Blindly Separating Mixtures of Multiple Layers with Spatial Shifts

Kun Gai[1], Zhenwei Shi[2], Changshui Zhang[1]
[1] State Key Laboratory on Intelligent Technology and Systems,
Tsinghua National Laboratory for Information Science and Technology (TNList),
Department of Automation,Tsinghua University, Beijing 100084, China.
[2] Image Processing Center, School of Astronautics,
Beijing University of Aeronautics and Astronautics, Beijing 100083, China.
gaik02@mails.thu.edu.cn, shizhenwei@mail.thu.edu.cn, zcs@mail.thu.edu.cn.

Abstract

We address the problem of blindly separating mixtures of multiple layer images with unknown spatial shifts and mixing coefficients. Our proposed method can handle the over-determined, determined and under-determined cases where mixtures are more than, as many as and fewer than layers, respectively. The method is fast in over-determined and determined cases, with the same complexity as the fast Fourier transform (FFT), and can separate more layers from fewer mixtures in the under-determined case. It consists of two main steps. First, a novel sparse blind separation algorithm is applied, to estimate the spatial shifts, the mixing coefficients and the edge image of each layer. Second, all layers are reconstructed, by large scale linear programming in the under-determined case, or by least-squares solutions in other cases. The effectiveness of this technology is shown in the experiments on two simulated mixtures of four layers with spatial shifts, real mixture photos containing transparency and reflections, and real mixture images in a dissolve from a video.

1. Introduction

When one takes some photos through a window, he or she often has a problem caused by the glass: the photos are *mixtures* of two *layer* images, one of which is the transmitted scene behind the window and the other is the scene reflected by the window. Then it is hard to distinguish the scene of interest. Similar problems can also be found in videos because of popular video editing technologies such as dissolve and slide-out/in. Both human perception and many computer vision algorithms, such as texture acquisition, object tracking and object recognition, can be seriously disturbed by the mixture images. Thus, the natural need is to separate these mixtures and restore the component layers.

The above problems can be formally described as follows. Suppose there exist several source layers, and we observe some of their mixtures. Then our goal is to recover the original layers from the mixtures. In different mixtures, the properties of each layer may change. Here we consider two kinds of properties. The first one is the layer's intensity, which may vary when lighting conditions or camera's settings change or photos are acquired through polarization filters (or because of dissolves in videos). We characterize this variation by the mixing coefficients. The other is the layer's position, which may be different by reason of the movements of the camera or objects (or by the slide-out/in effect in videos). This change is described by spatial shifts.

The problem of separating mixtures is challenging when the component layers have both unknown spatial shifts and varying mixing coefficients. Furthermore, if the number of layers is large, even larger than the number of mixtures, the problem will be particularly complicated. A number of approached have been proposed to separate two-layer mixtures containing transparency and reflections. When only one mixture is available, automatic separation is quite difficult because it is massively ill-posed (although Levin *et al*. attempted it on simple mixtures [9] and then Levin and Weiss [8] developed a two-layer separation system with user's assistances, the system is not automatic). However, when two or more mixtures are available, each slightly different, automatic separation can be achieved "by proper exploitation of the diversity in different mixtures" [1]. Some methods acquire different mixtures by diverse polarization [2, 4, 5, 16, 14] or focusing differently [13, 15]. They assume each layer is static. Other methods [11, 12, 18, 19] exploit relative motions of layers. They assume each layer (or at least the layer with the dominant motions) has fixed mixing coefficients. Be'ery and Yeredor proposed 2D-ACDC [1] to handle both unknown spatial shifts and varying mixing coefficients. Unfortunately, it has a problem of local optima and needs a good initial guess of spatial shifts.

978-1-4244-2243-2/08/$25.00 ©2008 IEEE

Besides, it can only handle the case of two mixtures with two layers.

We develop a method to blindly separate mixtures consisting of multiple layers with unknown spatial shifts and mixing coefficients. Our method has a number of desirable properties. (1) Both unknown spatial shifts and varying mixing coefficients can be handled. (2) It can deal with mixtures of multiple (including two and more) layers. (3) The algorithm is computationally fast when mixtures are no fewer than layers, with the same complexity as the fast Fourier transform (FFT). (4) More layers can be separated from fewer mixtures. (5) It does not need any initial guess of spatial shifts or mixing coefficients. Thus, it is more reliable than 2D-AC-DC.

Depending on the numbers of mixture images and layer images, the problem of separation can be divided into three categories: the over-determined, determined and under-determined problems where the number of mixtures is larger than, equal to and smaller than the number of layers, respectively. Our proposed method can handle problems of all the three categories. It has two main steps. (1) A novel blind separation algorithm, named sparse blind separation of mixtures with unknown spatial shifts (SP-BSS), is applied to estimate all parameters in the mixing model and the edge images of all layers. (2) Layers are reconstructed by different algorithms: a fast one by least-squares solutions for over-determined and determined cases, and another one by linear programming for the under-determined case.

The rest of this paper is organized as follows. Section 2 presents the problem formulation, including the mixing model and the sparsifying step. Section 3 proposes SP-BSS algorithm that estimates spatial shifts and mixing coefficients in two mixtures, and its extension to more mixtures is given in section 4. Then, we discuss two reconstruction methods in section 5. Section 6 shows the experiments on two simulated mixtures of four layers and real-world mixtures containing transparency and reflections, as well as real mixtures in a dissolve. Finally, we close with a conclusion.

