中央研究院
資訊科學研究所
Institute of Information Science, Academia Sinica • Taipei, Taiwan, ROC

TR-IIS-14-002

SIGNAL SEPARATION USING RE-WEIGHTED AND ADAPTIVE MORPHOLOGICAL COMPONENT ANALYSIS

GUAN-JU PENG AND WEN-LIANG HWANG

Feb. 24, 2014 || Technical Report No. TR-IIS-14-002
http://www.iis.sinica.edu.tw/page/library/TechReport/tr2014/tr14.html

SIGNAL SEPARATION USING RE-WEIGHTED AND ADAPTIVE MORPHOLOGICAL COMPONENT ANALYSIS

GUAN-JU PENG AND WEN-LIANG HWANG*

Abstract. Morphological component analysis (MCA) for signal separation decomposes a signal into a superposition of morphological subcomponents, each of which is approximately sparse in a certain dictionary. Some of the dictionaries can also be modified to make them adaptive to local structure in images. We show that signal separation performance can be improved over the previous MCA approaches by replacing L1 norm optimization with "weighted" L1 norm optimization and replacing their dictionary adaptation with regularized dictionary adaptation. The weight on an atom for sparse coding is commonly derived from the corresponding coefficient's value. In contrast, the weight of an atom in a dictionary for signal separation is derived from the mutual coherence between the atom and the atoms in the other dictionaries. The proposed solution for regularized dictionary adaptation is an extension of the K-SVD method, where the dictionary and "weighted" sparse coefficients are estimated simultaneously. We present a series of experiments demonstrating the significant performance improvement of the proposed algorithm over the previous approaches for signal separation.

Key words. Adaptive morphological component analysis, signal separation, image separation, mutual coherence, re-weighted method.

AMS subject classifications. 62H35, 68U10, 94A12

1. Introduction.

Signal separation is fundamental in noise removal (white noise, reflection, rain, etc.), segmentation, inpainting, and compression. To achieve signal separation, Morphological Component Analysis (MCA) has been shown to be very effective and has been deployed in numerous applications [24, 25, 12, 15, 14, 17]. MCA facilitates the observation of morphological diversity in an image by modeling the image as the superposition of subcomponents of different morphological hypotheses. Morphological hypothesis means that the structure of a component can be sparsely or nearly sparsely represented by a dictionary. For example, the wavelet and the curvelet dictionary can be used to represent the morphological concepts of edges and smooth contour respectively. Also, the local DCT or the learned patches are usually used to represent the texture in a natural image [18, 4, 23]. In mathematics, MCA can be formulated as to derive the solution y_i of the following inverse problem:

$$y = \sum_i y_i + \epsilon, \tag{1.1}$$

where y is the input image and ϵ is the modeling noise. MCA assumes that each subcomponent y_i has a sparse or approximately sparse representation with respect to the dictionary matrix D_i. The nearly sparse representation of y_i with respect to D_i can be derived using the following convex optimization problem (a.k.a., noisy Basic Pursuit):

$$x_i^\star = \arg\min_{x_i} \|x_i\|_1 \qquad \text{subject to} \qquad \|y_i - D_i x_i\|_2 \le \epsilon_i, \tag{1.2}$$

where ϵ_i is the error in representing y_i with the dictionary D_i.

According to theoretical analysis [26], the separation of mixing signals using MCA is determined by the mutual coherence between dictionaries and the number of atoms

*Institute of Information Science, Academia Sinica, Taipei, Taiwan. (whwang@iis.sinica.edu.tw). Questions, comments, or corrections to this document may be directed to that email address.

used to approximate a subcomponent. Mutual coherence can be regarded as a method to measure the similarity of patterns, represented as atoms, in dictionaries. High mutual coherence can impair MCA. Intuitively, if two dictionaries contain highly similar patterns, then some strcutre in the input image can belong to more than one subcomponent, causing ambiguity in resolving the signal separation. This problem is particularly serious in MCA because it imposes the sparseness assumption on the total number of atoms to represent all subcomponents in the signal separation problem. Specifically, if the atom d belongs to dictionaries D_i and D_j and the atom is used to represent the input image y, then MCA tends to pick the atom because using the atom can achieve a sparse representation of both y_i and y_j. Meanwhile, the performance of MCA can degrade if a subcomponent cannot be sparsely represented by a dictionary. This usually occurs when MCA is used to retrieve a complicated image structure, such as texture. In [23], an adaptive scheme, called adaptive MCA, was proposed to address this concern by iteratively modifying the dictionaries during the separation process. The approach modifies the dictionaries according to the reconstructed subcomponents in the previous iteration, as a result of which, the dictionaries can be over-constrained by the previous reconstruction. Hence, the error (due to the noise or the patterns of another subcomponents) in a subcomponent can persist in the adaptation process.

In this paper, we improve MCA in two aspects. For the atoms having high mutual coherence with the atoms in other dictionaries, we use the weighted approach by assigning larger weights to the atoms to make them less likely to be selected for signal representations. Therefore, the weight on an atom for signal separation is a function of mutual coherence, in contrast to the common approaches in sparse coding where the weight on an atom is a function of the corresponding coefficient's value. The weighted approach to achieve L1 norm optimization has been widely used and studied [8]. However, there seems to be little discussion of using the approach for signal separation using MCA. Meanwhile, for dictionary adaptation, we modify adaptive MCA by imposing a regularization constraint on the dictionaries with which the newly derived dictionaries become less constrained by the previously reconstructed subcomponents. Because our weighting scheme is dependent on the mutual coherence of dictionaries, and the dictionaries are updated at each iteration, the scheme constitutes iterative re-weighting. For convenience, we refer to the proposed approach as re-weighted and adaptive MCA because it uses the dictionary adaption and re-weighted optimization.

We present experimental results and demonstrate that weighted and adaptive MCA can significantly improve performance over adaptive MCA. We also demonstrate our method's robustness in solving signal separation, by varying the parameters of the proposed approach. The rest of the paper is organized as follows. Section 2 reviews the development of the MCA model for signal separation. Section 3 presents the proposed method. Section 4 compares the experimental results with those for adaptive MCA. Section 5 presents the concluding remarks.

2. The MCA Model for Signal Separation. This section briefly reviews the methods and algorithms that use MCA for signal separation. MCA is formulated in Equations (1.1) and (1.2). Equation (1.1) indicates that the input image is to be decomposed into a superposition of morphologically different subcomponents, whereas Equation (1.2) indicates that each subcomponent can be sparsely represented with an appropriate chosen dictionary. Equation (1.2) is also called the sparse coding problem, which has been widely studied and analyzed. One may choose a greedy method from the family of matching pursuit algorithms [7, 20, 19, 22], or an L1-norm optimization algorithm from the family of noisy Basic Pursuit algorithms [9, 21, 5, 10, 3].

The Lagrangian approach can be used to combine Equations (1.1) and (1.2), thereby obtaining the following optimization problem:

$$\min_{\{y_i,x_i\}} \frac{1}{2}\|y-\sum_i y_i\|_2^2+\mu\sum_i E_i(y_i,x_i,D_i,\lambda), \tag{2.1}$$

where μ and λ are the Lagrangian multipliers and E_i is the energy function of the sparse coding for the i-th component, which can be written as

$$E_i(y_i,x_i,D_i,\lambda)=\frac{1}{2}\|y_i-D_ix_i\|_2^2+\lambda\|x_i\|_1. \tag{2.2}$$

If the subcomponent y_i is divided into m patches, denoted as $R_k(y_i)$ with $k=1\cdots m$, then the energy function of the sparse coding E_i consists of the energy functions of the patches:

$$E_i(y_i,x_i,D_i,\lambda)=\sum_{k=1}^{m}\frac{1}{2}\|R_k(y_i)-D_ix_i^k\|_2^2+\lambda\|x_i^k\|_1, \tag{2.3}$$

where the sparse coefficient vector x_i^k represents the k-th patch of the i-th component.

2.1. Methods for Morphological Components Analysis. The first solution for MCA was proposed by Starck, Elad, and Donoho [24]. Their algorithm is comprised of two steps in each iteration: updating the subcomponents $\{y_i\}$ and decreasing the value of a threshold θ. The component y_i is updated using the following formula:

$$y_i \leftarrow D_i\delta_\theta(D_i^{-1}(y-\sum_{j\neq i}y_j)), \tag{2.4}$$

where $\delta_\theta(\cdot)$ set the values of the input coefficients at zero if the values are less than the threshold θ. The function $D_i^{-1}(y-\sum_{j\neq i}y_j)$ indicates the pursuit process with respect to the dictionary D_i, and the output of the function is the coefficients of representing $y-\sum_{j\neq i}y_j$ with D_i. After all y_i are modified, the value of the threshold θ is decreased.

As indicated in [23], selecting appropriate dictionaries for the morphological components can be considered the fundamental task of MCA. An effective dictionary should be able to provide a sparse representation for the target component, through the sparse coding algorithm. Some image features, such as edges and smooth contours, can usually be represented in a sparse way by well-chosen dictionaries. Unfortunately, many more features in natural images, such as texture, can have more complicated structures and thus are hardly captured by fixed dictionaries. To address such cases, the descriptive dictionary is usually built by applying learning algorithms that are based on the concept of regression [13] or classification [1].

The adaptive MCA algorithm was proposed in [23], its main feature being to include a dictionary adaptation process in performing MCA. If the dictionary is fixed, then the algorithm does not modify it. But if the dictionary is learned from the training data, the dictionary can be modified by the algorithm. Each iteration of adaptive MCA is comprised of three steps: sparse coding; updating the subcomponents; and adjusting the learned dictionaries using the subcomponents obtained from the second step. In the first step, for each i, the sparse coefficient vector x_i is calculated as

$$x_i \leftarrow \arg\min_x E_i(y_i,x,D_i,\lambda), \tag{2.5}$$

where $E_i(y_i, x, D_i, \lambda)$ is the energy function of the sparse coding detailed in Equation (2.2) (for a fixed dictionary) or Equation (2.3) (for a learned dictionary).

