14th European Signal Processing Conference (EUSIPCO 2006), Florence, Italy, September 4-8, 2006, copyright by EURASIP

RECONSTRUCTION OF HIDDEN IMAGES USING WAVELET TRANSFORM AND AN ENTROPY-MAXIMIZATION ALGORITHM

Naoto Nakamura, Shigeru Takano, Yoshihiro Okada, and Koichi Niijima

Department of Informatics, Kyushu University
6-1, Kasuga-koen, Kasuga, Fukuoka, 816-8580, JAPAN
phone: +81-92-583-7633, fax: +81-92-583-7635, email: {n-naka,takano,okada,niijima}@i.kyushu-u.ac.jp
web: http://www.i.kyushu-u.ac.jp/

ABSTRACT

This paper proposes a blind image separation method using wavelet transform and an entropy-maximization algorithm. Our blind separation algorithm is an improved version of the entropy-maximization algorithms presented by Bell-Sejnowsky and Amari. These algorithms work well for signals having a supergaussian distribution, such as speech and audio. The proposed method is to apply the improved algorithm to the wavelet coefficients of a natural image, whose distribution is close to supergaussian. Our method successfully reconstruct twelve images hidden in another twelve images which are similar each other.

1. INTRODUCTION

Blind image separation has attracted much attention in image processing. Many algorithms for blind source separation have been developed based on independent component analysis (ICA), and applied to signal processing such as audio source separation [3], separation of natural images [5], and information hiding [4]. Bell-Sejnowsky's method, which is one of ICA techniques, is an information-maximization approach to blind separation. This method does not require input data distributions, and is based on maximizing the entropy of neural network outputs. Actually, the method is realized as an algorithm for changing the connection weights of the neural network. Bell-Sejnowsky's algorithm is effective in blind separation of supergaussian data. However, their method does not work well for images, because the histogram of pixels has an irregular distribution.

In this paper, we first improve the entropy-maximization algorithms for training a neural network, presented by Bell-Sejnowsky [2] and Amari [1], since their algorithms have slow convergence for many signals. Next, we compute high-pass components by applying wavelet transform to mixing images, which have a supergaussian distribution. A neural network is trained by applying our improved algorithm to the computed wavelet coefficients. The reconstruction of hidden images is performed by computing linear combinations of the mixing images and the learned weights of the neural network.

In experiments, we make twelve mixing images by hiding eleven natural images in one natural image. Since the mixing rate is very low, the produced images look like the same. We try to reconstruct the hidden source images from the mixing images exploiting our method.

The outline of the paper is as follows. In Section 2, we introduce biorthogonal wavelet transform. Section 3 describes the entropy-maximization method. In Section 4, we present our blind separation method. Section 5 involves experimental results. We close in Section 6 with concluding remarks and plans for future work.

2. WAVELET TRANSFORM OF IMAGES

Let h_l and g_l be low-pass and high-pass analysis filters of biorthogonal wavelet, respectively. We denote an image by $C_{0,i,j}$. Then, wavelet decomposition formula on $C_{0,i,j}$ can be written as

$$C_{p,i,j} = \sum_{k,l} h_k h_l C_{p-1,2i+k,2j+l},$$

$$D_{p,i,j} = \sum_{k,l} h_k g_l C_{p-1,2i+k,2j+l}, \quad (1)$$

$$E_{p,i,j} = \sum_{k,l} g_k h_l C_{p-1,2i+k,2j+l}, \quad (2)$$

$$F_{p,i,j} = \sum_{k,l} g_k g_l C_{p-1,2i+k,2j+l}, \quad (3)$$

where p moves from 1 to L. The $C_{p,i,j}$ indicate low-pass coefficients of $C_{0,i,j}$ in p-resolution level. The $D_{p,i,j}$, $E_{p,i,j}$ and $F_{p,i,j}$ represent high-pass coefficients of $C_{0,i,j}$ in p-resolution level, in horizontal, vertical, and diagonal directions, respectively. It is known experimentally that the low-pass coefficients for a natural image have an irregular distribution, but the histogram of its high-pass coefficients behaves like a supergaussian distribution. In Fig.1, we illustrate high-pass coefficients obtained by wavelet decomposition of an example image. Figure 2 shows their histograms. In this paper, we use such a property of wavelet coefficients.

(a) (b)

Figure 1: (a) Original image, (b) wavelet decomposition.

