Efficient Recovery of Low-dimensional Structure from High-dimensional Data

Shyjan Mahamud Martial Hebert
Dept. of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

Many modeling tasks in computer vision. e.g. structure from motion, shape/reflectance from shading, filter synthesis have a low-dimensional intrinsic structure even though the dimension of the input data can be relatively large. We propose a simple but surprisingly effective iterative randomized algorithm that drastically cuts down the time required for recovering the intrinsic structure. The computational cost depends only on the intrinsic dimension of the structure of the task. It is based on the recently proposed Cascade Basis Reduction (CBR) algorithm that was developed in the context of steerable filters. A key feature of our algorithm compared with CBR is that an arbitrary apriori basis for the task is not required. This allows us to extend the applicability of the algorithm to tasks beyond steerable filters such as structure from motion. We prove the convergence for the new algorithm. In practice the new algorithm is much faster than CBR for the same modeling error. We demonstrate this speed-up for the construction of a steerable basis for Gabor filters. We also demonstrate the generality of the new algorithm by applying it to to an example from structure from motion without missing data.

1 Introduction

Many tasks in computer vision involve the determination of a low-dimensional structure from high-dimensional data. For example, in structure from motion [6], we are provided with a sequence of 2D images in which interest points have been located and tracked. The task is to recover the shape of one or many objects in the scene. The dimension of the input data which in this example is the number of frames in the sequence times the number of interest points in a frame, is typically very high. Nevertheless, the dimension of the shape to be recovered is typically very small (for eg., it is atmost 3 for each rigid body in the scene under scaled orthographic projection). Methods such as factorization [6] require the determination of the SVD of the measurement matrix constructed from the frame sequence. Even though the shape that is to be recovered has low dimension, the direct computation of the SVD of the measurement matrix depends on the dimensions of the matrix hence does not scale well.

Another example is in the construction of steerable filters [5, 3, 4]. Given a filter $\phi(x, y)$ and a set of transformations T, the task is to find a set of basis filters $\Phi = \{\phi_1, \ldots, \phi_k\}$ such that the response of the original filter ϕ under any transformation from T is some linear combination of the responses of the basis filters in Φ. For reducing the run-time cost, we would like to keep the number of basis filters in Φ as small as possible. Analytical solutions are known only for some filters under a restricted set of transformations. One straight-forward technique that was recently proposed for numerically constructing Φ [4, 5] is to sample ϕ under various transformations from T and then sample the transformed filter spatially upto some required resolution. The most significant left eigenvectors of the SVD of the matrix of such samples provide a basis set for steering ϕ. The cost of directly computing such an SVD can be high if the set of transformations over which the filter is to be steered and the required resolution are high-dimensional.

Both these examples along with other tasks in vision can be put in a common framework that highlights the common features that are shared across all such tasks. Consider a task where the input measurement data can be looked upon as columns of an $m \times n$ matrix M. For example, in the case of structure from motion, column i represents the coordinates of the interest points in frame i. In the case of steerable filter synthesis, each column is a transformed version of the filter to be steered.

We consider a task to have a low-dimensional structure when the corresponding measurement matrix can

be factored into two low-rank matrices $M = BC$, where B, C^T are both $m \times r$ matrices with $r \ll m, n$. Here, r is a measure of the dimension of the structure of the task. Thus, in the example of structure from motion, B is the "shape" matrix that describe the rigid structure of an object (or objects) in the scene and C is the "motion" matrix which is related to the camera parameters for each frame (for simplicity we consider scaled orthographic projection). In the case of steerable filter synthesis, the columns of B are the set of basis filters that steer the original filter and each column of C describe the "steering coefficients" that steer the basis set to the transformed filter in the corresponding column of M. The factorization $M = BC$ can be found with an application of SVD $M = U\Sigma V^T$. Under noise-free conditions, the dimension r of the task is the number of non-zero singular values in Σ (with noise and modeling errors, r is taken to be the number of singular values above some threshold). Then M can be factorized with $B = U_r \Sigma_r^{1/2} A$ and $C = A^{-1} \Sigma_r^{1/2} V_r^T$. Here, U_r, Σ_r, V_r denote the sub-matrices that correspond to the top r singular values in Σ. A is an unknown invertible $r \times r$ transformation that has to be further constrained by the task at hand; for example, in structure from motion, rigidity constraints are imposed to determine A.

The computation of the SVD of M is bounded below by roughly the cubic of the minimum of m and n both of which can be large for the given task. However, the structure of the problem (as measured by r) might be much smaller. Hence, it would be preferable to have a method for computing the SVD of M whose cost is dominated primarily by only r, the actual dimension of the matrix. We propose such a method of factoring M that is based on a recent algorithm that was developed in the domain of steerable filter synthesis.