2. Problem formulation

As discussed in section 1, in different mixtures, the same layer probably has unknown spatial shifts and unknown mixing coefficients. Here we assume linear mixing as in many other methods (*e.g.*, [2, 12, 16, 18]), and formulate the mixing model of m mixtures with n layers as:

$$I_i(x) = \sum\nolimits_{j=1}^{n} \bar{a}_{ij} \bar{L}_j^{\bar{s}_{ij}}(x), \qquad i = 1, \cdots, m, \qquad (1)$$

where I_i $(i = 1, \cdots, m)$ is the ith mixture, $\bar{L}_j$ $(j = 1, \cdots, n)$ is the jth layer, x is a 2D integer vector that represents the pixel location, and $\bar{a}_{ij}$ and superscript $\bar{s}_{ij}$ are the mixing coefficient and the spatial shift (also being a 2D integer vector) of layer $\bar{L}_j$ in mixture I_i, respectively. Actually, $\forall i, j,\ \bar{L}_j^{\bar{s}_{ij}}(x)$ is $\bar{L}_j(x - \bar{s}_{ij})$. Both $\bar{a}_{ij}$ and $\bar{s}_{ij}$ $(i = 1, \cdots, m,\ j = 1, \cdots, n)$ are called model parameters. For simplicity, all these parameters are described by a mixing matrix $\mathbf{A}$, a horizontal shift matrix $\mathbf{S}_h$ and a vertical shift matrix $\mathbf{S}_v$, where $\mathbf{A}(i, j)$ is $\bar{a}_{ij}$, and $\mathbf{S}_h(i, j)$ and $\mathbf{S}_v(i, j)$ are two elements of $\bar{s}_{ij}$, respectively.

Without loss of generality, the mixing model (1) can be rewritten as:

$$\begin{cases} I_1(x) = \quad L_1(x) \quad + \cdots + \quad L_n(x), \\ I_2(x) = a_{21} L_1^{s_{21}}(x) + \cdots + a_{2n} L_n^{s_{2n}}(x), \\ \quad \vdots \\ I_m(x) = a_{m1} L_1^{s_{m1}}(x) + \cdots + a_{mn} L_n^{s_{mn}}(x), \end{cases} \qquad (2)$$

where $\forall i, j$, $L_j = \bar{a}_{1j} \bar{L}_{1j}^{s_{1j}}$, $a_{ij} = \frac{\bar{a}_{ij}}{\bar{a}_{1j}}$ and $s_{ij} = \bar{s}_{ij} - \bar{s}_{1j}$.

Our task is to separate the layers L_j $(j = 1, \cdots, n)$ from the mixtures I_i $(i = 1, \cdots, m)$. Here only the mixtures I_i $(i = 1, \cdots, m)$ are known, and the model parameters are unknown. This is a typical problem of blind source separation [3], but has not been well addressed on 2D image mixtures with unknown spatial shifts in addition to mixing coefficients. We transform the problem (2) to another one by sparsifying images.

Recently several researchers [2, 8, 9] have noticed the fact that when an edge filter (usually a derivative filter) is applied to natural images, "the outputs of the filter tend to be sparse" [8]. The sparseness denotes that "only a small number" of the pixels in images "are significantly different from zero" [2]. The pixels significantly different from zero are called *significant* pixels, while others are called *non-significant* pixels. When a proper linear shift-invariant edge filter is applied to mixtures, the mixing problem (2) is transformed to a new one with the same model and model parameters, and the corresponding sparse mixtures and sparse layers. For simplicity, the mixing model (2) is still used to describe the new problem of sparse data.

In our experiments we use the vertical or horizontal first derivative filter as the edge filter. An example is shown in Fig. 1. Images in column 2 are mixtures of images in column 1, with the model parameters $\mathbf{A}$, $\mathbf{S}_h$ and $\mathbf{S}_v$ equal to

$$\begin{bmatrix} 0.3 & 0.45 & 0.25 \\ 0.2 & 0.25 & 0.45 \\ 0.4 & 0.3 & 0.3 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 \\ -7 & 15 & 15 \\ 6 & 20 & -10 \end{bmatrix} \text{ and } \begin{bmatrix} 0 & 0 & 0 \\ 6 & 12 & 12 \\ -9 & 15 & -6 \end{bmatrix}, \qquad (3)$$

respectively. Note that the largest spatial shift is $(20, 15)$, which is large relative to the size of images (128×128). Images in column 3 are the edges of mixtures by the vertical first derivative filter. They are sparse, as most pixels of them are black. Images in column 4 are reconstructed results, by our proposed method.

3. Sparse blind separation on two mixtures

This section assumes that the number of mixtures is equal to two, and presents a sparse blind separation method

Figure 1. Three 128×128 layers (column 1): a cartoon girl, Lena and a billiard table. Their mixtures with different spatial shifts and mixing coefficients (column 2). Edge images of the mixtures (column 3) and the reconstructed results (column 4).

that can estimate all the model parameters of n layers from two mixtures. Its extension to more than two mixtures will be discussed in the next section.

3.1. Specification

For mathematical convenience, we give the following hypotheses:

1. Sparsity: $\forall i$, L_i is sparse, and the number of the significant pixels of L_i is denoted by N_i^{sgf}.
2. Independency: $\forall$ shift v, when $i \neq j$, the locations of the significant pixels in L_i and L_j^v are statistically independent.
3. Non-periodicity: $\forall i$, when $v \neq 0$, $N_i^{prd}(v) \ll N_i^{sgf}$. $N_i^{prd}(v)$ is the number of spatial locations where both L_i and L_i^v are significant.