In the second step, a sub-loop is used to update the subcomponents. The residual r_i is defined as $r_i := y - \sum_{j \neq i} y_j^{l-1}$, where y_j^{l-1} is the estimation of the j-th component in the $l-1$-th iteration of the sub-loop. The subcomponent y_i is updated as

$$y_i \leftarrow \arg\min_f \|r_i - f\|_2^2 + \mu \|f - D_i x_i\|_2^2 \tag{2.6}$$

for the fixed dictionary, or

$$y_i \leftarrow \arg\min_f \|r_i - f\|_2^2 + \mu' \sum_{j=1}^{m} \|R_j(f) - D_i x_i^j\|_2^2 \tag{2.7}$$

for the learned dictionary. In the third step, the learned dictionaries are updated as

$$D_i \leftarrow \arg\min_D E_i(y_i, x_i, D, \lambda). \tag{2.8}$$

The details of the algorithm are shown in Table 1.

3. The Re-weighted and Adaptive MCA Approach . Our approach is similar to adaptive MCA in that dictionaries are modified in solving MCA signal separation. However, our approach differs from adaptive MCA in two aspects: using a re-weighted scheme to solve the sparse coding problem, where the weights are derived from the mutual coherence of dictionaries; and imposing a regularization constraint on the learned dictionary for the dictionary adaption process. Section 3.1 explains why weighted sparse coding is more appropriate than sparse coding, and we present the method to derive the weights on atoms. Section 3.2 presents the regularization penalty imposed on the dictionary adaption procedure. Section 3.3 presents the proposed signal separation algorithm and analyzes its computational complexity.

3.1. Re-weighted Sparse Coding for Signal Separation. Mutual coherence is the parameter that indicates whether a signal can be uniquely separated by MCA into morphologically different subcomponents. Theoretical analysis shows that a sufficient condition for the success of signal separation using MCA is the low mutual coherence of atoms in different dictionaries [26]. If the column 2-norm of a dictionary is normalized to 1, and if the inner product between atoms in dictionaries D_1 and D_2 exists, the mutual coherence of D_1 and D_2 is defined as

$$\mu_m(D_1, D_2) := \max_{d_1 \in D_1, d_2 \in D_2} |d_1^T d_2|. \tag{3.1}$$

In the extreme case, if there is an atom belonging to different dictionaries, implying that μ_m is 1 (the maximum), then any information encoded by the atom can cause ambiguity in signal separation, because MCA cannot determine to which subcomponent the atom should contribute. Thus, similar atoms possessed by different dictionaries incur ambiguity when calculating the corresponding coefficients through sparse coding. An intuitive way to decrease the mutual coherence between dictionaries is to remove their similar atoms. This reduces the number of atoms in a dictionary, therefore increasing the number of nonzero coefficients in a signal's representation. Also, removing atoms can be prohibited, such as where the dictionary is a basis.

Our approach is motivated by iteratively re-weighted sparse recovery methods, in which coefficients of smaller values are assigned heavier weights so that those coefficients are less likely to contribute to the spare signal representation in the next

iteration [8]. We assign heavier weights to atoms that have higher mutual coherence with the other dictionaries, thereby causing those atoms to have smaller values when solving the weighted sparse coding problem, and consequently making them less likely to be selected.

3.1.1. Mutual Coherence of an Atom across Dictionaries. We use x^F or x^L to indicate that x is an attribute of the fixed image dictionary or the learned patch dictionary respectively. For example, given that D is a dictionary, D^L is a learned patch dictionary and D^F is a fixed image dictionary. Another example: given that d is an atom, d^F is an atom from a fixed dictionary and d^L is one from a learned dictionary. The size of an atom in a fixed dictionary is equal to that of the whole image. If an image is divided into patches, then the size of an atom in a learned dictionary is the same as a patch in the image. Because the patches are allowed to overlap each other, they are not necessarily a partition of an image. For convenience, we assume all the patches in an image are represented with the same (patch) dictionary, even though this assumption is not necessary.

Mutual coherence can be used as an indicator of whether the sparse approach has produced a unique signal representation or separation. The objective of sparse coding is to determine whether there is a unique representation of a signal involving the atoms in a dictionary; therefore, mutual coherence is measured between atoms in the dictionary. In contrast, for signal separation, mutual coherence between atoms in the same dictionary is of no interest because the focus of the problem is not on the unique representation of a subcomponent, but on the unique separation of the signal into subcomponents; therefore, mutual coherence should be measured between atoms in different dictionaries, but not in the same dictionary. Mutual coherence is defined based on the inner product of two atoms. Because the sizes of atoms in different dictionaries may not be the same, the definition of mutual coherence must accommodate this.

In the following, $\mu_n(d, D)$ denotes the mutual coherence of the atom d measured against all atoms in the dictionary D. Recall that the function R_j takes the j-th patch from an image and that the 2-norm of an atom in normalized to 1. First, we define the mutual coherence of an image-sized atom d^F against the learned patch dictionary D^L as

$$\mu_n(d^F, D^L) = \frac{1}{m} \sum_{j=1}^{m} \mu_n^j(d^F, D^L), \tag{3.2}$$

and

$$\mu_n^j(d^F, D^L) = \max_{d^L \in D^L} \frac{|R_j(d^F)^T d^L|}{\|R_j(d^F)\|_2}. \tag{3.3}$$

The definition indicates that $\mu_n(d^F, D^L)$ is the average of the mutual coherence between all patches $R_j(R^F)$ and the patch dictionary D^L. The patch taken from the atom d^F is normalized to 1 in Equation (3.3).

Then, we define the mutual coherence of a patch-sized atom d^L to measure against an image dictionary D^F. Because the patch-sized atom is used to represent a patch in an image, say the j-th patch, the mutual coherence can be derived from the j-th patch of all atoms in D^F. We define

$$\mu_n^j(d^L, D^F) = \max_{d^F \in D^F} \frac{|R_j(d^F)^T d^L|}{\|R_j(d^F)\|_2}. \tag{3.4}$$

The definition of $\mu_n^j(d^L, D^F)$ and $\mu_n^j(d^F, D^L)$ is not symmetric by comparing Equations (3.3) and (3.4). Finally, the mutual coherence of an image-sized atom against an image dictionary and of a patch-sized atom against a patch dictionary can be defined respectively as

$$\mu_n(d^F, D^F) = \max_{d \in D^F} |(d^F)^T d|, \tag{3.5}$$

$$\mu_n(d^L, D^L) = \max_{d \in D^L} |(d^L)^T d|. \tag{3.6}$$

Let Ω, Ω^F, and Ω^L denote the set of all dictionaries, the set of fixed dictionaries for an image, and the set of learned dictionaries for patches respectively. It follows that $\Omega = \Omega^F \cup \Omega^L$. The definitions can be extended to measure the mutual coherence of an atom against a set of dictionaries where

$$\mu_n(d^F, \Omega^L) = \max_{D^L \in \Omega^L} \mu_n(d^F, D^L), \tag{3.7}$$

$$\mu_n^j(d^L, \Omega^F) = \max_{D^F \in \Omega^F} \mu_n^j(d^L, D^F) \text{ with } j = 1, \cdots, m, \tag{3.8}$$

$$\mu_n(d^F, \Omega^F) = \max_{D^F \in \Omega^F / D} \mu_n(d^F, D^F) \text{ and } d^F \in D, \tag{3.9}$$

$$\mu_n(d^L, \Omega^L) = \max_{D^L \in \Omega^L / D} \mu_n(d^L, D^L) \text{ and } d^L \in D, \tag{3.10}$$

where the dictionary D that contains the atom d^F (d^L) is removed from Ω^F (Ω^L) in calculating Equation (3.9) (Equation (3.10)).

In Equations (3.7), (3.8), (3.9), and (3.10), the mutual coherence of an atom is calculated against all atoms in the other dictionaries. Because of the sparse model assumption that each subcomponent can be represented with only few atoms in the dictionary, the mutual coherence can be derived from the subset of atoms, called the supports of dictionaries, that are used to represent the subcomponents. To compute the mutual coherence of an atom with the atoms in the support of the other dictionaries not only increases the computational speed, but prevents artifacts caused by the out-of-support atoms. The atoms that are not supporting any subcomponent presumably will not cause any ambiguity issue for signal separation, which implies that removing them can avoid the over-estimation of mutual coherence. As will be demonstrated in Section 4, if we can avoid over-estimating mutual coherence, our method can achieve high signal separation performance.

Because the subcomponents and their supports are not known before signal separation, at the onset of our algorithm, the mutual coherence of an atom is derived (as defined in this section) with all the atoms in other dictionaries. Then, the mutual coherence is updated by using the supports of the subcomponents estimated from the previous iteration.

3.1.2. Weighting Functions. Unlike the common approaches where the weighting on an atom is a function of the value of the corresponding coefficient, our weighting on an atom is a function of the mutual coherence of the atom against the dictionaries that do not contain the atom. Without loss of generality, we use $d_i[k]^a$ with $a \in \{L, F\}$ to denote the k-th atom in dictionary D_i^a. We also use $d_i^j[k]^L$ to denote the k-th atom in D_i^L associated with the j-th patch of an image. The mutual coherence of $d_i^j[k]^L$ and $d_i[k]^F$ with respect to the dictionary set Ω are defined respectively as

$$v_i^j[k] = \max\{\mu_n(d_i^j[k]^L, \Omega^L), \mu_n^j(d_i^j[k]^L, \Omega^F)\} \text{ with } j = 1, \cdots, m; \tag{3.11}$$

$$v_i[k] = \max\{\mu_n(d_i[k]^F, \Omega^F), \mu_n(d_i[k]^F, \Omega^L)\}. \tag{3.12}$$