The Cascade Basis Reduction (CBR) algorithm [5] was proposed as a method for reducing the cost of computing the SVD of M in the case of steerable filter synthesis. In this method, an a priori basis matrix B for steering the original filter was assumed and improved upon. The number of a priori basis filters was low enough to make the computation of the SVD feasible. A limitation of the approach is in the choice of an appropriate a priori basis. A poor choice will result in an inadequate basis for steering the original filter. In [5], the choice for an a priori basis was arbitrary and did not depend on the actual filter that was being steered. Also, while the number of a priori basis filters was low, it was still typically much higher than the actual dimension of the structure of the task at hand.

In this paper, we show that the CBR scheme can be extended in such a way that we do not have to make an arbitrary choice for an a apriori basis. It is shown that a natural choice for an a priori basis can be directly drawn from the particular task at hand. The number of a priori basis needed is typically far smaller than the number required if an arbitrary choice is made. Also, in contrast with CBR, the a priori basis can be improved upon monotonically as discussed and proved in § 2.2. As a result of the natural choice for the a priori basis, unlike CBR, the resulting algorithm is not limited to the task of steerable filter synthesis. It can be used for any task that has a low-dimensional intrinsic structure. We demonstrate its applicability with two examples. In § 3.1, we apply the algorithm for the task of steering a gabor filter over the 4-parameter 2D linear transform. We show that a significant reduction in the number of a priori basis required can be achieved as compared with CBR for the same modeling error. In § 3.2, we demonstrate the use of the new algorithm for the task of structure from motion without missing data, resulting in a large speedup compared to the use of the traditional SVD. In § 4 we also discuss its applicability as a component for other vision tasks such as the recovery of shape/reflectance from shading.

2 Iterative Randomized CBR

In this section, we start with a description of the original CBR algorithm. We then motivate and introduce the extension where the choice of an a priori basis is natural. We describe how we can monotonically improve this basis. We next prove the convergence of the algorithm. We finish the section with a demonstration of the effectiveness of the algorithm on a matrix with random entries where we can explicitly control the rank of the matrix.

2.1 Cascade Basis Reduction (CBR)

The original CBR algorithm was developed in the context of steerable filter synthesis. We recast the algorithm in terms of factoring the measurement matrix $M = BC$. As in the introduction, let M be an $m \times n$ matrix and B, C^T both be $m \times r$ matrices with $r \ll m, n$. Assume for now that we can choose an a priori set of k, m-dimensional basis vectors that spans the columns of M with $k \ll m, n$. Hence, it is necessary that $k \geq r$. Let B' be an $m \times k$ matrix whose columns are the a priori basis. Since we do not know B yet, nothing can be said about the relationship between B and B' a priori. Ideally, we would like the columns of B to be spanned by the columns of B'. Assuming for now, that the columns of B' span B, CBR gives an efficient algorithm for computing B

from M and B' as follows. Since the columns of B' span M, we can express $M = B'C'$ where the coefficient matrix C' can be determined by applying the pseudo-inverse of B' on M. Finding the SVD of M is then reduced to finding the SVD of $B'C'$, which can be computed efficiently by a sequence of two SVDs, one involving B' and the other involving C' as follows :

$$\begin{aligned} M &= B'C' \\ &= (U_{B'}\Sigma_{B'}V_{B'}^T)C' \quad (1) \\ &= U_{B'}C'' \\ &= U_{B'}(U_{C''}\Sigma_{C''}V_{C''}^T) \quad (2) \\ &= (U_{B'}U_{C''})\Sigma_{C''}V_{C''}^T \\ &= U_M\Sigma_M V_M^T \end{aligned}$$

where, $U_M = U_{B'}U_{C''}, \Sigma_M = \Sigma_{C''}$ and $V_M = V_{C''}$. Since B' is an $m \times k$ matrix with $k \ll m$, the SVD in eq 1 can be obtained by solving for the SVD for $B'^T B'$ which is only a $k \times k$ matrix, as follows. $B' = U'\Sigma'V'^T$ where $B'^T B' = V'\Sigma^2 V'^T$ and $U' = B'V'/\|B'V\|$. Similarly the SVD in eq 2 can be obtained by solving for the SVD for $C''C''^T$ which is also a $k \times k$ matrix. Hence, a significant reduction in the cost of computing the SVD for M can be achieved if $k \ll m, n$. From the SVD for M the factorization $M = BC$ can be achieved as discussed in the introduction. In practice, M can only be approximated (say in the least squares sense) by a low-dimensional factorization $M \approx BC$. In such a case, in the discussion above all the equalities above are replaced by least squares approximations.