The sparsity is introduced by the edge filter [8]. Since different layers derive from different scenes, the independency can be satisfied in most cases (Usually the filtered layers output by the edge filter are more statistically independent than the original layers [6].). The third hypothesis denotes that the shape of the edges in any layer has no strongly periodic structure, and guarantees that the solution of the spatial shifts is unique up to the order of the layers (The order can not be determined by the mixing model as in Independent Component Analysis [7].).

In order to search for the correct spatial shifts in the second mixtures, we move the second mixture towards the negative direction with a searching shift v. Then the searching model of 2 mixtures with n layers is written by:

$$\begin{cases} I_1(x) = \sum_{i \neq j} L_i(x) + L_j(x) \\ I_2^{-v}(x) = \sum_{i \neq j} a_{2i} L_i^{s_{2i}-v}(x) + a_{2j} L_j^{s_{2j}-v}(x) \end{cases}, \quad (4)$$

where two terms respecting layer L_j are listed separately for analysis. There will be different properties depending on whether v is equal to s_{2j}. Since most pixels in the layers (including the shifted layers) are nonsignificant and the locations of significant pixels in the different layers are statistically independent, there is a high probability that: where L_j is significant, only layer L_j contributes to the mixture pixel. It means for most $x \in B_j \doteq \{x | L_j(x) \text{ is significant}\}$,

$$I_1(x) = L_j(x), \text{ and } I_2^{-v}(x) = a_{2j} L_j^{s_{2j}-v}(x). \quad (5)$$

The case will be different depending on the relationship between s_{2j} and v. On one hand, if $s_{2j} = v$, then $L_j^{s_{2j}-v}(x) = L_j(x)$. Thus, for most $x \in B_j$, $I_2^{-v}(x) = a_{2j} L_j(x) = a_{2j} I_1(x)$. In the scatter plot of I_1 *vs.* I_2^{-v}, the corresponding points will cluster along the line whose slope is equal to a_{2j}. On the other hand, if $s_{2j} \neq v$, by reason of the hypothesis of non-periodicity, for most $x \in B_j$, $L_j^{s_{2j}-v}(x) = 0$, and $I_2^{-v}(x) = 0$. Thus in the scatter plot of I_1 *vs.* I_2^{-v}, the corresponding points will cluster along the horizontal axis. Because of the significant pixels in $L_j^{s_{2j}-v}$, there will be also a cluster along the vertical axis. For simplicity, if the spatial shift s_{2j} is equal to the searching shift v, then we call the corresponding layer L_j is matched by v. The above is summarized by follows. *In the scatter plot of* I_1 vs. I_2^v*: if a given layer is matched by* v*, the corresponding points will cluster along a line with a slope equal to the layer's mixing coefficient; otherwise, the corresponding points will cluster along two axes.*

Assume there are k $(0 \leq k \leq n)$ layers matched by a given searching shift v. In the scatter plot of I_1 *vs.* I_2^{-v}, the matched layers will cause k clusters along k lines with slopes equal to the matched layers' mixing coefficients. These k lines are called "*feature lines*". Besides, the unmatched layers will cause 2 additional clusters along two axes, and there will be $k+2$ clusters along $k+2$ lines in all. When we map the points of I_1 *vs.* I_2^{-v} to the angular space by $\arctan\left(I_1(x)/I_2^{-v}(x)\right)$, in the corresponding angular density plot there will be $k+3$ peaks (Clusters along axes will cause 3 peaks at: $-\frac{\pi}{2}$, 0 and $\frac{\pi}{2}$). For simplicity, the angle where there exists a peak is called "*peak angle*". Two angular density plots are shown in Fig. 2: when there are one or two matched layer(s), in the angular density plot there are the same number of clear peak(s) besides 3 peaks at axis angles.

The clustering algorithm can be used to detect the feature line(s). When we have a guess, denoted by g, on the number of matched layers, the "*line clustering*" algorithm with respect to shift v is as follows.

1. Set $D = \{(I_1(x), I_2^{-v}(x))\}$. Remove the points near to the original point from D, and map the remaining points to the angle space: $A^v = \{a(x) = \arctan\left(I_2^{-v}(x)/I_1(x)\right) | (I_1(x), I_2^{-v}(x)) \in D\}$.