In the following, to avoid overloading with notation, we use $v[k]$ to denote $v_i[k]$ or $v_i^j[k]$, provided that there is no confusing in contents. A weighting function must be a non-negative and monotonically increasing function of $v[k]$, and the range of the function is confined to ensure that the function's value does not become unbounded. We choose the following weighting function on an atom in the dictionary D:

$$w(v[k]) = e^{\gamma \frac{v[k]}{\max_l v[l]}} = \alpha^{\frac{v[k]}{\max_l v[l]}}, \tag{3.13}$$

where $\max_l v[l]$ is the maximum mutual coherence of the atoms in D, and γ is the parameter that depends on the image and the dictionaries. The range of the weighting function is $[1, \alpha]$, and dividing $v[k]$ by $\max_l v[l]$ can increase the variance on the values of the weighting function. Let the number of atoms in D be $|D|$, and let v represent the column vector of $v[k]$ with $k = 1, \cdots, |D|$; then, we define the k-th output element of the mapping $g(D)$ as

$$g(D)[k] := w(v[k]). \tag{3.14}$$

3.1.3. Weighted Sparse Signal Separation. We now consider the derivation of the representation coefficients for the case where the dictionaries are given. To incorporate the weighting information into the MCA model in Equation (2.1), we can modify the energy functions defined in Equations (2.2) and (2.3). This results in the following formulation, where D_i, λ, μ, and y are given:

$$\min\{\frac{1}{2}\|y - \sum_i y_i\|_2^2 + \mu\{\sum_{D_i \in \Omega^L} E_W^L(y_i, [x_i^1 \ x_i^2 \ \cdots \ x_i^m], D_i, \lambda) + \sum_{D_i \in \Omega^F} E_W^F(y_i, x_i, D_i, \lambda)\}\}, \tag{3.15}$$

where E_L^W and E_F^W are the weighted energy functions for the learned and the fixed dictionaries respectively and λ and μ are Lagrangian multipliers. The weighted energy functions are defined as follows:

$$E_W^F(y, x, D, \lambda) = \frac{1}{2}\|y - Dx\|_F^2 + \lambda \|g(D)^T x\|_1, \tag{3.16}$$

$$E_W^L(y, [x^1 \ x^2 \ \cdots \ x^m], D, \lambda) = \sum_{j=1}^{m} \frac{1}{2}\|R_j(y) - Dx^j\|_F^2 + \lambda \|g(D)^T x^j\|_1. \tag{3.17}$$

Since the optimization problem in Equation (3.16) is convex with respect to the subcomponents $\{y_i\}$ and representation coefficients $\{x_i\}$, we propose an iterative algorithm to minimize the Lagrangian. Each iteration of the algorithm is comprised of two steps: finding $\{x_i\}$, and finding $\{y_i\}$. In the first step, the coefficients x_i of each subcomponent are updated to the minimum of the weighted energy function:

$$\begin{cases} [x_i^1 \ x_i^2 \ \cdots \ x_i^m] \leftarrow \\ \qquad \arg\min_{[u^1 \ u^2 \ \cdots \ u^m]} E_W^L(y_i, [u^1 \ u^2 \ \cdots \ u^m], D_i, \lambda), & \forall \ D_i \in \Omega^L; \\ x_i \leftarrow \arg\min_u E_W^F(y_i, u, D_i, \lambda), & \forall \ D_i \in \Omega^F. \end{cases} \tag{3.18}$$

The minimization procedure for Equation (3.18) is usually referred to as the weighted sparse coding, which has been widely studied [27, 16, 8, 5, 10, 2]. In the second step, $\{y_i\}$ are updated to minimize the Lagrangian. If the other parameters are held

constant, the formulation of the minimization is exactly the same as those of adaptive MCA, which are manifested in Equation (2.6) for the fixed dictionaries and Equation (2.7) for the learned dictionaries. Because of the optimization of a convex function, our procedure to derive $\{y_i\}$ and $\{x_i\}$ with given dictionaries is always convergent.

3.2. Distance-constrained Dictionary Adaptation. Images usually contain complicated structure, such as texture, that cannot be simply captured with an image dictionary. [23] proposed an extension of MCA by dividing the image into patches and using a learning method to learn the dictionary for the patches. Because of huge variations of image structure, the learned dictionary is then modified to make it adaptive to the derived subcomponents from the previous iteration. This approach can improve the separation performance of the MCA-based signal separation approach. The dictionary adaptation method is to derive the patch dictionary for each texture subcomponent:

$$D^* \leftarrow \arg\min_D \sum_{j=1}^{m} \|R_j(y_i) - Dx_i^j\|_F^2, \tag{3.19}$$

where D^* is the patch dictionary for subcomponent i. This approach might work well for segmentation, where subcomponents are located at disjoint spatial areas in an image, but would work less well for signal separation. The dictionary depends on the subcomponents derived from the previous iteration, and the subcomponents may contain noise or patterns from other subcomponents. Therefore, dictionary distortion can persist is the adaptation process. In addition, several iterations of this approach can result in a dictionary that is quite different from the original learned dictionary, which usually capture certain desired structure of a subcomponent. We, thus, impose a regularization penalty on the distance between the derived dictionary and the original learned dictionary at each iteration. Hence, the dictionary derived from our dictionary adaptive process is based on the information derived from the previous iteration as well as the original learned dictionary.

In Equation (3.19), if Y were to denote the matrix $[R_1(y_i), R_2(y_i), \cdots, R_m(y_i)]$ and X were to denote the matrix $[x_i^1, x_i^2, \cdots, x_i^m]$, then the equation can be rewritten as

$$D^* \leftarrow \arg\min_D \|Y - DX\|_F^2. \tag{3.20}$$

Applying the regularization penalty to Equation (3.20) yields

$$D^* \leftarrow \arg\min_D \|Y - DX\|_F^2 + \lambda_{dict}\|D - D^{\text{train}}\|_F^2. \tag{3.21}$$

Equation (3.21) can be solved by standard matrix calculation technique. Because X is sparse, we can use an alternative approach, motivated by the K-SVD algorithm [1], to derive its solution with an additional constraint on the supports on nonzero coefficients in X. Let $supp(X)$ denote the positions of the nonzero coefficients in X, and let $supp(X^{old})$ denote those of X derived from the previous iteration. The proposed dictionary adaptation process requires that $supp(X^{old})$ is retained in $supp(X)$. Dictionary adaptation is an iterative process, formulated as follows for one iteration:

$$[D, X] \leftarrow \arg\min_{D,X} \|Y_i - DX\|_F^2 + \lambda_{dict}\|D - D^{\text{train}}\|_F^2 \quad \text{with } supp(X) \subseteq supp(X^{old}). \tag{3.22}$$

In Equation (3.22), both D and X are updated at each iteration. Also, if the Lagrangian λ_{dict} is set to zero, this optimization problem is identical to the K-SVD problem. The optimization problem can be solved by using the SVD to update one column of the dictionary and the corresponding matrix nonzero row coefficients one at a time in sequence. The Appendix presents the detailed procedure to derive the optimal D and X.

3.3. Proposed Signal Separation Algorithm. The re-weighted and adaptive MCA approach intends to derive the subcomponents $\{y_i\}$, the weighted sparse coding coefficients $\{x_i\}$, and the dictionaries $\{D_i\}$ that minimize the following optimization function:

$$E^{all}(\{y_i\},\{x_i\},\{D_i\}) = \tag{3.23}$$

$$\frac{1}{2}\|y-\sum_i y_i\|_F^2 + \tag{3.24}$$

$$\mu\{\sum_{D_i\in\Omega^F}\frac{1}{2}\|y_i-D_ix_i\|_F^2+\sum_{D_i\in\Omega^L}\sum_{j=1}^{m}\frac{1}{2}\|R_j(y_i)-D_ix_i^j\|_F^2\}+ \tag{3.25}$$

$$\mu\lambda\{\sum_{D_i\in\Omega^F}\|g(D_i)^Tx_i\|_1+\sum_{j=1}^{m}\sum_{D_i^j\in\Omega^L}\|(g(D_i^j))^Tx_i^j\|_1\}+ \tag{3.26}$$

$$\mu\lambda_{dict}\{\sum_{D_i\in\Omega^L}\|D_i-D_i^{\text{train}}\|_F^2\}, \tag{3.27}$$

where $\{D_i^{\text{train}}\}$ are the trained dictionaries. Recall that Ω^F and Ω^L are the set of fixed image dictionaries and the set of learned patch dictionaries respectively; an image is divided into m patches of equal size; and μ, λ, and λ_{dict} are Lagrangian multiplier parameters. In Equation (3.26), the patch dictionary D_i^j as the input argument to the weighting function g is location-dependent, because the supports for sparse representations with the patch dictionaries vary with the locations of patches. The proposed method is an iterative algorithm, comprised of four steps at each iteration: estimating the weighted sparse coefficient vectors $\{x_i\}$; updating the subcomponents $\{y_i\}$; dictionary adaptation $\{D_i\}$; and calculating the weights to each atom, $\{g(D_i), g(D_i^j)\}$.

In the first step, a weighted L1 minimization algorithm is used to derive the coefficient vectors that minimize the second and third terms of E^{all} (Equations (3.25) and (3.26)). In the second step, the subcomponents are derived by minimizing the first and second terms of E^{all} (Equations (3.24) and (3.25)). Details on the two steps can be found in Section 3.1.3. In the third step, the patch dictionaries $\{D_i \in \Omega^L\}$ are updated by minimizing the second and fourth terms of E^{all} (Equations (3.25) and (3.27)) by using the method detailed in Section 3.2. In the fourth step, the weight to each atom is updated according to the current supports of sparse coding coefficients, as described in Section 3.1.2. The step-by-step procedure of the proposed algorithm is presented in Table 2.

The following analyzes the computational complexity of our algorithm. The computational complexity of the first and second steps is bounded by the weighted sparse coding, which calculates the representing vectors for $|\Omega|$ dictionaries. We use different algorithms to derive the weighted sparse coding solutions for image dictionaries versus patch dictionaries. Because there are numerous atoms in an image dictionary, using conventional proximal methods to solve L1 minimization is too computational intensive. Therefore, we use fast wavelet/curvelet decomposition [18, 4] and "weighted"

soft thresholding to derive the coefficients. "Weighted" soft thresholding updates the decomposition coefficients vector x_i as

$$x_i[k] \leftarrow \begin{cases} sign(x_i[k])(|x_i[k]| - \lambda g(D_i)[k], \text{ if } |x_i(k)| > \lambda g(D_i)[k], \\ 0, \text{ otherwise,} \end{cases} \tag{3.28}$$

which can be obtained by modifying L1 optimization sparse coding methods straightforwardly [11]. The complexities of fast wavelet/curvelet analysis and soft thresholding are $\mathcal{O}(Nlog(N))$ and $\mathcal{O}(N)$ respectively, where N is the number of pixels in an image. On the other hand, the weighted sparse coding for patch dictionaries can be implemented using an algorithm in the block descent family [8, 21, 5, 10, 3, 6]. The complexity of these algorithms depends on the required accuracy of the estimation of coefficients. If the desired accuracy of the approximate solution is ϵ_{sparse}, i.e., $||x_i - x_i^*||_2 \leq \epsilon_{sparse}$ where x_i^* is the estimated and x_i is the true coefficient, then the number of the steps required to achieve that accuracy is $\mathcal{O}(\frac{1}{\epsilon_{sparse}^2})$ for the Nestrov algorithm [21]. Asif et al. proposed a method to solve the weighted sparse coding method [2]. The method uses a guess of x_i as the initial to speed up the convergence. The method is suitable for solving the weighted sparse coding in our iterative algorithm because the x_i obtained in the previous iteration should be an ideal warm start for the current iteration. We use their algorithm to derive the weighed sparse coding coefficients for patch dictionaries.