3. ENTROPY-MAXIMIZATION METHOD

We employ basically the entropy-maximization method proposed by Bell and Sejnowski for blind separation [2].

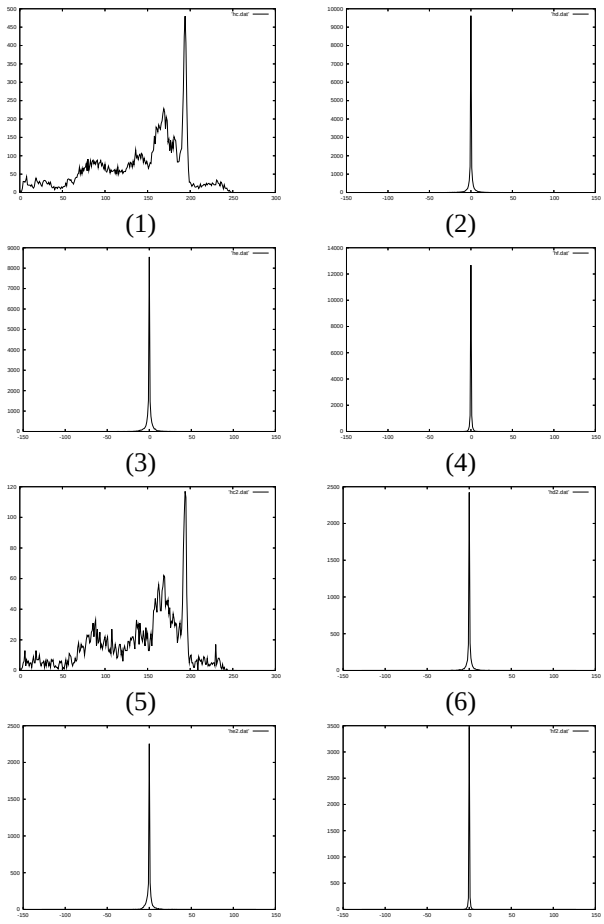

Figure 2: Histograms of (1) $C_{1,i,j}$, (2) $D_{1,i,j}$, (3) $E_{1,i,j}$, (4) $F_{1,i,j}$, (5) $C_{2,i,j}$, (6) $D_{2,i,j}$, (7) $E_{2,i,j}$ and (8) $F_{2,i,j}$.

3.1 Bell-Sejnowski's algorithm

Let s_ν, $\nu = 1,\ldots,N$, be unknown signal sources, and define a vector

$$\mathbf{s} = (s_1,\ldots,s_N)^T,$$

where 'T' denotes a transpose. We assume that s_ν are statistically independent, and that the mixing signals x_ν, $\nu = 1,\ldots,N$, can be written by a linear combination of s_ν, i.e.,

$$x_\nu = \sum_{\mu=1}^{N} a_{\nu\mu} s_\mu, \qquad \nu = 1,\ldots,N. \tag{4}$$

A vector form of (4) is

$$\mathbf{x} = A\mathbf{s}.$$

Here, $\mathbf{x} = (x_1,\ldots,x_N)^T$ and $A = (a_{\nu\mu})$ is an $N \times N$ matrix. ICA is to obtain the inverse matrix A^{-1} from the vector $\mathbf{x}$. To solve this problem, Bell and Sejnowski exploited a neural network with N input and N output nodes,

$$y_\nu = g(W_\nu \cdot \mathbf{x} - \theta_\nu), \qquad \nu = 1,\ldots,N. \tag{5}$$

Here $W_\nu = (w_{\nu 1},\ldots,w_{\nu N})^T$ denote weight vectors, θ_ν biases, $g(u)$ a sigmoid function, and '$\cdot$' inner product. This neural network is shown in Fig. 3.

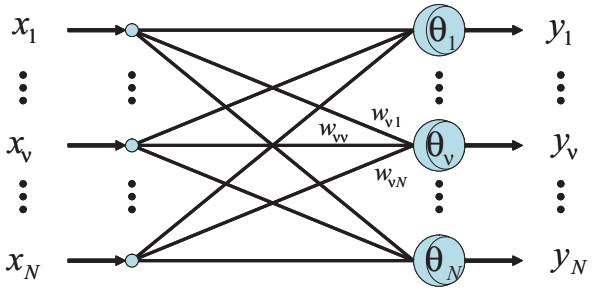

Figure 3: Neural network.