2.2 The New Algorithm

In the case of steerable filter synthesis for which the CBR was developed, the only constraint made on the choice of the a priori basis B' was that the basis were few in number and was itself steerable. Apart from this, no constraint was imposed by the actual filter to be steered. Hence, with respect to the filter to be steered, the choice is arbitrary. In this section, we propose a simple and yet surprisingly effective choice for the a priori basis that is directly drawn from the measurement data. We then describe how the basis can be improved iteratively and prove its convergence in the next section.

Consider again the SVD of $M = U\Sigma V^T$. As discussed, the left-eigenvectors in U that correspond to the r most significant singular values in Σ are the ones that will be taken as the basis set B that will span the columns of M. It is well-known that each left-eigenvector in U is spanned by the columns of M. As a simple generalization, the subspace spanned by any subset S of the left-eigenvectors is also spanned by the columns in M. This implies that if S has dimension k, then there exists a subset of columns of M that is k-dimensional that spans S. This observation gives us a simple method for choosing a natural a priori basis that can approximate the columns of M. We can sample the columns of M a number of times and use the resulting samples to form the a priori basis B'. The hope is that a large enough number of samples will have enough independent columns to span S. The number of samples required to span S is dependent on two factors. The first is the dimension k of S. The lower the dimension k, the lower the expected number of samples required from the columns of M that will span S. However, a more important factor is the magnitude (and hence significance) of the singular values associated with the vectors in S. A high singular value associated with a given left-eigenvector $u \in S$ means that the columns of M will have on average, a large projection onto u. Hence, we make the important observation that the higher the sum of the singular values associated with the vectors in S (relative to the sum over all singular values), the more likely that a randomly drawn column from M will have a large projection lying in the subspace spanned by S. Hence, with high likelihood, in a sample of $l \times k$ columns of M, where l is small, there will be at least k sampled columns that are independent and will span all of S.

Once we have chosen a sample number of columns as the a priori basis B', we approximate M with B', giving the coefficient matrix C'. We can think of approximating M with B' as the process of modeling the columns in M with a basis B' giving rise to a coefficient matrix C'. Applying the CBR algorithm on $B'C'$ then gives us feedback on which linear combinations of the basis vectors in B' (the most significant vectors in $U_M = (U_{B'}U_{C''})$ in equation 2) span the columns of M. The k most significant linear combinations (i.e., the k top vectors in U_M) are chosen as the basis for approximating the columns in M.

It is important to realize that it is not necessary for the distribution of the samples in the a priori basis B' to match the distribution of the columns in M. If in a given trial, due to a bad draw, the distribution of the samples in B' does not resemble the actual distribution of the column vectors in M, as along as there exists some k linear combinations of the chosen samples in B' that span S, we will be able to recover a basis that spans S. This is due to the crucial feedback provided by the coefficient matrix C' which can be thought of as offsetting the effects of poor sampling.

It can happen that in a given trial, the k-dimensional subspace S is not fully spanned by the samples in B'. In such a case, the top k-dimensional

initialize B' with s columns sampled from M
do
  approximate M with $B' : M \approx B'C'$
  apply CBR on $B'C'$ and obtain $U\Sigma V^T$
  let $S' = k$ most significant vectors from U
  reset B' with :
    the k columns from S'
    $s-k$ new columns sampled from M
until $(\sum_{i=1}^{k} \sigma_i^2)$ converges
return S'

Figure 1: Pseudo-code for the iterative randomized extension of CBR: s is the total number of columns sampled from F per iteration, k is the number of significant basis vectors to be returned, the σ_i's are the singular values from Σ arranged in decreasing order.

subspace S' recovered from B' through the CBR algorithm is not optimal, i.e., there is only a partial overlap between S' and the desired subspace S. Nevertheless, we need not throw away the information contained in the partial overlap. We can improve the recovered subspace S' by augmenting it with information from additional columns sampled from M and iterating the algorithm. More precisely, we create a new a priori basis B'' where k of the columns are the k vectors spanning S' (i.e., the top k vectors from B') and the rest of the columns of B'' are additional columns sampled from M. We then iterate the CBR algorithm and recover a new k-dimensional subspace S''. Since the new B'' contains the old k-dimensional subspace S', intuitively the new subspace S'' that is recovered should be at least as good as S'. We prove this more formally in the following theorem. Hence by successive application of the algorithm, we can expect to monotonically improve the k-dimensional subspace that approximates M.