The Appendix analyzes the complexity for dictionary update (the third step of our algorithm). If the number of atoms in each patch dictionary is bounded by K, the number of patch dictionaries is $|\Omega^L|$, and N is the size of the image, then the computational complexity is $\mathcal{O}(K^2N|\Omega^L|)$. The fourth step of our algorithm calculates the weight of each atom in the support of dictionaries, i.e., the atoms whose coefficients are used in the weighted sparse representations of subcomponents. The major computational cost is consumed in calculating the mutual coherence of an atom with atoms in the supports of other dictionaries. Let S denote the largest number of coefficients selected for weighted sparse representation of a subcomponent. The number of atoms in a dictionary by wavelet/curvelet analysis is bounded by $N \log N$, where N is the size of an image, and the total number of dictionary is $|\Omega|$. To simplify the analysis, we assume that the atoms in all $|\Omega|$ dictionaries have size N. To calculate the weight of an atom in a dictionary requires at most $(|\Omega| - 1)S$ inner products. Each inner product takes $\mathcal{O}(N)$ multiplications. So, the weights can be derived by $\mathcal{O}(|\Omega|^2SN)$.

Summarizing from the above complexity analysis, we obtain the cost $\mathcal{O}(N \log N + |\Omega|^2SN + K^2N|\Omega^L|)$ for executing one iteration of our algorithm. The number of atoms for a patch dictionary is K, of order $\mathcal{O}(\frac{N}{m})$ with $m = kN$ and the constant $k < 1$. In our implementation, k is set at about 0.01. If $S > \log N$, then the weight update step (the fourth step) dominates the computational cost; otherwise, the fast wavelet/curvelet decomposition in the first step dominates.

4. Experimental Results. We compared our results with those of adaptive MCA [23] in solving the signal separation problem. Adaptive MCA is much more complicated and computationally intensive than is MCA [24]. However, as shown in [23], adaptive MCA outperforms MCA in retrieving the texture component from the image. Therefore, we compared the results of our method only with those of adaptive MCA. We implemented adaptive MCA using the algorithm shown in Table 1, and our method's algorithm is shown in Table 2. The maximum number of iterations was set to 50. However, our experiments usually reached convergence earlier.

Our experiments used one-dimensional signals and two-dimensional images. Our methods have four parameters: μ, λ, λ_{dict}, and α. μ reflects the noise level ϵ in Equation (1.1) and was set to 0.1 in all experiments. λ_{dict} constrains the distance from the trained dictionary in the dictionary adaptation and was set to 10 in all experiments. The other two parameters were varied to demonstrate their effect on separation performance. The parameter λ and μ are also in the adaptive MCA. For our implementation of adaptive MCA, μ was set to 0.1 and λ was permitted to vary, as with our method.

Both methods have two types of dictionaries: fixed image dictionary and learned patch dictionary. We can therefore categorize the experiments according to the types of dictionaries used to represent each subcomponent. In all experiments, we separated an input signal into two subcomponents; hence, the experiments can be categorized into "Fixed vs. Fixed," "Fixed vs. Learned," and "Learned vs. Learned." Although the image dictionary is fixed in the signal separation process, the learned patch dictionary is constantly updated by both methods. We demonstrate the results of these three categories as follows.

A. Fixed vs. Fixed

In the first experiment, two signals $y_1^\star$ and $y_2^\star$ were constructed to be sparse with respect to the DCT and DWT basis respectively. The size of each signal was 128; the sparsity of the signals, denoted by S, ranged from 4 to 32 with randomly selected supports. The nonzero coefficients were sampled randomly from a Laplace (15, b) distribution, with b adjusted so that the L2 norm of both signals is the same. The input signal was $y = y_1^\star + y_2^\star + n$, where n was $\mathcal{N}(0, 1)$ noise.

For each value of S, we performed our algorithm and adaptive MCA on 200 trails of y. The results are shown in Figure 1. In the left sub-figure of Figure 1, the SNR performance of both algorithms decreases when S increases. As shown in the right sub-figure, this resulted from the mutual coherence increment caused by the enlarged support of representing each signal. This result indicates that defined mutual coherence is negatively correlated to separation performance. Also, as shown in Figure 1, the SNR performance of our method is superior to that of adaptive MCA for any value in the range of α, listed in the legend of the figure.

The second experiment was conducted on the superposition of two images, where one can be sparsely represented by DCT and the other by DWT. Figure 2 shows and compares the signal separation results. Both methods can separate the two images quite well, with only slight perceptual difference. However, the methods' PSNR values indicate that our method had average gain of about 6 dB greater than that of adaptive MCA. The PSNR gain of our method is due to better contrast recovery than adaptive MCA.

B. Fixed vs. Learned

We performed separation on the superposition of two images y_1 and y_2, where y_1 was approximately sparse by a fixed image dictionary and y_2 by the learned patch dictionary. The image size was 256×256, the size of a patch was 10×10, and the overlap of the adjacent patches was 1 pixel. Image y_1 was taken from the first category, consisting of images at the top two rows of Figure 3. Image y_2 was taken from the second category, consisting of images at the bottom two rows of the figure. The training patches to derive the learned patch dictionary for texture image y_2 were randomly picked from the image. Therefore, the learned patch dictionary well matches the texture image. The dictionary was learned using the K-SVD algorithm.

The input signal was obtained as the linear combination of y_1 and y_2 with $y =$

$(1-\beta)y_1 + \beta y_2$, where the value of β ranged from 0.2 to 0.8. The average PSNRs obtained by adaptive MCA with different β and λ values for the image y are shown in the subfigure at the top row of Figure 4. Because the dictionary adaptation of adaptive MCA may result in a dictionary quite distant from the original learned dictionary, this can degrade the separation performance. Thus, for each input image y, we executed the algorithm twice: one with dictionary adaption and the other without. Then, the better result of the two was taken to represent performance, at the top row of Figure 4. The PSNR gains of our method over adaptive MCA are shown in the subfigures at the middle and bottom rows of Figure 4. Each measurement point in the subgraphs is the average gain of separating 16 composite images, derived using the same β and λ values in both methods. Because of how we obtained the performance of adaptive MCA, our weighting scheme is the main factor contributing to the gain of our method. The higher the gain of our method, the better our weighting scheme can resolve the ambiguity of signal separation.

As shown in the middle and bottom rows of Figure 4, our method outperforms adaptive MCA in all cases. Some interesting observations: (1) With fixed α and λ, increasing β (the energy of the texture subcomponent) decreases the gain. Since the learned patch dictionary can sparsely represent the texture subcomponent, ambiguity is not serious for signal separation when β is large, where the input image is dominated by the texture subcomponent. This is why the proposed weighting scheme achieves only a slight improvement. (2) The gain at $\lambda = 1$ ranges from 0.5 to 2 dB, but at other λ values, it ranges from 2 to 4 dB. Smaller λ implies the representation is not sparse. This caused overestimating of the mutual coherence and the weighting values on atoms (as is described in the last paragraph of Section 3.1.1). (3) The gain is quite robust to changes in α.

The top row of Figure 5 demonstrates the separation results of the input image. The visual quality of our reconstruction, in the right-hand column, is noticeably better than that of the compared method, in the middle column. The average PSNR of our reconstruction is about 4 dB higher than that of the compared method.

C. Learned vs. Learned

The setting of the experiments and the range of parameter values were the same as that in part **B** of Section 4, except that the image y_1 is also a texture image, taken from the bottom two rows of Figure 3. The training procedure and training examples for the dictionaries of y_1 and y_2 were obtained similarly to part **B** of Section 4.

The performance of adaptive MCA was measured as described in part **B** of Section 4 and is shown in the subgraph at the top of Figure 6. The average PSNR gains of our method over that of adaptive MCA are demonstrated in the subfigures at the middle and bottom rows of Figure 6. The weighting scheme is the main reason for the gain of our method. Except in some cases when $\alpha = 64$ (the weighting parameter), our method outperformed the compared method. Compared to that in part **B** of Section 4, the PSNR gain of separating two texture images was lower. Some interesting points: (1) The gain with $\lambda = 1$ is the lowest, which is consistent with the results obtained in part **B** of Section 4. For small λ, the representation is not sparse. This causes overestimating the mutual coherence values. (2) Our method's performance was robust over a range of α values. (3) The gain was higher when β was low or high. In the cases, one texture subcomponent dominated the input. The dominating texture can be sparsely represented by its corresponding dictionary. The representation coefficients of the other texture are spread out, yielding the overestimation of the mutual coherence value.

Figure 7 demonstrates the separation results of the composite image, shown at the top row of the figure. The visual quality of our reconstruction, in the right-hand column, is obviously better than the compared method, in the middle column. The average PSNR of our reconstruction is about 2 dB higher than that of the compared method.

Summarizing from our experiments, we conclude that adopting the proposed weighting scheme and dictionary adaptation in MCA can improve the signal separation results. As demonstrated, the parameters used in our method are quite robust. To achieve high performance, the following conditions must be satisfied: (1) the representation is approximately sparse (or λ cannot be too small); and (2) the weighting parameter, α, to derive the defined mutual coherence, cannot be too high.