The entropy-maximization method is to determine the weight matrix W and the bias vector θ so as to maximize the entropy

$$H(\mathbf{y}) = < \ln|J| > . \tag{6}$$

Here '$<>$' denotes an expected value and J represents the determinant

$$J = \det \begin{pmatrix} \frac{\partial y_1}{\partial x_1} & \frac{\partial y_2}{\partial x_2} & \cdots & \frac{\partial y_1}{\partial x_N} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial y_N}{\partial x_1} & \frac{\partial y_N}{\partial x_2} & \cdots & \frac{\partial y_N}{\partial x_N} \end{pmatrix}.$$

The problem (6) is equivalent to minimizing a minus of $H(\mathbf{y})$. For the sigmoid function

$$g(u) = \frac{1-e^{-u}}{1+e^{-u}},$$

Bell-Sejnowski's algorithm for changing W and θ is written as

$$W^{(t+1)} = W^{(t)} + \gamma(((W^{(t)})^T)^{-1} - 2\mathbf{y}(t)\mathbf{x}(t)^T)W^{(t)}, \tag{7}$$

$$\theta^{(t+1)} = \theta^{(t)} + 2\gamma\mathbf{y}(t). \tag{8}$$

Here γ is a sufficiently small positive number. In this algorithm, the inverse matrix $((W^{(t)})^T)^{-1}$ has to be calculated per iteration.

3.2 Amari's natural gradient method [1]

From the viewpoint of information geometry, Amari derived a natural gradient algorithm

$$W^{(t+1)} = W^{(t)} + \gamma(E - 2\mathbf{y}(t)(\mathbf{u}(t) + \theta^{(t)}))W^{(t)} \tag{9}$$

instead of (7), where E denotes an $N \times N$ unit matrix and

$$\mathbf{u}(t) = W^{(t)}\mathbf{x}(t) - \theta^{(t)}. \tag{10}$$

This algorithm does not need to calculate the inverse matrix $((W^{(t)})^T)^{-1}$. However, it does not produce good convergence results for a lot of mixing images, even if we choose initial values of the matrix W and the bias vector θ carefully.

3.3 Improved version

We resolve the difficulty above by adding a penalty term to (9). From (9), the condition for convergence is

$$E = 2\mathbf{y}(t)(\mathbf{u}(t) + \theta^{(t)})^T.$$

This suggests that at least off-diagonal elements of the right hand side should be zero. Therefore, we add the penalty term F, which is given by (12), to get

$$W^{(t+1)} = W^{(t)} + \gamma((E - 2\mathbf{y}(t)(\mathbf{u}(t) + \theta^{(t)})^T)W^{(t)} - CF). \quad (11)$$

Here C is a penalty constant and F is

$$F = \begin{pmatrix} 0 & y_1(u_2+\theta_2) & \cdots & y_1(u_N+\theta_N) \\ y_2(u_1+\theta_1) & 0 & \cdots & y_2(u_N+\theta_N) \\ \vdots & \vdots & \cdots & \vdots \\ y_N(u_1+\theta_1) & y_N(u_2+\theta_2) & \cdots & 0 \end{pmatrix}. \quad (12)$$

4. OUR BLIND SEPARATION METHOD

Bell-Sejnowski's algorithm works well in the case that the probability density function of mixing signals is closed to supergaussian such as speech and audio. Since that of an image is irregular as shown in Fig.2(1), it is not suitable to apply Bell-Sejnowski's type of ICA to images themselves. Fortunately, the histogram of wavelet coefficients for a natural image behaves like a supergaussian distribution. So, we apply our improved algorithm in Section 3.3 to them for blind separation.

Our blind separation algorithm involves the following steps:

1. Prepare N mixing images $C^{\nu}_{0,i,j}$, $\nu = 1, \cdots, N$.
2. Compute the high-pass components $D^{\nu}_{p,i,j}$, $E^{\nu}_{p,i,j}$ and $F^{\nu}_{p,i,j}$, defined by (1), (2) and (3), respectively, by using the biorthogonal wavelet filters.
3. Construct N high-pass images whose ν-th image consists of $D^{\nu}_{p,i,j}$, $E^{\nu}_{p,i,j}$ and $F^{\nu}_{p,i,j}$.
4. Scan the high-pass images both top and bottom, left and right, and construct the input data $x_1(t)$, $\cdots$, $x_N(t)$ of the neural network.
5. Inputting $x_1(t)$, $\cdots$, $x_N(t)$ into our blind separation algorithm successively, learn the weight matrix W and the bias vector θ.
6. Compute
$$\mathbf{z}(t) = W\mathbf{c}(t) - \theta \quad (13)$$
using the learned W and θ, where $\mathbf{c}(t)$ is a vector whose components represent the pixels of the mixing images at the location t.