Figure 1 gives the pseudo-code for the iterative algorithm. It is useful to interpret what goes on in each iteration as follows. At the start of each iteration, new information is added to the current estimate for S', by adding new samples drawn from M. At the end of the iteration, some information is thrown out ($s-k$ least significant columns of U). The following theorem proves that in effect the "net" information gained at each iteration is non-negative.

Theorem 1 *Let S' be the $m \times k$ matrix whose columns are the top k-dimensional basis set returned by the new algorithm at some iteration i. Let $\sigma_1, \ldots, \sigma_k$ be the corresponding singular values. Let $B' = [S'|T]$ be the new basis set at the start of iteration $i+1$, where T is an $m \times (s-k)$ matrix whose columns are new samples from the columns of M. Then, if $\sigma'_1, \ldots, \sigma'_k$ are the top k singular values of the new approximation $M \approx B'C'$:*

$$\sum_{j=1}^{k} \sigma_j'^2 \geq \sum_{j=1}^{k} \sigma_j^2$$

Furthermore, the singular values are bounded above by those for M, and hence convergence follows.

Proof. Let $U_1 = [u_1, \ldots, u_k]$ be the matrix of orthonormal left-eigenvectors u_i spanning S' that is returned at iteration i. Then the best k-dimensional approximation of M at iteration i is $U_1\Sigma_1 V_1^T$ where $\Sigma_1 = \mathrm{Diag}(\sigma_1, \ldots, \sigma_k)$ and V_1 is the corresponding set of right-eigenvectors. The span of the new basis $B' = [S'|T]$ is the sum of the span of S' (which is U_1) and the span of T. Let $[U_1, U_2]$ be an orthonormal basis spanning B' such that we choose the first k basis to be U_1 (hence, U_2 spans any remaining independent dimensions that are introduced by T). Since, U_1 and U_2 are subspaces that are orthogonal to each other by construction, the new approximation $M \approx B'C'$ at iteration $i+1$ is the sum of the projections of M onto the subspaces spanned by U_1 and U_2. The projection of M onto U_1 is simply $U_1\Sigma_1 V_1^T$. Let the projection of M onto U_2 be given by U_2C_2 where C_2 is the matrix of coefficients for projecting the columns of M onto U_2. The SVD of $U_2C_2 = U_2'\Sigma_2 V_2^T$ gives us a new set of basis vectors U_2' that is still spanned by U_2 (since U_2 is the column space of U_2C_2) and hence U_2' is still orthogonal to U_1. As noted above, the projection of M onto the orthogonal subspaces U_1 and U_2 (and hence U_2') denoted by P is the sum of these two projections :

$$P = U_1\Sigma_1 V_1^T + U_2'\Sigma_2 V_2^T$$

Since V_1 and V_2 are the right-eigenvectors of their respective SVD's, the vectors from each set are orthonormal to the other vectors from the same set, i.e., $V_1^T V_1 = I, V_2^T V_2 = I$. However, in general a vector from V_1 need not be orthogonal to vectors from V_2 since they are a result of independent SVD's. Let $R = V_1^T V_2$ be the correlation matrix between the subspaces V_1 and V_2. Then, taking PP^T above, we get :

$$PP^T = U_1\Sigma_1^2 U_1^T + U_2'\Sigma_2^2 U_2'^T + U_1\Sigma_1 R\Sigma_2 U_2'^T + U_2'\Sigma_2 R^T\Sigma_1 U_1^T$$

It can be verified from the expression above that since the subspace U_1 is orthogonal to the subspace U_2', we get

$$\mathrm{tr}(U_1^T PP^T U_1) = \sum_{j=1}^{k} \sigma_j^2 \qquad (3)$$

where tr stands for the trace of a matrix and the σ_j's are the singular values in Σ_1 in descending order.

Now, at iteration $i+1$, M is projected onto the combined subspace spanned by U_1, U_2 to give another expression for the same projection $P = U'\Sigma' V'^T$. Taking PP^T

again, we get $PP^T = U'\Sigma U'^T$. Since PP^T is symmetric, we can apply a generalization of the *Poincare separation theorem* [2], which states that (adapting the statement for the real matrix PP^T) :

$$\max_{U^TU=I} \mathrm{tr}(U^TPP^TU) = \sum_{j=1}^{k} \sigma_j'^2 \quad (4)$$

where U is any matrix with k orthonormal vectors. Since U_1 is a particular matrix with k orthonormal vectors, we get after combining equations 3 and 4 :

$$\sum_{j=1}^{k} \sigma_j'^2 \geq \sum_{j=1}^{k} \sigma_j^2$$

Convergence follows from the fact that during all iterations, P is a projection of the measurement matrix M and hence the sum of all singular values squared of P are bounded above by the sum of the singular values squared of M. □

In the proof, there is no improvement at a given iteration if the new samples in T are already spanned by U_1. However, since sampling is uniform, if there exists samples not spanned by U_1, they will be spanned in future iterations with non-zero probability. Nothing in the proof gives a hint about the speed of convergence. We show below that in practice, the convergence is very fast for problems with low-dimensional structure.