5. Conclusion. We proposed a novel method, called re-weighted and adaptive MCA, to facilitate the morphological diversity of the signals. To increase the diversity of the resulting signals, we assigned "heavier" weights to atoms of a subcomponent that are highly coherent with atoms of other subcomponents. In addition, we imposed on dictionary adaptation the constraint that the adaptive dictionaries be preserved inside a ball centered at the original trained dictionaries. Our experiments showed that the proposed method significantly outperforms adaptive MCA in various MCA-based applications. Our future work includes building dictionaries having great diversity and investigating the properties of the weighting scheme introduced in this paper.

Appendix A. Appendix.

To solve the problem in (3.22), we update one column at a time in the dictionary along with the corresponding nonzero row coefficients. Let d_j and r_j^T denote the j-th column in D and the j-th row in X respectively. Let $p_j(k)$ be the j-th nonzero index in r_j^T, and let $||r_j^T||_0$ be the number of nonzero coefficients. If we let Ψ_j be an $L \times \|r_k^T\|_0$ matrix in which the entries $(p_j(k), k)$ are set at one and other entries are set at zero, $r_j^T\Psi_j$ is a $1 \times ||r_j^T||_0$ row vector with nonzero entries. For example, if $r_j^T = [0\ 1\ 0\ 0\ 2]$, then $r_j^T\Psi_j$ is $[1\ 2]$.

Next, we consider the procedure used to update the j-th column d_j in the dictionary and the corresponding nonzero coefficients $r_j^T\Psi_j$. First, we have

$$||Y_i\Psi_j - DX\Psi_j||_F^2 = ||(Y - \sum_{l\neq j} d_l r_l^T)\Psi_j - d_l r_l^T\Psi_j||_F^2 \tag{A.1}$$

$$= ||E_j\Psi_j - d_l r_l^T\Psi_j||_F^2, \tag{A.2}$$

$$= ||\tilde{E}_j - d_j x_j^T||_F^2, \tag{A.3}$$

where $E_j = Y - \sum_{l\neq j} d_l r_l^T$, $\tilde{E}_j = E_j\Psi$, and $x_j^T = r_j^T\Psi_j$. Equation (A.3) is the K-SVD updating rule, which uses a rank 1 matrix, $d_j x_j^T$, to approximate the error matrix $\tilde{E}_j$. Let d_j^{train} denote the j-th column in the dictionary D_i^{train}. Then, we have

$$\begin{aligned} &||Y\Psi_j - DX\Psi_j||_F^2 + \lambda_{dict}||D - D^{\text{train}}||_F^2 \\ &= ||\tilde{E}_j - d_j x_j^T||_F^2 + \lambda_{dict}||d_j - d_j^{\text{train}}||_F^2 + C_j, \end{aligned} \tag{A.4}$$

where C_j is a constant that is independent of d_j and x_j^T. Now, to obtain the optimal solution of the problem in Equation (3.22), we have to solve the following subproblem:

$$\min_{d_j, x_j^T} ||\tilde{E}_j - d_j x_j^T||_F^2 + \lambda_{dict}||d_j - d_j^{\text{train}}||_F^2. \tag{A.5}$$

Comparison of Equations (A.3) and (A.5) shows that our approach is a rank 1 approximation of $\tilde{E}_j$, like the K-SVD approach. However, our d_j is penalized because of its distance from d_j^{train}. Therefore, the optimal solution cannot be obtained by the K-SVD method, which takes the SVD of $\tilde{E}_j = U\Delta V^T$ and then assigns $d_j = u_1$ (the first column of U) and $x_j^T = \Delta(1,1)v_1^T$ (by multiplying the first row of V^T by the $(1,1)$ element of Δ).

Let $f(d_j, x_j; \tilde{E}_j, d_j^{\text{train}}) = \|\tilde{E}_j - d_j x_j^T\|_F^2 + \lambda_{dict}\|d_j - d_j^{\text{train}}\|_F^2$. After taking the partial derivatives of the function f with respect to d_j and x_j and setting the results to zero, we obtain the following equations for the optimal solution of Equation (A.5):

$$(||x_j||^2 + \lambda_{dict})d_j = \tilde{E}_j x_j + \lambda_{dict} d_j^{\text{train}} \tag{A.6}$$

$$||d_j||^2 x_j = \tilde{E}_j^T d_j, \tag{A.7}$$

where $||d_j|| = 1$ because each column of the dictionary is normalized. Next, we derive the solutions of d_j and x_j in (A.6) and (A.7). For convenience, we omit the subscript indices in the following equations. Equation (A.6) can be rewritten as

$$||x||^2 d = \tilde{E}x + \lambda_{dict}(d^{\text{train}} - d). \tag{A.8}$$

If we fix x to find d, Equation (A.8) can be formulated as a root-finding problem and use Newton's method to update the vector d. Let the root-finding problem be $g(d) := (||x||^2 + \lambda_{dict})d - \tilde{E}x + \lambda_{dict}d^{\text{train}}$, where the root of $g(d)$ is the solution of Equation (A.8). Then, based on Newton's method, the vector d is updated by

$$d \leftarrow d_o - \frac{g(d_o)}{g'(d_o)} = d_o - \frac{(||x||^2 + \lambda_{dict})d_o - \tilde{E}x + \lambda_{dict}d^{\text{train}}}{||x||^2 + \lambda_{dict}}, \tag{A.9}$$

where the subscript o indicates the current estimation, so that d_o is the current estimation of d.

Assume that x is in the direction of an eigenvector of $\tilde{E}^T\tilde{E}$; that is, $x = ||x||\hat{x}$ and

$$\mu\hat{x} = \tilde{E}^T\tilde{E}\hat{x}, \tag{A.10}$$

where μ is the corresponding eigenvalue. Then, Equation (A.8) becomes

$$\|x\|^3\hat{x} = \mu\|x\|\hat{x} + \lambda_{dict}\tilde{E}^T(d^{\text{train}} - d). \tag{A.11}$$

Now we need only to derive the norm of x. From Equation (A.7), we have $x^T x = \|x\|^2 = d^T\tilde{E}\tilde{E}^T d$. In addition, we let the SVD of $\tilde{E}\tilde{E}^T$ be

$$\tilde{E}\tilde{E}^T = \sum_{i=1}^{l} \sigma_i^2 v_i v_i^T. \tag{A.12}$$

The norm of x can be written as

$$\|x\|^2 = \sum_{i=1}^{l} \sigma_i^2 < d, v_i >^2, \tag{A.13}$$

where l is the rank of $\tilde{E}$ and $\sigma_1 \geq \cdots \geq \sigma_l$. This equation gives the bounds of the norm by

$$\sigma_l \leq ||x|| \leq \sigma_1. \tag{A.14}$$

Because of Equation (A.7), we can substitute x for $\tilde{E}^T d$ in Equation (A.11) and obtain

$$(\|x\|^2 + \lambda_{dict} - \mu)x = \lambda_{dict}\tilde{E}^T d^{\text{train}}. \tag{A.15}$$

Taking the norm on both sides of the above equation, we obtain

$$(\|x\|^2 + \lambda_{dict} - \mu)\|x\| = \lambda_{dict}\|E^T d^{\text{train}}\|. \tag{A.16}$$

To update the dictionary column (atom) d and the corresponding coefficient vector x, we use the following procedure: (1) if $\lambda_{dict} = 0$, we return the result of the K-SVD method; or (2) if $\lambda_{dict} > 0$, we substitute the current estimate d into Equation (A.13) to derive the rough estimation of the norm, which is denoted by $\|x_o\|$ and used to replace $\|x\|$ in the expression $\|x\|^2 + \lambda_{dict} - \mu$ in Equation (A.16). Using rough estimation, the norm of x is updated by by

$$\|x\| \leftarrow \begin{cases} \frac{\lambda_{dict}\|E^T d^{\text{train}}\|}{\|x_o\|^2+\lambda_{dict}-\mu}, \text{ if } \|x_o\|^2 + \lambda_{dict} - \mu \neq 0, \\ \|x_o\|, \text{ otherwise.} \end{cases} \tag{A.17}$$

The term $||x^{new}||$ replaces $||x_o||$ in subsequent iterations until the norm does not change. At that point, we substitute $||x^{new}||$ for $||x||$ in Equation (A.9) to derive the new atom d. The process is repeated until convergence or some stopping condition on the number of iterations is reached.

The computational complexity of the above procedure is bounded by the SVD performed on the matrix $E_j\Psi_j$, where j indicates that E_j is the residual of the j-th column. The average complexity of SVD is bounded by the size of the matrix $E_j\Psi_j$. The size of the received signal is N, and the number of patches is m, which means that the number of rows in $E_j\Omega_j$ is N/m. The number of columns is the number of nonzero coefficients in the j-th row of the coefficient matrix X of size $K \times m$, where K is the number of atoms possessed by the dictionary. If each column of X contains at most S nonzero coefficients, the number of nonzero coefficients in X is boundedby mS. If the nonzero coefficients are distributed uniformly in X, the average nonzero coefficient in each row is $\frac{mS}{K}$. Thus, the size of $E_j\Omega_j$ is $\frac{N}{m} \times \frac{mS}{K}$. Being that a dictionary has K columns, the computing time consumed by each iteration of the dictionary's update procedure can be calculated as

$$\begin{aligned} &\mathcal{O}(K(Km\frac{N}{m} + (\frac{N}{m})^2(\frac{mS}{K}) + \frac{N}{m}(\frac{mS}{K})^2 + (\frac{mS}{K})^3)) \\ &= \mathcal{O}(K(KN + \frac{N^2S}{mK} + \frac{NmS^2}{K^2} + (\frac{mS}{K})^3)) \\ &= \mathcal{O}(K^2N + \frac{N^2S}{m} + \frac{NmS^2}{K} + \frac{(mS)^3}{K^2}). \end{aligned} \tag{A.18}$$

Usually, the number of atoms K is much larger than the size of the received signal N, the average sparsity S, and the number of patches m, meaning that each iteration of the dictionary's update procedure takes approximately $\mathcal{O}(K^2N)$ time.

REFERENCES

[1] M. Aharon, M. Elad, and A. Bruckstein, *K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation*, IEEE Transactions on Signal Processing, 54 (2006), pp. 4311–4322.

