in such cases, the data lacks global linearity. However, there is considerable interest in applying subspace analysis on such types of data. For instance, image sequences of a person's face while talking would be of high interest for analysis in computer vision. The top row of Fig. 3 shows three frames of a sequence of 640 images of a person talking in front of a camera. The image size is 240×360. It is expected that the data matrix for this type of data becomes full-rank and we cannot reduce the dimensionality through global PCA. On the other hand, with clustered blockwise PCA, we are able to not only exploit the local linearity and the spatio-temporal appearance correlations, but also take advantage of the regions that remain static over time.

We applied blockwise/clustered blockwise PCA with blocksize $40 \times 40 \times 40$ to this data set. While applying blockwise PCA, we can easily measure how much variation each block contains. For instance, as each eigenvector is 1,600 pixels long, on average, each element in the eigenvectors has $\frac{1}{\sqrt{1,600}} = \frac{1}{40}$ gray levels to contribute to each pixel's variation. Thus, we can use the value obtained by multiplying the first eigenvalue with $\frac{1}{40}$ as a rough indicator of how much the appearance varies within a block. This value indicates the variance of each pixel along the first eigenvector. We thresholded this value with 1.0 and all the blocks that fell below the threshold were considered to be static; only their mean vectors were stored. Table 2 shows the results of applying blockwise/clustered blockwise PCA to the talking head video. We applied the snapshot method [25] to find the the global subspace that gives a three gray levels RMS error. Since the dimensionality in this case was extremely high, we did not store the eigenvector matrix on local memory. Thus, we could not reconstruct images from the global subspace. Clearly, blockwise PCA and clustered blockwise PCA outperform global PCA in terms of compression. Also, while we had to run a greedy algorithm to construct the covariance matrix for global PCA (which took several hours), blockwise PCA and subsequent clustering took less than 2 minutes each.

6 Clustered Subspaces as a Representation

Fig. 3 shows the clustering results for three slices of blocks corresponding to frames 441 to 480, 481 to 520, and 521 to 560, for

TABLE 2
Comparison between the Results of Applying Global PCA, Blockwise PCA, and Clustered Blockwise PCA on an Image Sequence of a Person Talking

	Raw	G-PCA	B-PCA	CB-PCA
Storage(MB)	217.6	179.4	36.3	23.1
Dimensionality	–	586	18.9	16.0
# of Eigenspaces	–	1	256	172
RMS Error	0.0	3	3.5	4.1
Compression Ratio	**1.0**	**1.2**	**6.0**	**9.4**

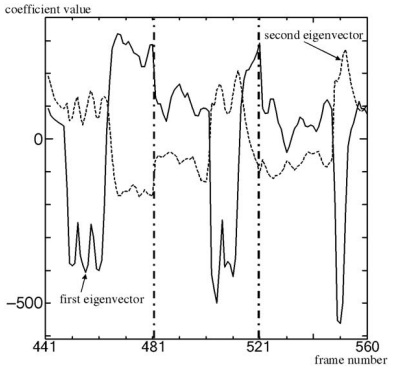

Fig. 4. Plots of the coefficients related to the first and second eigenvectors of subspace number 621, plotted for frame 421 to 560 of the talking head sequence. We see the strong periodicity of the coefficient signatures. This is due to the blinking of the right eye of the person in the sequence. This simple example illustrates why the representation produced by clustered blockwise PCA can prove very useful in a variety of visual tasks.

the talking head sequence. The first frames of these block slices are shown in the first row with their clustered block numbers in the second row. Black regions without block numbers indicate the blocks that were detected as static. As can be seen, only the moving parts, such as eyes and lips, were extracted and represented as subspaces. We can clearly see that clustered blockwise PCA identifies similar visual variations and represents them using a smaller number of subspaces. The eyes are represented in two subspaces, one for each eye. Also, note how the subspaces corresponding to block number 310, 446, 447, and 525 are reused. These blocks appear in frames earlier than 441. This shows that variations in the data are efficiently reused which results in a high-compression ratio.

From these clustering results, we can see that clustered blockwise PCA not only exploits the local linearity and spatio-temporal correlations to achieve higher compression, but also provides us with a representation that can be used for further analysis of the data. For instance, Fig. 4 shows plots (from frame 441 to frame 580) of the coefficients corresponding to the first two eigenvectors of subspace number 621 (which has three dimensions). We can clearly see the periodicity of the coefficient signatures. This results from the blinking of the right eye.

7 Conclusion

We introduced clustered blockwise PCA for representing visual data. Clustered blockwise PCA not only takes advantage of the fact that typical visual data contains localized variations, but also exploits the spatio-temporal correlations within the data. We have shown that clustered blockwise PCA achieves higher efficiency not only in terms of storage, but also computational cost. As a result, we have successfully scaled PCA to be applied to large problems without loosing any of the inherent advantageous properties of PCA itself.

In future work, we will investigate automatic methods for estimating the optimal spatial and temporal block size and/or allowing it to vary across the data volume. This remains an open problem which hinges upon the availability of novel practical algorithms.

Acknowledgments

This work was partly supported by US National Science Foundation ITR Award IIS-00-85864 and US National Science Foundation ITR Award CCR-03-12690.