We illustrate the performance of the new algorithm for computing the SVD for a randomly constructed matrix whose rank can be explicitly controlled. We generated a 100×1000 random matrix R of rank 50 with entries in the range $[0.0, 1.0]$. We deliberately chose a large rank so that computing the SVD for the random matrix with the new algorithm would be expected to be a difficult test since it does not have a low-dimensional structure. As can be seen from Fig. 2, the singular values of R (top-most curve) decay gradually. In other words, the columns of R cannot be approximated well by a small number of basis vectors.

We choose the desired number of basis vectors k to be 10. The total number of columns sampled from R in each iteration was $s = 20$. Despite the fact that R does not have a low-dimensional structure $r = 50$, the singular values in the very first iteration is close to the actual values computed using an ordinary SVD on R. Successive iterations improve the first 10 singular values (the ones before the vertical line) monotonically as expected [1]. By iteration number 10, the singular values have converged. The computational cost for 10 iterations is dominated by 10 times the SVD for a 20×20 matrix (the pseudo-inverse required to approximate R is also obtained as a byproduct of the same SVD), whereas the full SVD is applied on a 100×100 matrix. SVD is roughly cubic in the dimension of the matrix, and hence we achieve a substantial speedup. On the other hand, the remaining singular values (after the vertical line in the figure) do not behave in any predictable manner as expected and on average do not improve their values. Thus even for a matrix for which there is no low-dimensional structure, the new algorithm is still able to recover the most significant basis vectors in a few number of iterations.

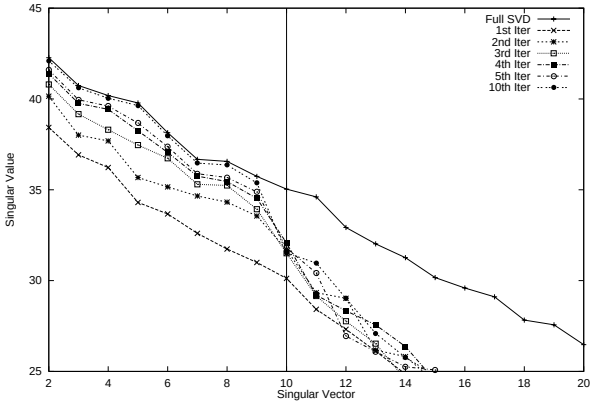

Figure 2: Singular Values vs. Singular vectors across iterations for a 100×1000 random matrix of rank 50. See text for details.

3 Results

3.1 Steerable Filters

The original CBR algorithm was developed in the context of steering filters. Here, we compare both the original and the new version for the example task of steering an odd-phase Gabor filter $\phi(x, y) = \sin(x/\sigma_x)\exp(-((x/\sigma_x)^2 + (y/\sigma_y)^2))$. The objective is to find a low-dimensional basis set $\Phi = \{\phi_1, \ldots, \phi_k\}$ that can steer the Gabor filter under the set of 4-parameter 2D linear transformations :

$$g \cdot \phi = \sum_{i=1}^{k} \alpha_i(g)\phi_i \quad (5)$$

where $g \cdot \phi$ denotes the application of the transformation g on ϕ. The coefficients α_i are the "steering" functions that depend on only the parameters describing the transformation g. The condition above is for exact steerability. In practice, for many of the filters and transformations of interest, the above condition

[1] the highest singular value (corresponding to the average of all the columns) is not shown since it is out of the range of the graph, however it shows the same behavior as the rest of the first 10 singular values

cannot be met exactly. In such cases, the goal is relaxed to find a basis Φ that will steer the original filter ϕ approximately. In the condition above, this means replacing the equality with a least squares approximation. Given a threshold for the least squares error, we find the minimum possible number of basis filters that will meet the threshold. In addition to the determination of Φ, we also need to determine the steering functions (the α_i's in equation 5).