Figure 4 illustrates an overview of our algorithm.

5. EXPERIMENTAL RESULTS

In simulation, we use two groups of 12 mixing images, as shown in Figs.5 and 7, respectively. Each image has the size of 256×256 , and 8 bits expression. The training images in each group are generated by mixing 12 source images different from each other, using the mixing rates shown in Table 1.

For wavelet transform, we employ the B-spline wavelet filers, which are listed in Table 2. The wavelet decomposition was executed until 8 resolution levels.

The input vectors $(x_1(t), \cdots, x_{12}(t))^T$ of the neural network (5) were constructed from the high-pass images produced by wavelet transform. We determine the weight matrix

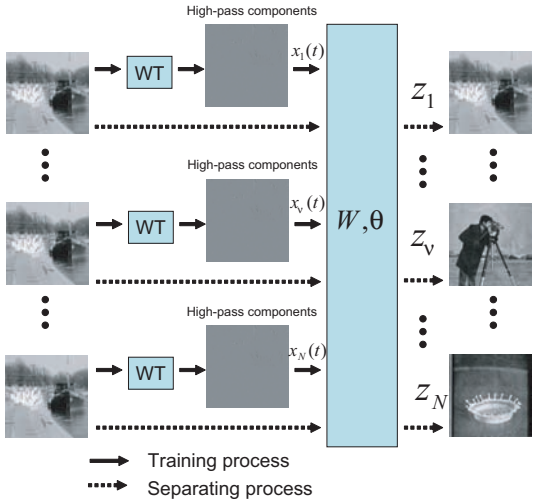

Figure 4: Overview of our algorithm.

Table 1: Mixing rates.

A	1	2	3	4	5	6	7	8	9	10	11	12
1	0.89	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
2	0.78	0.06	0.03	0.03	0.02	0.02	0.01	0.01	0.01	0.01	0.01	0.01
3	0.78	0.01	0.06	0.03	0.03	0.02	0.02	0.01	0.01	0.01	0.01	0.01
4	0.78	0.01	0.01	0.06	0.03	0.03	0.02	0.02	0.01	0.01	0.01	0.01
5	0.78	0.01	0.01	0.01	0.06	0.03	0.03	0.02	0.02	0.01	0.01	0.01
6	0.78	0.01	0.01	0.01	0.01	0.06	0.03	0.03	0.02	0.02	0.01	0.01
7	0.78	0.01	0.01	0.01	0.01	0.01	0.06	0.03	0.03	0.02	0.02	0.01
8	0.78	0.01	0.01	0.01	0.01	0.01	0.01	0.06	0.03	0.03	0.02	0.02
9	0.78	0.02	0.01	0.01	0.01	0.01	0.01	0.01	0.06	0.03	0.03	0.02
10	0.78	0.02	0.02	0.01	0.01	0.01	0.01	0.01	0.01	0.06	0.03	0.03
11	0.78	0.03	0.02	0.02	0.01	0.01	0.01	0.01	0.01	0.01	0.06	0.03
12	0.78	0.03	0.03	0.02	0.02	0.01	0.01	0.01	0.01	0.01	0.01	0.06

W and the bias vector θ in the neural network by using the iterations (11) and (8). The iteration was started from $W = E$, where E is the $N \times N$ unit matrix, and $\theta = \mathbf{0}$, and the step size γ and the penalty constant C were chosen as $\gamma = 10^{-8}$ and $C = 1000$.

As a result of the iteration, we obtained the trained weight matrix and the bias vector as shown in Tables 3 and 4. To extract the hidden images, we computed $z_\nu, \nu = 1, \cdots, 12$ by using (13). The separated images are illustrated in Fig. 6 for group 1, and Fig. 8 for group 2.

We denote a raster-scanned signal of the ν-th source image by $s_\nu(t), 0 \le t \le M-1$. The mean square error (MSE) between $s_\nu(t)$ and the raster-scanned signal $z_\nu(t)$ is written

Table 2: B-spline wavelet filters

l	h_l	g_l
-1	0.5000	0.0833
0	1.0000	-0.5000
1	0.5000	0.8333
2	0.0000	-0.5000
3	0.0000	0.0833

Table 3: Trained weights and biases for group 1.