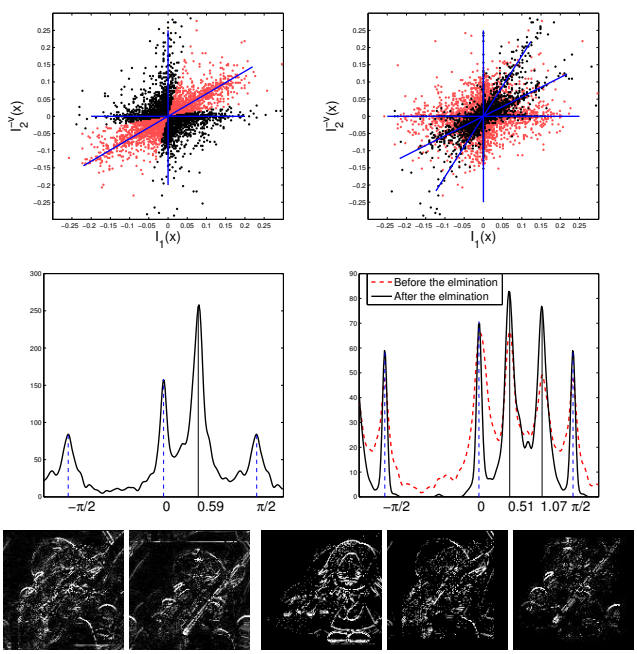


Figure 2. Illustration (mixture I_1 and I_2 are the images in top 1 and 2 in column 3 of Fig. 1). The scatter plot of I_1 *vs.* $I_2^{-(-7,6)}$ when one layer is matched (Left top) and the corresponding angular histogram (Left medium). The scatter plot of I_1 *vs.* $I_2^{-(15,12)}$ when the other two layers are matched (Right top) and the corresponding angular histogram (Right medium, peaks are more clear after the elimination). Two mixtures after the first matched layer (the cartoon girl) is eliminated (Left 1 and 2 in the bottom row). Three extracted layers from I_1 and I_2 (Left 3, 4, and 5 in the bottom row).

2. When the n layers are all matched by v, there is no cluster along axes. To handle all situations, we add A^v and many angles of $-\frac{\pi}{2}$, 0 and $\frac{\pi}{2}$ to form A_{ext}. Setting the cluster number equal to $g+3$, implement a basic clustering algorithm (such as k-means [10] or spectral clustering [17]) on the angular data in A_{ext}, and get the label of each $a(x)$ ($a(x) \in A_{ext}$), denoted by $l(x)$.

3. Calculate each cluster center (the same as k-means)

$$\mu_j = \Big(\sum\nolimits_{l(x)=j} a(x)\Big) / \Big(\sum\nolimits_{l(x)=j} 1\Big), \quad 1 \le j \le g+3, \tag{6}$$

and the sum-of-squared error

$$J^v(g) = \sum\nolimits_{j=1}^{g+3} \sum\nolimits_{l(x)=j, a(x)\in A^v} (a(x)-\mu_j)^2. \tag{7}$$

Whether our guess g is equal to the correct number (k) of the matched layer(s) can be reflected by the sum-of-squared error $J^v(g)$. If $g \ge k$, for each peak angle there will be at least one cluster center very close to it, and $J^v(g)$ will be small. If $g < k$, there will exist at least one peak angle without any very close cluster center, and $J^v(g)$ will be significantly large.

If $g = k$, as discussed, the $k+3$ cluster centers will be very close to $g+3$ peak angles. After we remove the three centers close to axis angles, the remaining k centers are the estimated angles of the feature lines. Then mixing coefficients can be gotten by calculating the slopes of the feature lines. The key problem is to determine the correct number of the layers matched by v, and will be discussed in section 3.3.

If $g \ge k$, for each peak angle there will be at least one cluster center very close to it. Thus, different peak angles will not be in the same cluster. In the space of I_1 *vs.* I_2^{-v}, the points that are in the same cluster with the points on axes will not contain any point on feature lines, and hence are recognized as "*unmatched*" points. Other points are recognized as "*matched*" points. These procedures are feasible without k known (*e.g.*, we set g equal to n), and are named as "*recognizing matched points*". An example is shown in the left top plot of Fig. 2, where the black points are unmatched points, and the gray (red in the electronic copy) points are matched points. These procedures will be used to eliminate matched layers in section 3.2.

In the next two subsections, we first search for n shift candidates to include all correct spatial shifts, and then determine the correct number of matched layers in each shift candidate to get all model parameters.

3.2. Searching for shift candidates

To search for spatial shifts, we define the indication function as:

$$\mathrm{idc}(v) = \sum\nolimits_x \left| I_1(x) I_2^{-v}(x) \right|, \tag{8}$$

which is small when there are only clusters along axes in the scatter plot of I_1 *vs.* I_2^{-v} (the product $\left|I_1(x)I_2^{-v}(x)\right|$ is zero when $I_1(x)$ or $I_2^{-v}(x)$ is zero), and is large when there is any cluster along lines other than axes. Thus this function can be used to search for spatial shifts in the mixing model.

In order to get the spatial shifts in the mixing model, we search for n different shifts that make the indication function values larger than others (The shift are 2D integer vector, and the number of all available shifts is 4 times of the pixel number). If the spatial shifts of any two layers are different, all the searching results will be correct spatial shifts. However, if some of the layers have the same spatial shift, the searching results will be n shift candidates that include all the spatial shifts in the mixing model.

When we find any shift candidate, denoted by c, the significant pixels belonging to the matched layers can be picked out by *recognizing matched points*. Thus we can eliminate these pixels to avoid that the matched layers influence the following searching, *i.e.*, if $(I_1(x), I_2^{-c}(x))$ is recognized as a matched point by *recognizing matched points*, then we set $I_1(x)$ and $I_2(x+c)$ equal to zero. By doing this, we find n shift candidates one by one, and eliminate the matched layers once finding a candidate.

The elimination technology is a key step to make our algorithm robust. Taking the first two images of column 3 in Fig. 1 as two mixtures (I_1 and I_2), we show an example of the searching procedures in Fig. 2. Our first searching result is $(-7, 6)$. In the corresponding scatter plot (Left top) the matched points are plotted by gray (red) points, and the unmatched points are plotted by black points. The effects of eliminating matched points can be seen in the remaining mixtures (Left 1 and 2 in the bottom row of Fig. 2. See the original mixtures in Fig. 1 for contrast.), where the edges of the cartoon girl layer have been removed, while most edges of other two layers (Lena and the billiard table) remain. Our second searching result is $(15, 12)$. Note that in the scatter plot of I_1 *vs.* $I_2^{-(15,12)}$ (Right top) the feature lines are more distinct after the elimination (Only the black points) than before the elimination (All points). In the corresponding angular density plot (Right medium), the peaks are clearer after the elimination (Solid line) than before the elimination (Dashed line, justified to the same scale). Because the elimination removes the influence of already matched layers on the following searching, it makes the searching process more robust.