The determination of the basis Φ is computed numerically by first constructing a measurement matrix M where each column corresponds to a gabor filter that has undergone a sample 2D transformation. The column is the resulting filter sampled to some fixed spatial resolution. In order to make a fair comparison, we use the same parameters as the ones used in [5]. The Gabor filter was sampled spatially over $[-1,1] \times [-1,1]$ with a resolution of 64×64 (hence the number of rows of M is 4096). As in [5] we parametrize the 2D linear transformation uniquely as : $A = R(\theta_2)S_x(s_x)S_y(s_y)R(\theta_1)$, where R denotes rotations and the S_x, S_y denote scaling in the x and y directions. The filter was steered over the following range of parameters : $\theta_1, \theta_2 \in [0, 2\pi)$ and $s_x, s_y \in [1, 5/3]$. A total of $22,500$ samples of the Gabor filter steered over the above range of parameters were used to form the columns of M. For the original CBR algorithm, 231 2D Legendre polynomials with a total degree upto 20 were used as the a priori basis B'. For the new algorithm, we used the number of desired basis vectors k to be 11 to remain compatible with the choice in [5] where sum of the first 11 singular values squared is 99.9% of the total. Note that we could have chosen k at run-time automatically by using a threshold on the sorted singular values. We ran two experiments, one with total number of samples $s = 231$ in each iteration, and another experiment in which the total number of samples $s = 30$. The rationale for the first experiment was to keep the number of a priori basis B' the same as that for the original algorithm. The total number sampled across all iterations is more than that used for the original algorithm, hence for a fair comparison, we report results only for the first iteration. For this task, the algorithm converges in the very first iteration itself. In the second experiment, we show that we get identical results as in the first experiment, despite using a far fewer number of total samples s. Again the algorithm converges in the first iteration itself.

Fig. 3 shows the results for the case $s = 231$. Four of the 11 basis filters corresponding to the top 11 singular values obtained after the first iteration are shown

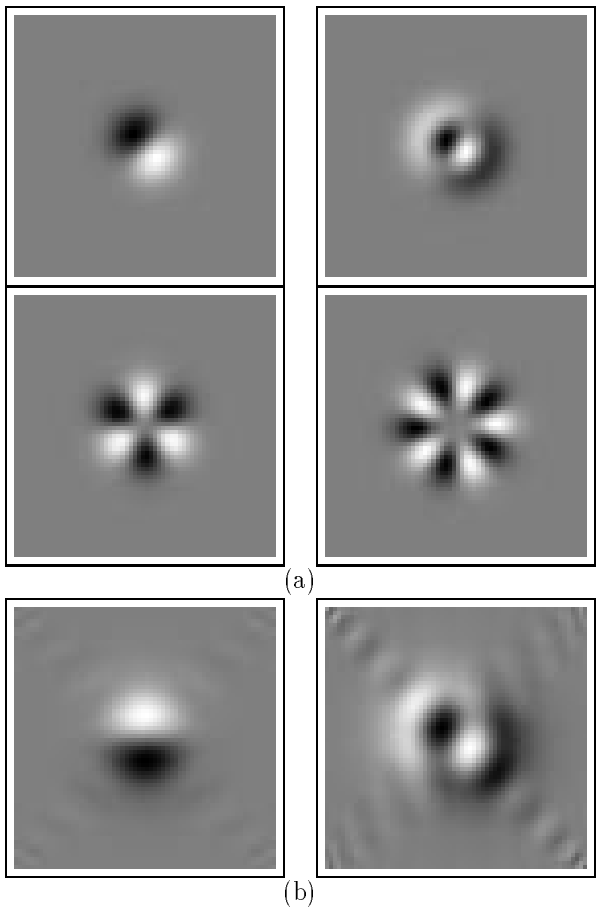

Figure 3: (a) Four of the 11 filters recovered by the new algorithm in the first iteration with $s = 231$. (b) Two of the 11 filters recovered by the original CBR using 231 2D Legendre polynomials

in Fig. 3 (a). We also show the first and third most significant basis filter returned by the original CBR algorithm in Fig. 3 (b). As can be seen from the singular value plots in Fig. 5, the results for the two algorithms are comparable. The original algorithm returns filters with slight ripples in the otherwise uniform region of the basis filters. This is due to the mismatch between the functional form of the Gabor filter and that of the Legendre polynomials. The mean squared error in approximating 1000 test gabor samples under various transformations was only 0.01% (expressed with respect to the squared norm of the gabor sample in each case). Fig. 4 (a) shows the results for the case $s = 30$ after the first iteration. Despite the much smaller number of a priori basis vectors used, the recovered filters are indistinguishable from that for the first experiment. This is also seen from the plot for the singular values. For comparison, we also ran the original CBR algorithm with a similar number of a