W	1	2	3	4	5	6
1	0.9751	-0.1947	-0.2406	0.0746	-0.1423	0.0110
2	-1.0613	3.0504	-0.9267	-0.7035	-0.0052	-0.1344
3	-1.1149	-0.0377	3.2171	-0.9985	-0.8515	-0.0178
4	-1.2535	-0.0305	-0.0260	3.2560	-1.2099	-0.5018
5	-1.2365	-0.0619	-0.0311	0.0255	3.0748	-0.9364
6	-1.0845	-0.1886	-0.1091	-0.0864	-0.0129	3.3473
7	-0.9320	0.0084	-0.0349	-0.0905	0.0262	-0.1033
8	-1.0692	0.2355	0.0011	-0.1408	-0.0364	0.0391
9	-1.1011	-0.2067	0.1700	-0.0410	-0.0682	-0.1206
10	-1.1372	-0.0141	-0.1669	0.1692	0.0500	-0.0930
11	-1.1993	-0.9257	-0.0853	-0.1371	0.3189	-0.0580
12	-0.9332	-1.1029	-0.7077	-0.1033	-0.1700	0.0628
θ	0.0067	0.0016	-0.0003	-0.0008	-0.0032	0.0000

W	7	8	9	10	11	12
1	0.0498	0.0723	-0.1944	0.0411	0.1428	0.2677
2	0.2727	0.0047	-0.0030	-0.0843	-0.1255	-0.1285
3	-0.1275	0.2938	0.0396	-0.1002	-0.0553	-0.0882
4	-0.0343	-0.1958	0.2723	-0.0055	-0.1097	0.0361
5	-0.7849	0.1419	-0.2077	0.1998	-0.0916	0.0741
6	-1.1133	-0.7884	0.0169	-0.1388	0.1569	0.1544
7	3.0861	-1.1610	-0.6883	0.0715	-0.2194	0.1810
8	0.0262	3.1391	-1.1199	-0.7166	0.0612	-0.2006
9	0.0538	0.0788	3.1833	-1.0908	-0.7013	-0.0103
10	-0.0708	-0.0347	0.0243	3.3006	-1.1423	-0.7259
11	-0.0815	-0.1196	-0.0017	0.0035	3.2857	-0.8397
12	0.0891	-0.1233	-0.0480	-0.0343	-0.1390	3.3694
θ	-0.0003	0.0009	-0.0013	-0.0001	-0.0011	0.0008

Table 4: Trained weights and biases for group 2.

W	1	2	3	4	5	6
1	1.1569	-0.0595	-0.0555	-0.0550	-0.0743	-0.0686
2	-0.8925	2.5837	-1.0023	-0.6443	0.1208	-0.0996
3	-0.7818	-0.0091	2.5997	-1.0471	-0.6349	0.0915
4	-0.4796	0.0364	-0.0284	1.8321	-0.7381	-0.5494
5	-0.8520	-0.0374	0.0451	-0.0140	2.3726	-0.9337
6	-1.0706	-0.1403	-0.0506	-0.0153	0.0224	2.8991
7	-1.1659	-0.0140	-0.1266	-0.0312	-0.0332	-0.0769
8	-1.2078	0.3119	0.0294	-0.1175	-0.0533	-0.0519
9	-1.1924	-0.1545	0.2860	0.1007	-0.1608	-0.1063
10	-1.1639	0.1125	-0.1604	0.3870	-0.0109	-0.1032
11	-0.9866	-0.6431	0.0879	-0.1338	0.3116	-0.0383
12	-0.9061	-1.0283	-0.6342	0.1184	-0.1423	0.0063
θ	-0.0119	-0.0021	-0.0027	0.0023	-0.0043	0.0038