References

[1] S. Avidan, "EigenSegments: A Spatio-Temporal Decomposition of an Ensemble of Images," *Proc. European Conf. Computer Vision,* vol. 3, pp. 747-758, 2002.
[2] C.M. Bishop, "Bayesian PCA," *Advances in Neural Information Processing Systems,* vol. 11, pp. 382-388, 1999.
[3] J.R. Bunch and C.P. Nielsen, "Updating the Singular Value Decomposition," *Numerische Mathematik,* vol. 31, pp. 111-129, 1978.
[4] S. Chandrasekaran, B.S. Manjunath, Y.F. Wang, J. Winkler, and H. Zhang, "An Eigenspace Update Algorithm for Image Analysis," *Graphical Models and Image Processing,* vol. 59, no. 5, pp. 321-332, 1997.
[5] F.R.K. Chung, *Spectral Graph Theory.* Am. Math. Soc., 1997.
[6] P. Debevec, T. Hawkins, C. Tchou, H.-P. Duiker, W. Sarokin, and M. Sagar, "Acquiring the Reflectance Field of a Human Face," *Proc. ACM SIGGRAPH 2000,* pp. 145-156, July 2000.
[7] K. Fukunaga, *Introduction to Statistical Pattern Recognition.* Academic Press, 1990.
[8] G.H. Golub and C.F. Van Loan, *Matrix Computation.* Johns Hopkins Univ. Press, 1996.
[9] M. Gu and S.C. Eisenstat, "A Stable and Efficient Algorithm for the Rank-One Modification of the Symmetric Eigenproblem," *SIAM J. Matrix Analysis and Application,* vol. 15, no. 4, pp. 1266-1276, 1994.
[10] P. Hall, D. Marshall, and R. Martin, "Merging and Splitting Eigenspace Models," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 22, no. 9, pp. 1042-1049, Sept. 2000.
[11] P. Hallinan, "A Low-Dimensional Representation of Human Faces for Arbitrary Lighting Conditions," *Proc. Computer Vision and Pattern Recognition,* pp. 995-999, 1994.
[12] H. Hotelling, "Analysis of a Complex Of Statistical Variables into Principal Components," *J. Educational Psychology,* vol. 24, pp. 417-441, 1933.
[13] T. Jebara, K. Russell, and A. Pentland, "Mixtures of Eigenfeatures for Real-Time Structure from Texture," *Proc. Int'l Conf. Computer Vision,* pp. 128-135, 1998.
[14] I.T. Jolliffe, *Principal Component Analysis.* New York: Springer-Verlag, 1986.
[15] A. Leonardis, H. Bischof, and J. Maver, "Multiple Eigenspaces," *Pattern Recognition,* vol. 35, pp. 2613-2627, 2002.
[16] A. Levin and A. Shashua, "Principal Component Analysis Over Continuous Subspaces and Intersection of Half-Spaces," *Proc. European Conf. Computer Vision,* vol. 3, pp. 635-650, 2002.
[17] Z. Lin, T.-T. Wong, and H.-Y. Shum, "Relighting with the Reflected Irradiance Field: Representation, Sampling and Reconstruction," *Int'l J. Computer Vision,* vol. 49, nos. 2-3, pp. 229-246, 2002.
[18] W. Matusik, H. Pfister, A. Ngan, P. Beardsley, R. Ziegler, and L. McMillan, "Image-Based 3D Photography Using Opacity Hulls," *ACM Trans. Graphics,* vol. 21, no. 3, pp. 427-437, 2002.
[19] B. Moghaddam and A. Pentland, "Probalistic Visual Learning for Object Detection," *Proc. Int'l Conf. Computer Vision,* pp. 786-793, 1995.
[20] H. Murakami and B.V.K.V. Kumar, "Efficient, Numerically Stabilized Rank-One Eigenstructure Updating," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 4, no. 5, pp. 511-515, 1982.
[21] H. Murase and M. Lindenbaum, "Spatial Temporal Adaptive Method for Partial Eigenstructure Decomposition of Large Image Matrices," *IEEE Trans. Image Processing,* vol. 4, no. 5, pp. 620-629, 1995.
[22] K. Pearson, "On Lines and Planes of Closest Fit to Systems of Points in Space," *The London, Edinburgh, and Dublin Philosophical Magazine and J. Science,* six series 2, pp. 559-572, 1901.
[23] A. Shashua, "On Photometric Issues in 3D Visual Recognition from a Single 2D Image," *Int'l J. Computer Vision,* vol. 21, no. 1, pp. 99-122, 1997.
[24] E.P. Simoncelli and B.A. Olshausen, "Natural Image Statistics and Neural Representation," *Ann. Rev. Neuroscience,* vol. 24 pp. 1193-1216, 2001.
[25] L. Sirovich, "Turbulence and the Dynamics of Coherent Structures," *Quarterly Applied Math.,* vol. 45, pp. 561-571, 1987.
[26] G.W. Stewart and J.-G. Sun, *Matrix Perturbation Theory.* Academic Press, 1990.
[27] L. Zhao and Y.-H. Yang, "Mosaic Image Method: A Local and Global Method," *Pattern Recognition,* vol. 32, no. 8, pp. 1421-1434, 1997.