3.3. Estimating all the model parameters

Given n shift candidates, denoted by $c(i)$ $(1 \le i \le n)$, the next problem is how to distribute n layers to these shift candidates. Assume that the correct number of the layers matched by each shift candidate $c(i)$ is $d(i)$, and our guess on $d(i)$ is $\tilde{d}(i)$. Then consider the minimizing problem of the global loss function $L(\tilde{d}(1), \cdots, \tilde{d}(n))$:

$$F = \min_{\tilde{d}(i)} L(\tilde{d}(1), \cdots, \tilde{d}(n)) = \min_{\tilde{d}(i)} \sum_{i=1}^{N} J^{c(i)}(\tilde{d}(i)), \quad (9)$$
$$\text{s. t. } \sum\nolimits_{i=1}^{n} \tilde{d}(i) = n, \text{ and } \forall i,\ \tilde{d}(i) \in \mathbb{Z}^+ \cup \{0\},$$

where $J^{c(i)}(\tilde{d}(i))$ is the sum-of-squared error in Eq. (7). Recall the influences of our guess $\tilde{d}(i)$ on $J^{c(i)}(\tilde{d}(i))$. If our guesses are right, $\forall i,\ \tilde{d}(i) = d(i)$, and $J^{c(i)}(\tilde{d}(i))$ will be small, leading to a small $L(\tilde{d}(1), \cdots, \tilde{d}(n))$. If there is any wrong guess, $\exists l,\ \tilde{d}(l) < d(l)$, and $J^{c(l)}(\tilde{d}(l))$ will be significantly large, leading to a large $L(\tilde{d}(1), \cdots, \tilde{d}(n))$. Consequently, the correct number of matched layer(s) by each shift candidate can be given by the optimal solution of the above problem.

Minimizing the global loss function in (9) can be solved by dynamic programming. Consider the subproblem:

$$F_q(k) = \min_{\tilde{d}(i)} \sum\nolimits_{i=1}^{q} J^{c(i)}(\tilde{d}(i)), \text{ s. t. } \sum\nolimits_{i=1}^{q} \tilde{d}(i) = k, \quad (10)$$

which denotes the minimal sum when we distribute k layers to the first q shift candidates. It can be iteratively decomposed by itself until q decreases to 1. Thereby, the iterative formulas are:

$$\text{Initialization: } F_1(k) = J^{c(1)}(k) \quad (11)$$
$$\text{Iteration: } F_q(k) = \min_{i \in \{0, \cdots, k\}} (F_{q-1}(k-i) + J^{c(q)}(i)) \quad (12)$$
$$\text{Termination: } F = F_n(n) \quad (13)$$

which can be used to find the minimal value of the global loss function in (9). Record the optimal solution of each minimizing problem in (12) when calculating, and finally track back to get the whole optimal solution of (9).

Now we know the number of the layers matched by each shift candidates. The remaining is to implement the line clustering algorithm with respect to each shift candidate that matches layer(s), with the correct cluster number. The mixing coefficient is given by the $\arctan$ value of the center (or the densest angle) of each cluster not at axis angles.

3.4. Extracting sparse layers

Because of the sparsity and independence, most of significant pixels in any mixture are contributed by only one layer. Consequently, we can approximately estimate all layers by assigning pixels of mixtures to the n layers. Now the spatial shift s_{2j} and the mixing coefficient a_{2j} of each layer L_j $(1 \le j \le n)$ have been estimated. Based on preceding analysis, for any location x, if $L_j(x)$ is significant, there will be a high prolixity that $\left(I_1(x), I_2^{-s_{2j}}(x)\right)$ is on the feature line with a slope equal to a_{2j}. This property can be used to decide the assignation. For any given location x, we calculate the angular difference between the corresponding point and the feature line of the jth layer:

$$\text{angle}(x, j) = \left| \arctan\left(I_2^{-s_{2j}}(x)/I_1(x)\right) - \arctan(a_{2,j}) \right|, \quad (14)$$

choose the layer index $\text{idx}(x)$ that minimizes $\text{angle}(x, j)$:

$$\text{idx}(x) = \arg\min\nolimits_{j \in \{1, \cdots, n\}} \text{angle}(x, j), \quad (15)$$

and finally assign the pixel to layer $\tilde{L}_{\text{idx}(x)}$:

$$\tilde{L}_{\text{idx}(x)}(x) =: \quad I_1(x) \quad \text{or} \quad \frac{I_1(x) + a_{2\,\text{idx}(x)} I_2^{-s_{2\,\text{idx}(x)}}(x)}{1 + a_{2\,\text{idx}(x)} \cdot a_{2\,\text{idx}(x)}}, \quad (16)$$

where $\tilde{L}_j$ is the extracted layer $(1 \le j \le n)$. In the extracted layers the pixels without any assignation are set equal to zero. Three sparse layers extracted from two mixtures are shown in Fig. 2 (Left 3, 4 and 5 in the bottom). The extracted sparse layers will be used to extend our algorithm to more mixtures.

Note that with the sparse edges output by any edge filter, the above procedures can extract the corresponding sparse layers. Given an edge filter e_k, we use E_j^k $(1 \le j \le n)$ to denote the corresponding extracted layer. These edges will help us reconstruct the original non-sparse layers in which case mixtures are fewer than layers.