W	7	8	9	10	11	12
1	0.1919	-0.0883	0.0134	0.0590	-0.1993	-0.0333
2	0.2432	0.0352	-0.1114	-0.0694	-0.0207	-0.0042
3	-0.2137	0.2101	-0.0092	-0.1228	0.0684	-0.0292
4	0.0648	-0.0435	0.0343	0.0685	-0.0714	-0.0200
5	-0.5139	0.1268	-0.1493	0.1105	0.0213	-0.0463
6	-0.8531	-0.7288	-0.0163	-0.2064	0.3439	0.0197
7	3.2471	-1.1209	-0.6670	-0.1065	-0.1363	0.3759
8	-0.0214	3.1556	-0.9992	-0.5915	-0.0736	-0.1820
9	-0.1737	-0.0523	3.2767	-0.6437	-0.9727	-0.0253
10	0.0163	-0.0817	-0.1208	3.3816	-1.2840	-0.8037
11	-0.1089	0.1702	-0.0160	-0.1547	2.6884	-1.0399
12	-0.0451	-0.1270	-0.0367	-0.01402	0.03772	2.6169
θ	0.0035	-0.0037	0.0037	-0.0050	-0.0011	-0.0025

as

$$MSE = \frac{1}{M}\sum_{t=0}^{M-1}(z_v(t) - s_v(t))^2.$$

The image quality is often measured by using the MSE-based evaluation standard

$$PSNR = 10\log_{10}\frac{255\times 255}{MSE}.$$

The value of PSNR is shown in the bottom of each reconstructed image in Figs. 6 and 8. As seen from these results, the hidden images are almost perfectly reconstructed.

Simulation was performed on the laptop computer with Pentium M 1.2 GHz and 768MB SDRAM. Training time was about 290 sec..

6. CONCLUSION

We proposed a blind separation method in order to reconstruct hidden images. Our method combines biorthogonal wavelet transform with an improved version of Bell-Sejnowski's blind separation algorithm. We computed the high-pass components of the mixed images exploiting wavelet transform. Using the obtained wavelet coefficients, we trained a neural network based on our blind separation algorithm to find source images. The learned weight matrix and bias vector were utilized to recover hidden images.

In simulation, we attempted to reconstruct hidden images from two groups, each of which includes 12 mixed images. In both groups, we could reconstruct the hidden images almost perfectly. It is future work to apply our algorithm to other problems such as the separating reflections and blind separation from a single image.

REFERENCES

[1] S.-I. Amari, A. Cichocki and H.H. Yang, "A new learning algorithm for blind signal separation," *Advances in Neural Information Processing Systems*, vol.8, pp. 757–763, 1996.

[2] A. J. Bell and T. J. Sejnowski, "An information-maximization approach to blind separation and blind deconvolution," *Neural Computation*, vol. 7, pp. 1129–1159, 1995.

[3] F. Bimbot and L. Benaroya, "Audio source separation with a single sensor," *IEEE Trans. Audio, Speech and Language Processing*, vol. 14, no. 1, pp 191–199, 2006.

[4] S. Bounkong, B. Toch, D. Saad and D. Lowe, "ICA for watermarking digital images," *Journal of Machine Learning Research, 4*, pp. 1471–149, 2003.

[5] A. Cichocki and W. Kasprzak, "Nonlinear learning algorithms for blind separation of natural images," *Neural Network World*, vol.4, no.4, pp. 515–523, 1996.

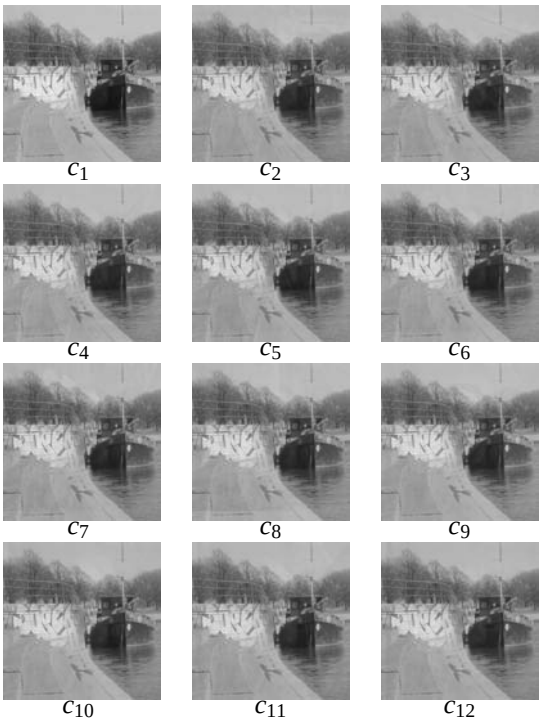

Figure 5: 12 mixing images in group 1.