[2] M.S. Asif and J. Romberg, *Fast and accurate algorithms for re-weighted L1-norm minimization*, IEEE Transactions on Signal Processing, 61 (2013), pp. 5905–5916.

[3] T. Blumensath and M.E. Davies, *Normalized iterative hard thresholding: Guaranteed stability and performance*, IEEE Journal of Selected Topics in Signal Processing, 4 (2010), pp. 298–309.

[4] E.J. Candés, L. Demanet, D.L. Donoho, and L. Ying, *Fast discrete curvelet transforms*, Multiscale Modeling and Simulation, 5 (2006), pp. 861–899.

[5] E.J. Candès and T. Tao, *Decoding by linear programming*, IEEE Transactions on Information Theory, 51 (2005), pp. 4203–4215.

[6] S.S. Chen, D.L. Donoho, and M.A. Saunders, *Atomic decomposition by basis pursuit*, SIAM Review, 43 (2001), pp. 129–159.

[7] S. Chen and J. Wigger, *Fast orthogonal least squares algorithm for efficient subset model selection*, IEEE Transactions on Signal Processing, 43 (1995), pp. 1713 –1715.

[8] I. Daubechies, M. Defrise, and C. De Mol, *An iterative thresholding algorithm for linear inverse problems with a sparsity constraint*, Communications on Pure and Applied Mathematics, 57 (2004), pp. 1413–1457.

[9] D.L. Donoho and M. Elad, *Optimally sparse representation in general (non-orthogonal) dictionaries via l1 minimization*, Proceedings of the National Academy of Sciences of the United States of America, 100 (2003), pp. 2197–2202.

[10] M. Elad, *Why simple shrinkage is still relevant for redundant representations?*, IEEE Transactions on Information Theory, 52 (2006), pp. 5559–5569.

[11] ———, *Sparse and Redundant Representations - From Theory to Applications in Signal and Image Processing*, Springer, 2010.

[12] M. Elad, J.-L. Starck, P. Querre, and D.L. Donoho, *Simultaneous cartoon and texture image inpainting using morphological component analysis (MCA)*, Applied and Computational Harmonic Analysis, 19 (2005), pp. 340–358.

[13] K. Engan, S. O. Aase, and J. Hakon Husoy, *Method of optimal directions for frame design*, in IEEE International Conference on Acoustics, Speech, and Signal Processing, vol. 5, 1999, pp. 2443–2446.

[14] M.J. Fadili, J-L Starck, J. Bobin, and Y. Moudden, *Image decomposition and separation using sparse representations: An overview*, Proceedings of the IEEE, 98 (2010), pp. 983–994.

[15] M-J. Fadili, J-L. Starck, M. Elad, and D.L. Donoho, *MCALab: Reproducible research in signal and image decomposition and inpainting*, Computing in Science and Engineering, 12 (2010), pp. 44–63.

[16] I. F. Gorodnitsky and B. D. Rao, *Sparse signal reconstruction from limited data using focuss: a re-weighted minimum norm algorithm*, IEEE Transactions on Signal Processing, 45 (1997), pp. 600–616.

[17] L-W. Kang, C-W. Lin, and Y-H. Fu, *Automatic single-image-based rain streaks removal via image decomposition*, IEEE Transactions on Image Processing, 21 (2012), pp. 1742–1755.

[18] S. Mallat, *A Wavelet Tour of Signal Processing: The Sparse Way*, Elsevier Science, 2008.

[19] S. Mallat and Z. Zhang, *Matching pursuits with time-frequency dictionaries*, IEEE Transactions on Signal Processing, 41 (1993), pp. 3397–3415.

[20] B. K. Natarajan, *Sparse Approximate Solutions to Linear Systems*, SIAM Journal on Computing, 24 (1995), pp. 227–234.

[21] Y. Nesterov, *Gradient Methods for Minimizing Composite Objective Function*, CORE discussion paper: Center for Operations Research and Econometrics, CORE, 2007.

[22] Y. C. Pati, R. Rezaiifar, Y. C. Pati R. Rezaiifar, and P. S. Krishnaprasad, *Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition*, in Proceedings of the 27 th Annual Asilomar Conference on Signals, Systems, and Computers, 1993, pp. 40–44.

[23] G. Peyré, J. Fadili, and J-L. Starck, *Learning the morphological diversity*, SIAM Journal on Imaging Sciences, 3 (2010), pp. 646–669.

[24] J.-L. Starck, M. Elad, and D.L. Donoho, *Redundant multiscale transforms and their application for morphological component analysis*, Advances in Imaging and Electron Physics, 132 (2004).

[25] ———, *Image decomposition via the combination of sparse representations and a variational*

approach, IEEE Transactions on Image Processing, 14 (2005), pp. 1570–1582.
[26] C. Studer and R.G. Baraniuk, *Stable restoration and separation of approximately sparse signals*, Applied and Computational Harmonic Analysis, (to appear 2014).
[27] R. Tibshirani, *Regression shirnkage and selection via the lasso*, Journal of the Royal Statistical Society Seires B (Methodologocal), 58 (1996), pp. 267–288.

TABLE 1
The Adaptive MCA algorithm.

Input	y: The input signal. $\{D_i\}$: The dictionaries describing the morphologies. I_{max}: The maximal number of iterations allowed.
Output	$\{y_i\}$: The morphological components.
	1. **Initialize the components:** For each subcomponent i, $y_i \leftarrow \vec{0}$. $y_i^{'} \leftarrow \vec{0}$. **Loop 1**: Loop $k = 1$ to I_{max}: For each subcomponent i: 2. **Update the coefficient vectors:** $x_i \leftarrow \arg\min_x E_i(y_i, x, D_i)$. ($E_i$ is the energy function in Equation (2.2) for the fixed dictionary or Equation (2.3) for the learned dictionary.) **Loop 2**: Loop until $\{y_i\}$ converge: For each subcomponent i, perform the following steps: 3. **Calculate the i-th residual:** $r_i \leftarrow y - \sum_{j \neq i} y_j^{'}$. 4. **Update the i-th component:** For the fixed dictionaries: $y_i \leftarrow \arg\min_f \|r_i - f\|_2^2 + \mu \|f - D_i x_i\|_2^2$. For the learned dictionaries: $y_i \leftarrow \arg\min_f \|r_i - f\|_2^2 + \mu' \sum_{j=1}^{m} \|R_j(f) - D_i x_i^j\|_2^2$. 5. **Store the components:** $\{y_i^{'}\} \leftarrow \{y_i\}$. **End Loop 2**. 6. **Update the learned dictionary:** For each learned dictionary D_i, $D_i \leftarrow \arg\min_D E_i(y_i, x_i, D)$. **End Loop 1**.

TABLE 2
The Proposed Algorithm.

Input	y: The input signal. $\{D_i\}$: The dictionaries describing the morphologies. I_{max}: The maximal number of iterations allowed.
Output	$\{y_i\}$: The morphological components.
	1. **Initialize the components:** For each subcomponent i; let $y_i \leftarrow \vec{0}$; $y_i^{'} \leftarrow \vec{0}$. compute the weighting factors according to Equation (3.14). **Loop 1**: Loop $k = 1$ to I_{max}: **Loop 2**: Loop until $\{x_i\}$ and $\{y_i\}$ converge: 2. **Update the coefficient vectors:** For each subcomponent i: $x_i \leftarrow \arg\min_x E_i^W(y_i, x_i, D_i)$. ($E_i^W$ is defined in Equation (3.17).) **Loop 3**: Loop until $\{y_i\}$ converge: 3. **Update the components:** For each subcomponent i, perform the following: If D_i is the fixed dictionary: $y_i \leftarrow \arg\min_f \|y - \sum_{j\neq i} y_j^{'} - f\|_2^2 + \mu\|f - D_i x_i\|_2^2$. If D_i is the learned dictionary: $y_i \leftarrow \arg\min_f \|y - \sum_{j\neq i} y_j^{'} - f\|_2^2 + \mu\sum_{j=1}^{m}\|R_j(f) - D_i x_i^j\|_2^2$. 4. **Store the components:** $\{y_i^{'}\} \leftarrow \{y_i\}$. **End Loop 3**. **End Loop 2**. 5. **Update the learned dictionary:** For each learned dictionary D_i, update D_i and x_i to satisfy Equation (3.21). 6. **Update the coherence vectors:** Update the weighting factors according to Equation (3.14). **End Loop 1**.

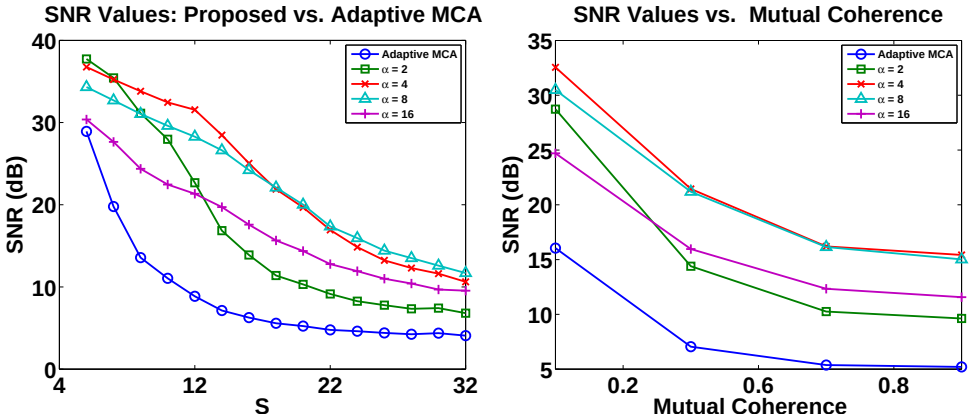

FIG. 1. *The average SNR values of the two components reconstructed using adaptive MCA and our method. Left: The horizontal axis indicates the sparse level S of the subcomponents used to synthesize the input signal. The size of each input signal is* 128. *At each value of S,* 200 *trails of input signals are separated. Our method outperforms adaptive MCA in every case. Right: The mutual coherence calculated with atoms in the supports of the two subcomponents is collected and plotted against SNR. Each measurement point is the average SNR obtained with trails of similar mutual coherence. SNR decreases when the mutual coherence increases.*