3.5. Complexity analysis

First let us clarify the denotations used in this paragraph: the number of pixels in each mixture is denoted by N, the number of significant pixels in mixtures is denoted by N_s, and the number of layers is denoted by c_l. Now we consider the whole processes of our algorithm. The computational complexity of maximizing the indication function for one time is $O(N \log N)$, because the indication function is actually the correlation function and can be accelerated by FFT. Thus finding c_l shift candidates has a complexity of $O(c_l N \log N)$. Then the line clustering algorithm with respect to each shift candidate need to be applied on n_s angular data for c_l times, to give all the sum-of-squared errors. The complexity of all clustering processes is $O(c_l^2 C(N_s))$, where $C(\cdot)$ denotes the complexity of the basic clustering algorithm applied in our line clustering algorithm. Finally, the complexity of dynamic programming is $O(c_l^3)$. Consequently, the whole complexity of our algorithm is $O(c_l N \log N + c_l^2 C(N_s) + c_l^3)$. In fact, N_s is far smaller than N because of the sparsity, and c_l is a constant and small comparable to N. Therefore the whole complexity is $O(N \log N)$, which is equal to the complexity of FFT. In our experiments we use k-means in the line clustering, and most computational time is occupied by FFT for every run, consistent with our analysis.

Now the sparse-data-based blind separation on two mixtures has been addressed, and its natural extension to more mixtures will be discussed in the next section.

4. Extending to more mixtures

When more than two mixtures are available, we transform the problem of separation on m mixtures to $m-1$ problems of separation on 2 mixtures. Given m mixtures, denoted by $I_1, \cdots, I_m$, we apply the sparse blind separation algorithm on two mixtures for $m-1$ times, each time on I_1 and another mixture I_j ($2 \le j \le m$), and get $m-1$ result sets. Thus all the model parameters in the mixing model of m mixtures have been gotten. However, when we put these model parameters together, a problem is caused by the ambiguity of the order in the blind separation: model parameters of different layers may be wrongly matched, *i.e.*, be put to the same column in the parameter matrix.

We use the correlation between extracted layers to match the corresponding parameters. Then all the parameters of m mixtures without any wrong match can be gotten.

5. Reconstruction of original layers

5.1. Over-determined/determined reconstruction

With all the model parameters known, the reconstruction of original non-sparse layers is relatively easy in which case the mixtures are no fewer than the layers. When the mixing model (2) is transformed into frequency domain, it is linear at any given frequency $u = (u_1, u_2)$:

$$F_I(u) = \mathbf{M}(u) F_L(u), \tag{17}$$

where $F_I(u) = (ft(I_1)(u), \cdots, ft(I_m)(u))^T$, $F_L(u) = (ft(L_1)(u), \cdots, ft(L_n)(u))^T$, $ft(\cdot)$ denotes the Fourier transform, and $\mathbf{M}(u)$ is the frequent mixing matrix. The element in row k and column l of $\mathbf{M}(u)$ is $\mathbf{A}(k,l) \cdot \exp\{-2\pi j(\mathbf{S}_h(k,l)u_1/W + \mathbf{S}_v(k,l)u_2/H)\}$, in which j satisfies j^2 is equal to -1, and H and W are the height and width of images. The least-squares solution of (17) is:

$$\tilde{F}_L(u) = \left(\mathbf{M}^*(u)\mathbf{M}(u)\right)^{-1}\mathbf{M}^*(u)F_I(u), \tag{18}$$

where $\tilde{F}_L(u)$ is the reconstructed layers in frequency domain. Then we transform them to spatial domain, and complete the reconstruction. The complexity of this reconstruction step is $O(N \log N)$. Thereby, combing the complexity of estimating model parameters, the whole separation method has a complexity of $O(N \log N)$ in over-determined and determined cases.

An example can be seen in Fig. 1. Images in column 4 are three reconstructed layers with the estimated model parameters, and they are almost the same as the original layers in column 1.

5.2. Under-determined reconstruction

When the mixtures are fewer than the layers, there are an infinite number of images fitting the mixing model. Among them, we want to find the images that most agree with the already extracted layer edges. We define the reconstruction loss function that shall be minimized as:

$$J(\mathbf{L}) = \sum\nolimits_{k,j,x} g(\left|E_j^k(x)\right|)\left|(e_k \cdot L_j)(x) - E_j^k(x)\right|, \tag{19}$$

where $\mathbf{L}$ is a large column vector consisting of all pixels of all layers, E_j^k is the extracted sparse edges of the jth layer when the edge filter e_k is applied (by the method in section 3.4), and 1-norm distance is used as it tends to offer layers with sparse edges [8]. If an estimated edge point has a large magnitude, then it is reliable, and the corresponding disagreement shall be punished much. Thus $g(\cdot)$ is a positive and monotonously increasing function. In our experiments the horizontal and vertical first derivative filters are applied to offer two types of edges, and:

$$g(y) = \begin{cases} 1 & \text{if } y = 0; \\ 4 & \text{otherwise.} \end{cases} \tag{20}$$

Thanks to the linearity of the edge filter, $J(\mathbf{L})$ can be rewritten in a matrix form:

$$J(\mathbf{L}) = \sum\nolimits_j \left|A_j \mathrm{vec}(L_j) - b_j\right|, \tag{21}$$

where $\mathrm{vec}(L_j)$ denotes the vector consisting of all pixels of L_j. Minimizing Eq. (21) must be subject to the constraint

of the mixing model (2). The mixing model (2) is linear with all model parameters known, because the spatial shift is a location to location operation without any pixel value changed, and the operation of mixing coefficients is only to linearly change the magnitude of each pixel. Thus the mixing model can be rewritten as a matrix form:

$$\mathbf{ML} = \mathbf{I}, \tag{22}$$

where $\mathbf{I}$ is a large column vector consisting of all pixels of all mixtures.