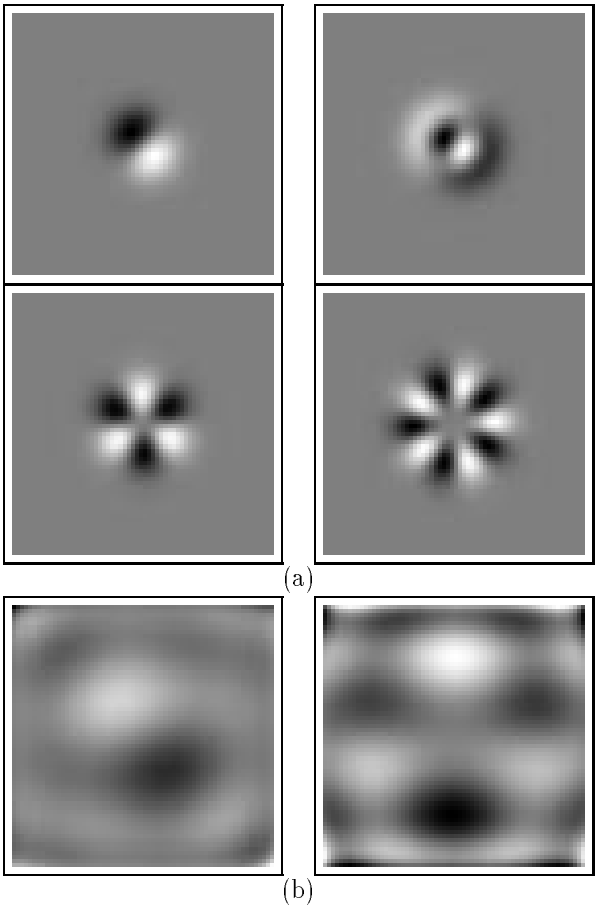

Figure 4: (a) Four of the 11 filters recovered by the new algorithm in the first iteration with $s = 30$. These and the other filters are identical to the ones recovered with $s = 231$ shown in Fig. 3 (a). (b) Two of the 11 filters recovered by the original CBR using 36, 2D Legendre polynomials. As can be seen the filters are of poor quality.

priori basis : to be conservative, we chose 36 Legendre polynomials of degrees upto 7. The recovered filters are of very poor quality as shown in Fig. 4 (b). This illustrates the fact that the choice of a natural a priori basis can allow us to reduce the number of a priori basis vectors used dramatically and hence also reduce the computational cost as compared to the case where we choose an arbitrary a priori basis.

In [5], the steering functions (α_i in equation 5) were derived through the analytic steering functions that are easily derived for the Legendre polynomials (this is the main reason for using these polynomials). For the new algorithm, a simple interpolation scheme is sufficient to steer the filter. For the interpolation, the coefficients that steer the basis filters to approximate 10000 gabor samples sampled under various transformation were recovered and stored in an interpolation

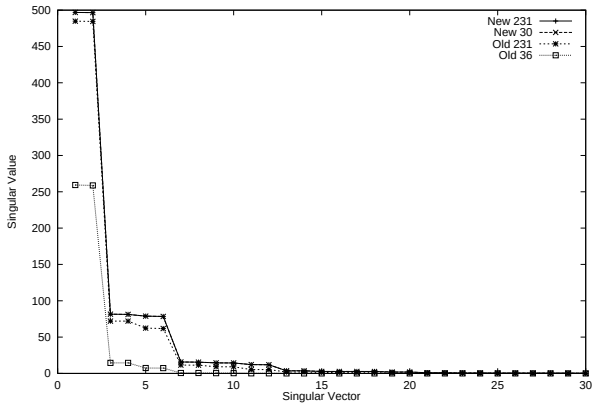

Figure 5: Singular values vs. singular vectors for the various experiments in steering the gabor filter. The plots for the new algorithm are shown for the first iteration and can be seen to be indistinguishable for $s = 231$ and $s = 30$. Both these plots in turn are similar to the plot for the original CBR that uses 231 2D Legendre polynomials. However, CBR with only 36 polynomials has much worse performance compared to all others, specifically with the new algorithm with $s = 30$.

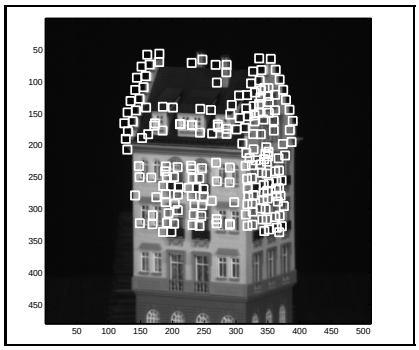

Figure 6: One frame from the hotel sequence

table. Given a new transformation over which the Gabor filter is to be steered, we obtain the steering coefficients corresponding to this transformation by using a 4-th order polynomial interpolation on the coefficients stored in the interpolation table. The mean squared error involved in using the interpolation scheme to steer the basis filters over a sample of 1000 test transformations as compared to actually steering the basis filters themselves was only 0.17% (again expressed with respect to the squared norm of the resulting gabor sample in each case).