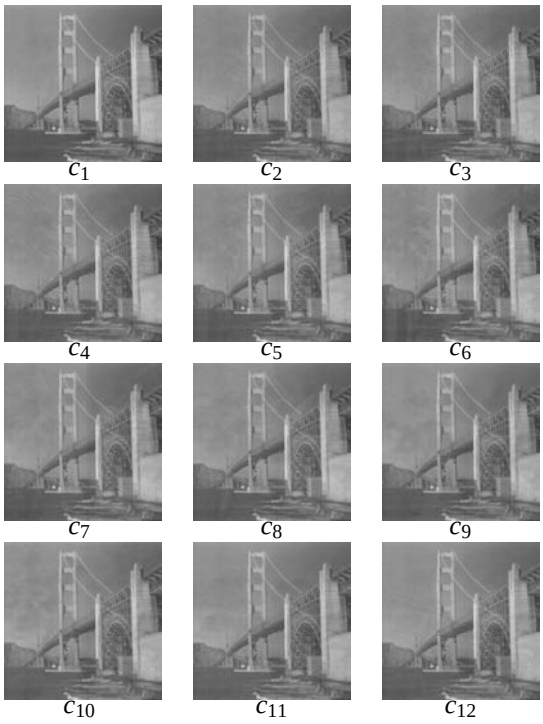

Figure 7: 12 mixing images in group 2.

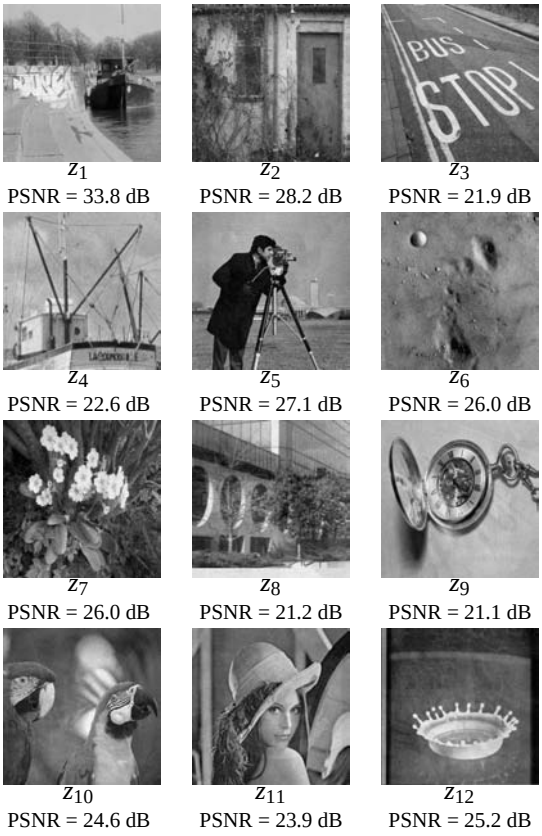

Figure 6: Images separated from the 12 mixing images in group 1.

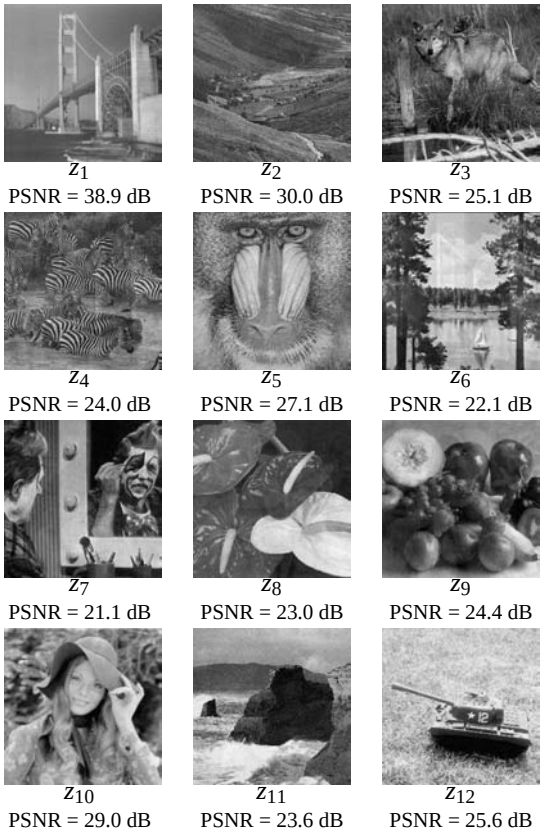

Figure 8: Images separated from the 12 mixing images in group 2.