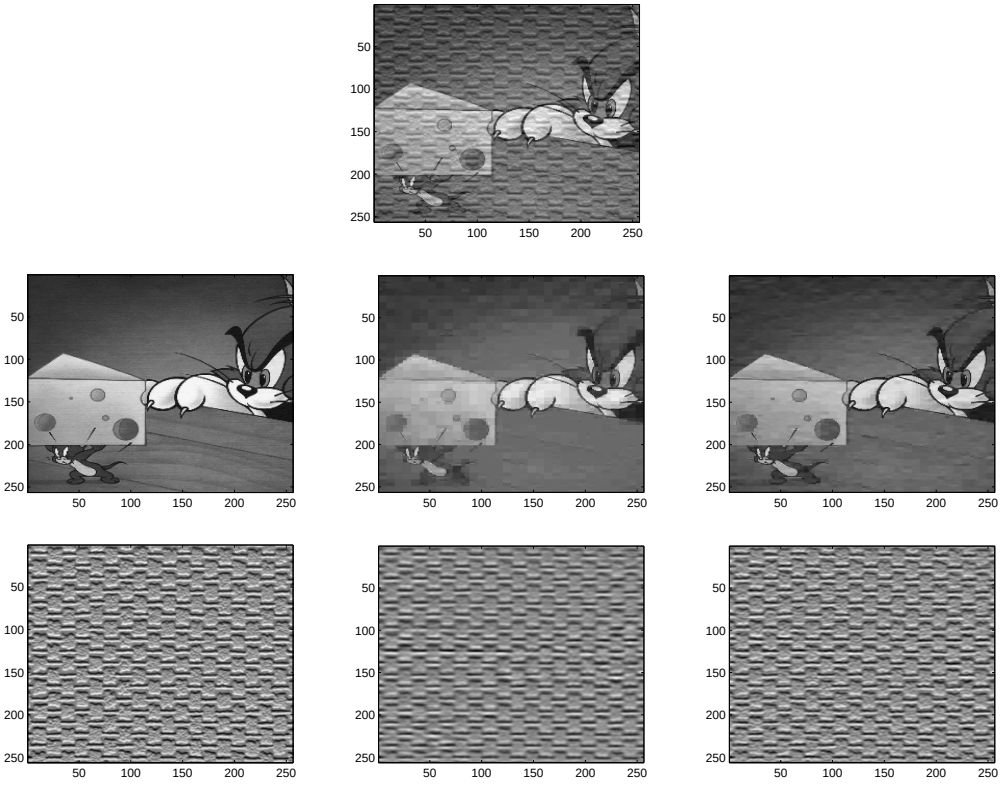

FIG. 2. *First row: The composite of a cartoon and a texture image. Second row: From left to right, the cartoon image is approximately sparse with respect to the wavelets, the subcomponent reconstructed using adaptive MCA (PSNR is 31.1929 dB), and that obtained by using the proposed method (PSNR is 37.2473 dB). Third row: From left to right, the texture image is approximately sparse to the DCT, the subcomponent reconstructed using adaptive MCA (PSNR is 31.7873 dB), and that reconstructed by the proposed method (PSNR is 37.2473 dB).*

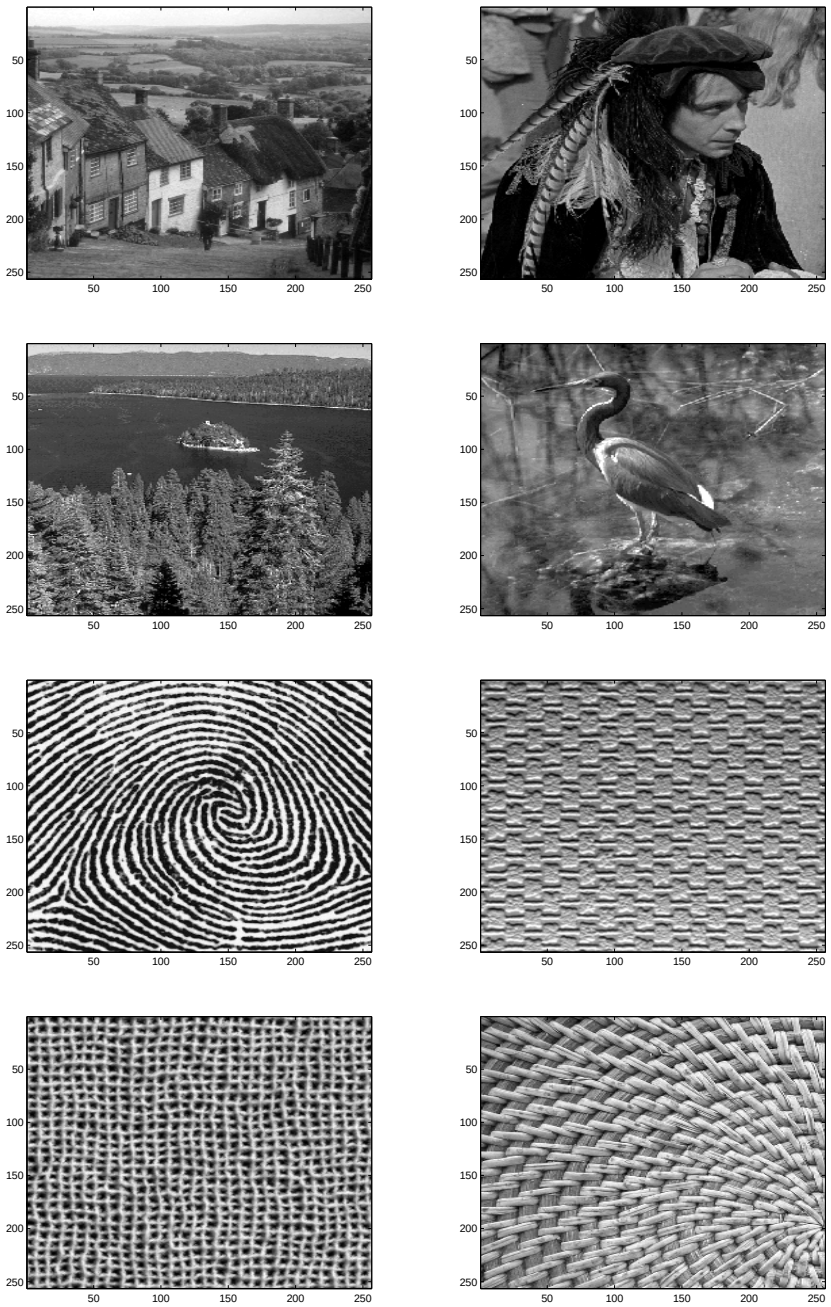

FIG. 3. *The images used in our experiments. The images in the top two rows are approximately sparse to the wavelets. The dictionaries of the texture images in the bottom two rows were learned using K-SVD.*

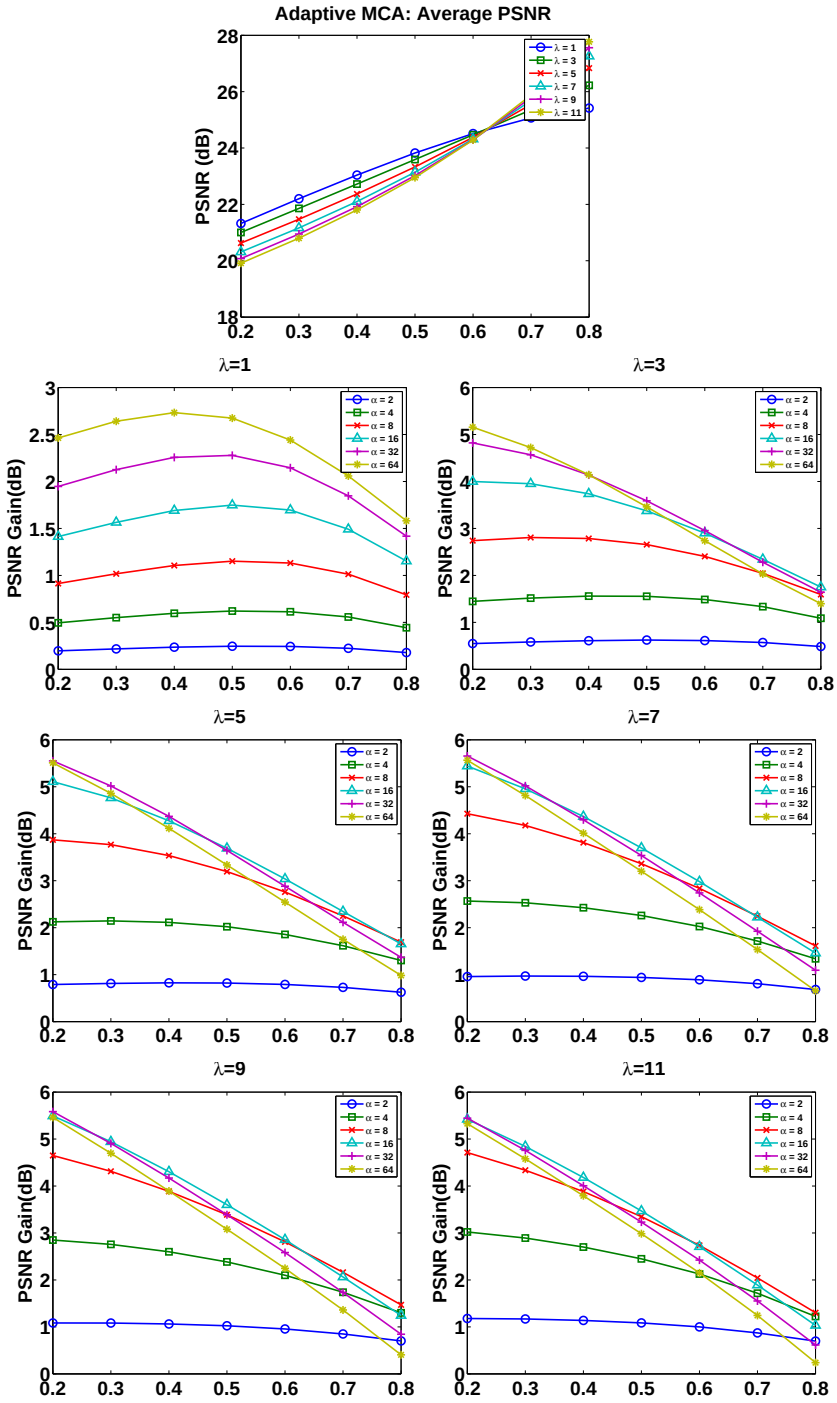

FIG. 4. *Horizontal axis: β value. Top: The performance of adaptive MCA. The second and third rows show the average PSNR gain of our method over adaptive MCA.*