3.2 Structure From Motion

We had claimed that the new algorithm can be used for tasks other than that for steering filters. We illustrate the use of the new algorithm as part of a structure from motion task without missing data. Fig. 6

shows one frame from a 181-frame 2D image sequence of a model hotel. In the whole sequence, 197 interest points have been located and tracked from frame to frame (see [6] for details). A 394×181 measurement matrix M is formed where each column contains the x and y coordinates of the interest points for each frame. We assume that all the points are visible in all the frames, i.e., there is no missing data. For an orthographic projection model for the camera, the measurement matrix can be decomposed into a shape S and motion C matrix : $M = SC$ where both S and C have rank 3 for each rigid body in the scene under noise-free conditions. The shape and motion matrix can be recovered using factorization [6], which uses the SVD of M to compute the best rank 3 approximation to M. Rigidity constraints are imposed on this rank 3 approximation to recover the S and C matrices. Here, we deal only with the efficient computation of the SVD. The imposition of the rigidity constraints and the rest of the factorization algorithm is exactly the same.

We applied the new algorithm with the desired number of basis vectors set to $k = 3$ and the total number of samples per iteration set to $s = 10$. The algorithm converges quickly in the first two iterations. The Frobenius norm of the residue between the best rank 3 approximation and M normalized with respect to the Frobenius norm of M was only 0.02%. The time for the two iterations was 0.54 secs compared with 20 secs for the ordinary SVD for M, hence resulting in a speedup of 37.

4 Conclusion

We have presented a new algorithm that can be used for the efficient recovery of the low-dimensional structure of a task even if the measurement data is high-dimensional. Convergence of the iterative algorithm was proven to be monotonic and also fast in practice. Unlike the original CBR algorithm from which it was derived, the algorithm is applicable to a wider variety of tasks in vision that have a low-dimensional structure. It should be noted that based on the evidence from the experiment on a random matrix in § 2.2, the algorithm performs well even if the matrix does not have a low-dimensional structure. However, it does not converge as fast as in the case when the task has a low-dimensional structure (see the example for steering the Gabor filter and the recovery of structure from motion). Another task for which the algorithm is applicable is the recovery of shape or reflectance from shading [1].

An important obstacle in applying the algorithm to tasks such as structure from motion or shape/reflectance from shading is the issue of missing data. Due to occlusion, some of the data can be unobservable. Missing data affects the algorithm in two places. First, when sampling the basis vectors for B' from the data, a newly sampled vector can contain missing data. One way of dealing with missing data is to set the missing components of the sampled vector to random values. The feedback from approximating the data with B' as well as the true values for the same component from other sampled vectors should utilize the random settings that were set in the right direction and disregard the others. The second place where missing data is an issue is in approximating the data with B' (see the pseudo-code in Fig. 1). For each vector v in the data, the coefficients for recovering the best least squares approximation for that vector is obtained by applying the pseudo-inverse of B' on v. One way of dealing with missing components in approximating v is to recompute the pseudo-inverse after first deleting the rows of B' that correspond to the missing components in v. Recomputation of the pseudo-inverse is cheap if the number of basis in B' is small (use the SVD of $B'^T B'$ where the appropriate rows corresponding to missing components in v have been removed). Also, if the data is such that there are only a few patterns of missing data, i.e. the pattern of which components are missing, then the number of pseudo-inverses to compute may not be prohibitive. In the future, we intend to investigate such schemes for handling missing data for large datasets.

Acknowledgements. Data for the structure from motion task was provided by the Modeling by Videotaping group in the Robotics Institute, Carnegie Mellon University.

References

[1] Epstein, R. and Yuille, A.L. and Belhumeur, P.N., "Learning Object Representations from Lighting Variations", In *ORCV*, pp 179, 1996.

[2] Horn, R.A. and Johnson, C.R. 1990. *Matrix Analysis*, Cambridge University Press, Cambridge, UK.

[3] Perona, P., "Deformable Kernels for Early Vision", *PAMI*, **17**(5):488–499, 1995.

[4] Perona, P., "Efficient Implementation of Deformable Filter Banks", Tech Rept. CNS-TR-97-04, California Institute of Technology, 1997.

[5] Teo, P.C. and Hel-Or, Y., "Design of Multi-Parameter Steerable Functions Using Cascade Basis Reduction", In *ICCV98*, 187–192, 1998.

[6] Tomasi, C. and Kanade, T., "Shape and Motion from Image Streams under Orthography: A Factorization Method", *IJCV*, **9**(2):137–154, 1992.