By introducing slack variables z_i^- and z_i^+, the problem of minimizing Eq. (21) subject to Eq. (22) becomes:

$$\begin{array}{ll} \text{Min}: & \sum_i c_i^T (z_i^- + z_i^+), \\ \text{s. t.}: & \mathbf{ML} = \mathbf{I}, \\ & A_j \cdot \text{vec}(L_j) - b_j + z_i^- - z_i^+ = \mathbf{0}, \\ & z_i^- \geq \mathbf{0}, z_i^+ \geq \mathbf{0}. \end{array} \tag{23}$$

The above problem can be solved by linear programming, and the global optimal solution can be gotten.

6. Experiments

In this section, we show three sets of experimental results. (1) Four layers separated from two synthetic mixtures (The under-determined case). (2) A transmitted layer and a reflected layer from four real snapshots on glass (The over-determined case). (3) A fade-in layer and a fade-out layer from two real dissolve images (The determined case. Another determined separation has been shown in Fig. 1).

For comparison, we also apply other blind separation algorithms: FastICA [7] and 2D-AC-DC [1]. The settings of the experiments are as follows. (1) As the outputs of a edge filter are usually more independent [6], FastICA, 2D-AC-DC and our method (referred to as SP-BSS) are applied on the identical edge images output by a first derivative filter. (2) 2D-AC-DC can only directly handle two mixtures with two layers, and is only used to separate two dissolve mixtures in the third experiments. In 2D-AC-DC, the initial guesses of $\mathbf{A}$, $\mathbf{S}_h$ and $\mathbf{S}_v$ are the identity, all-zeros and all-zeros matrix, respectively, as in [1]. (3) The real-data are color images. We apply these methods only on the grayscale images of the mixtures to estimate the model parameters, without using the information of colors. Then, R, G and B channels are separately reconstructed.

The under-determined case is simulated in Fig. 3. We artificially mix four layer images into two mixtures, with the parameter matrix $\mathbf{A}$, $\mathbf{S}_h$ and $\mathbf{S}_v$ equal to

$$\begin{bmatrix} 0.25 & 0.25 & 0.25 & 0.25 \\ 0.29 & 0.21 & 0.23 & 0.27 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 & 0 \\ -1 & 4 & 7 & -4 \end{bmatrix} \text{and} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & -2 & 5 & 0 \end{bmatrix}, \tag{24}$$

respectively. Note the mixtures (Row 2 in Fig. 3) are quite complicated, and even human self can not easily distinguish every layer. However, the reconstructed result (Row 3) provides four layers clear enough to show most objects. To the best of our knowledge, the under-determined separation of mixtures with spatial shifts has not yet been addressed by other methods in open literature.

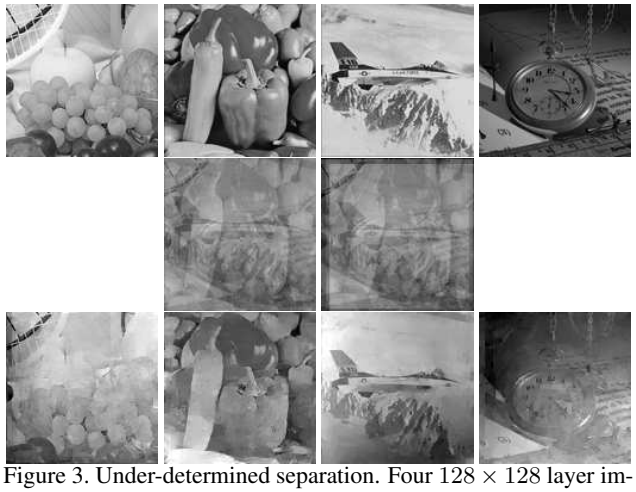


Figure 3. Under-determined separation. Four 128×128 layer images (Row 1). Two mixtures with spatial shifts (Row 2). Four reconstructed layers from two mixtures (Row 3).

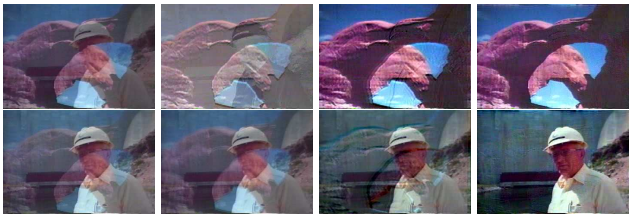
(Four snapshots) (FastICA) (SP-BSS)

Figure 4. Transparency and reflections. Four snapshots (Column 1 and 2). Results of FastICA and SP-BSS (Column 3 and 4).

The second experiment of separating transparency and reflections (the over-determined case) is shown in Fig. 4. Four snapshots are shot towards a window by a outside camera, to record a poster on a rising curtain, but they also record a reflected scene (Column 1 and 2 in Fig. 4). In different snapshots, both the transmitted and the reflected layers have different spatial shifts because of the poster's movements and hand jitter. We apply FastICA and SP-BSS to separate them. (1) FastICA does not achieve successful separation (Column 3), as it can not handle different spatial shifts of layers. (2) SP-BSS gives two clear layer images, where there is almost no superposing effect (Column 4). It is because that SP-BSS can deal with unknown spatial shifts in addition to mixing coefficients.