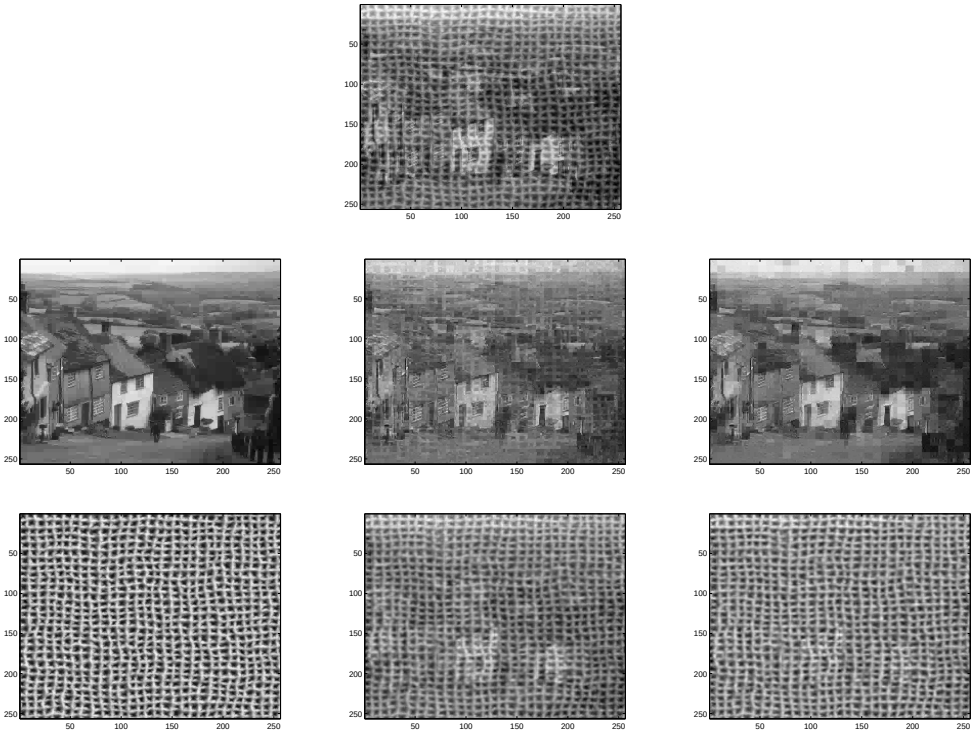

FIG. 5. *Comparison of the visual quality of the reconstructed subcomponents. First row: The composite image. Second row: From left to right, the original image, the subcomponents reconstructed by adaptive MCA (PSNR is* 24.8149 *dB), and that obtained by the proposed method (PSNR is* 28.8784 *dB). Third row: From left to right, the original texture image, the subcomponents reconstructed by adaptive MCA (PSNR is* 24.8146 *dB), and that reconstructed by the proposed method (PSNR is* 28.8795 *dB).*

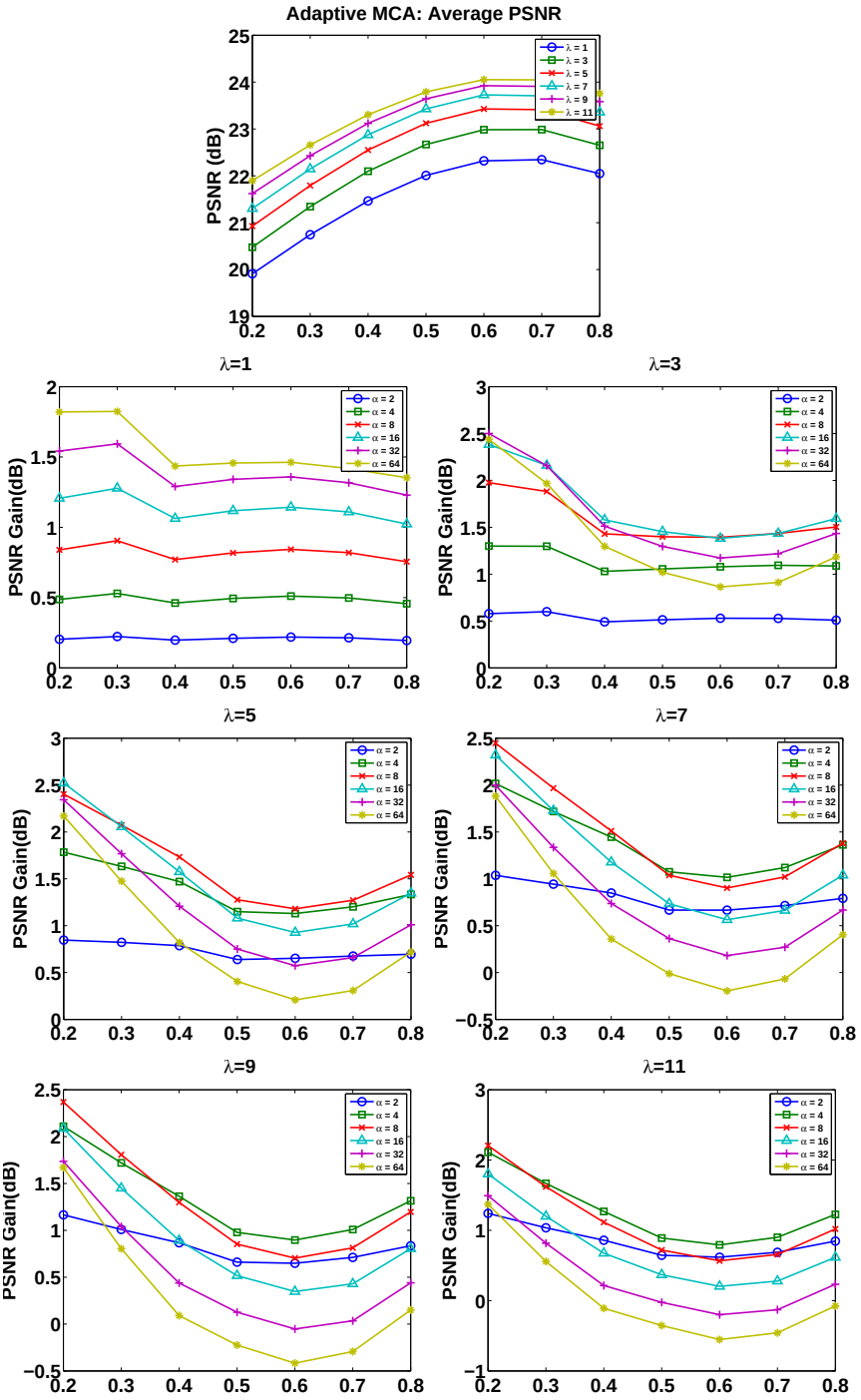

FIG. 6. *Horizontal axis: β value. Top: The performance of adaptive MCA. The second and third rows show the average PSNR gain of our method over adaptive MCA.*

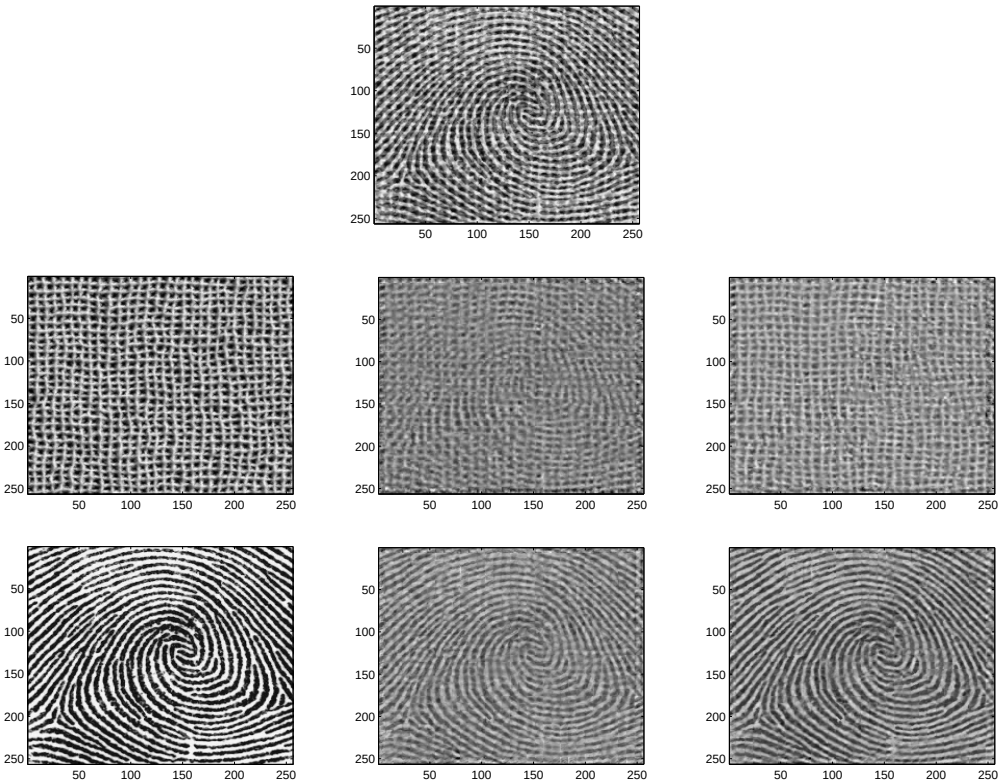

FIG. 7. *First Row: The composite of two texture images. Second row: From left to right, the original texture image, the subcomponent reconstructed using adaptive MCA (PSNR is* 18.6648 *dB), and that obtained by the proposed method (PSRN is* 20.8336 *dB). Third row: From left to right, the fingerprint image, the subcomponent reconstructed using adaptive MCA (*21.3252 *dB), and that reconstructed by the proposed method (*23.6710 *dB).*