The third experiment on two real dissolve images (the determined case) is shown in Fig. 5. The images[1] in col-

[1]The images are from 'BOR19.mpg' in 2001 TREC Video Retrieval Test Collection, at http://open-video.org/



(Mixtures) (FastICA) (2D-AC-DC) (SP-BSS)

Figure 5. Separating the dissolve images.

umn 1 are two mixtures, each of which contains a fade-out layer of a rock shot and another fade-in layer of an engineer shot. For the rock shot keeps moving downwards, two layers have relative spatial shifts. Two layers also have varying mixing coefficients by reason of fades. We apply FastICA, 2D-AC-DC and SP-BSS to separate them. (1) FastICA does not achieve successful separation by the above same reason (Column 2). (2) 2D-AC-DC get a better result by considering both different spatial shifts and varying mixing coefficients (Column 3). However, the superposing effect still can be seen, especially in the engineer layer (This is the best result when we try different sizes of truncation windows in 2D-AC-DC). (3) SP-BSS gives the best result: images in column 4 show almost complete separation. Besides, 2D-AC-DC spends 217.4s on estimating the model parameters in two 352×240 mixtures with 22 iterations, while SP-BSS spends 2.6s on the same task (The running environment is matlab 7.4 with Intel Core 2 E6300 1.86GHz and 2GB memory). SP-BSS is much faster, with more reliable results.

7. Conclusion

To recover layers from their shifted superimposed mixtures, we exploit the sparsity of edges in many aspects. First, we extend sparse ICA [2], which uses the scatter plot to estimate the mixing coefficients in the static mixing case, to the shifted mixing case. Second, a dynamic programming formulation is presented to determine the number of layers matched by each spatial shift. With these two methods combined, all the model parameters can be automatically gotten. Then, we use the sparsity to extract edges of each layers. Finally, a linear programming formulation, which also implies utilizing the sparsity, is presented for the reconstruction of original layers in the under-determined case. (In the determined and over-determined cases, the frequency method is used, as in many other methods, *e.g.* [1].)

With these uses of sparsity, we address the problem of blindly separating mixtures of multiple layers with unknown spatial shifts and mixing coefficients. Our method is quite fast in over-determined and determined cases (with the same complexity as FFT), and can reconstruct more layers from fewer mixtures in the under-determined case. The effectiveness of this technology is proved by both simulations and real-data experiments.

Acknowledgment

The work was supported by the National Science Foundation of China (60605002 and 60721003).

References

[1] E. Be'ery and A. Yeredor. Blind separation of reflections with relative spatial shifts. In *ICASSP*, 2006.

[2] A. M. Bronstein, M. M. Bronstein, M. Zibulevsky, and Y. Y. Zeevi. Sparse ica for blind separation of transmitted and reflected images. *IJIST*, 15(1):84–91, 2005.

[3] A. Cichocki and S. Amari. *Adaptive Blind Signal and Image Processing: Learning Algorithms and Applications*. Wiley, New York, 2002.

[4] K. Diamantaras and T. Papadimitriou. Blind separation of reflections using the image mixtures ratio. In *ICIP*, 2005.

[5] H. Farid and E. H. Adelson. Separating reflections and lighting using independent components analysis. In *CVPR*, 1999.

[6] A. Hyvärinen. Independent component analysis for time-dependent stochastic processes. In *ICANN*, 1998.

[7] A. Hyvärinen and E. Oja. Independent component analysis: algorithms and applications. *NN*, 13(4-5):411–430, 2000.

[8] A. Levin and Y. Weiss. User assisted separation of reflections from a single image using a sparsity prior. *TPAMI*, 29(9):1647–1654, 2007.

[9] A. Levin, A. Zomet, and Y. Weiss. Learning to perceive transparency from the statistics of natural scenes. In *NIPS*, 2002.

[10] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In *5-th Berkeley Symposium on Mathematical Statistics and Probability*, 1967.

[11] T. Oo and H. K. Y. O. K. Ikeuchi. The separation of reflected and transparent layers from real-world image sequence. *MVA*, 18:17–24, 2007.

[12] B. Sarel and M. Irani. Separating transparent layers through layer information exchange. In *ECCV*, 2004.

[13] Y. Y. Schechner, N. Kiryati, and R. Basri. Separation of transparent layers using focus. In *ICCV*, 1998.

[14] Y. Y. Schechner, J. Shamir, and N. Kiryati. Polarization-based decorrelation of transparent layers: The inclination angle of an invisible surface. In *ICCV*, 1999.

[15] Y. Y. Schechner, J. Shamir, and N. Kiryati. Blind recovery of transparent and semireflected scenes. In *CVPR*, 2000.

[16] Y. Y. Schechner, J. Shamir, and N. Kiryati. Polarization and statistical analysis of scenes containing a semireflector. *JOSA A*, 17:276–284, 2000.

[17] J. Shi and J. Malik. Normalized cuts and image segmentation. In *CVPR*, 1997.

[18] R. Szeliski, S. Avidan, and P. Anandan. Layer extraction from multiple images containing reflections and transparency. In *CVPR*, 2000.

[19] Y. Tsin, S. B. Kang, and R. Szeliski. Stereo matching with reflections and translucency. In *CVPR*, 2003.