Removing the Blocking Effects Using Artificial Neural Network

Chin-Chen Chang, Chi-Shiang Chan and Chun-Sen Tseng

Department of Computer Science and Information Engineering
National Chung Cheng University
Chiayi, Taiwan 621, R.O.C.
E-mail: {ccc, cch, tcs92}@cs.ccu.edu.tw
FAX: 886-5-2720859

Correspondence address:
Chin-Chen Chang
Department of Computer Science and Information Engineering
National Chung Cheng University
Chiayi, Taiwan 621, R.O.C.
Email: ccc@cs.ccu.edu.tw
FAX: 886-5-2720859
Removing the Blocking Effects Using Artificial Neural Network

Abstract

In 2003, Ying and Rabab proposed their method to eliminate blocking effects by using the so-called zero-masking method with the information of neighboring blocks. Ying and Rabab believed that the relationship among the information of neighboring blocks is linear. Therefore, the way to eliminate blocking effects is designed by using the information of neighboring blocks in a linear way. To make a difference, we use an artificial neural network to build up the approximate relationship. After modifying the AC coefficients according to the approximate relationship and reversing the DCT frequency domain to the pixel domain, the blocking effects can be successfully eliminated. Experimental results show that our method can get better image quality than Ying and Rabab’s method.

Key words: Blocking effects, blocking artifacts, DCT, artificial neural network
1. Introduction

Due to the astonishing growth of popularity of computers and computer networks, traditional pictures and ways of using and processing pictures have nowadays been digitized. However, it is storage consuming to save electronic pictures (or more frequently called digital images) in the form of raw data. In order to lighten the heavy burden on the storage, many image compression methods have been proposed and applied, JPEG being one of the most celebrated methods [11]. The kernel of JPEG image compression method is discrete cosine transformation (DCT). DCT image compression methods are lossy techniques that are designed to be used on digital images and video data. The concept of DCT compacts the pixel values in a block so that the storage for the electronic pictures can be reduced. Although JPEG image compression methods are useful, they can cause blocking effects. The reason is that JPEG divides a raw picture into non-overlapping blocks before it does DCT. When the DCT frequency domain is reversed back to pixel domain, the block effects show up in the form of visible block boundaries.

To get rid of blocking effects, many researchers have been trying to create and offer their revised DCT-based methods. Some of them try to retain some additional information to alleviate the block effects at the encoding end [5, 9, 15,]. However, in order to save the information, the storage of the encoded image becomes large.
Moreover, those methods must modify the existing image compression standard so that the image compression procedure can handle the additional information. On the contrary, other methods use post-processing methods to eliminate block effects. These methods are more intuitive. To eliminate the blocking effects, only the compressed image is required; there is no need for the additional information to exist here. In this case, the existing image compression standard does not need to be modified.

The post-processing techniques used to eliminate blocking effects can be roughly classified into two categories: spatial domain methods [4, 6, 7] and frequency domain methods [10, 12, 13]. Spatial domain methods function in the form of image filtering. In other words, these methods use low-pass filtering to filter out the block effects from the decompressed images. Gaussian low-pass filtering is the most effective filtering method to eliminate blocking effects. However, although it can eliminate blocking effects, as a side effect, the detail-containing areas are also eliminated. Consequently, although no awkward blocks are present, the processed images look blurred.

On the other hand, in frequency domain methods, the DC and AC coefficients are directly modified. After converting those DC and AC coefficients into the pixel domain, the blocking effects are eliminated. The most famous frequency domain
method is Zeng’s zero-masking method [13]. It first checks whether the visible boundary has arisen between two adjacent blocks in the decompressed image. Then, it draws out the block that contains visible boundary in it. After that, the zero-masking method is applied to modify DCT coefficients so as to reduce blocking effects in this block. However, this method directly modifies DCT coefficients without referring to the information from its neighborhood, which may cause artifacts.

In 2003, Ying and Rabab [12] proposed their method to eliminate blocking effects by using zero-masking with the help of the information of its neighboring blocks. In their method, it combines the right half of the block on the left and the left half of the block on the right to form a covering block. The physical positions between two horizontally adjacent blocks and their corresponding covering block are illustrated in Fig. 1.

![Fig. 1. Horizontally adjacent blocks](image)

In Fig. 1, block A and block B are two non-overlapping neighboring blocks
coming from the original DCT blocks. Block C is the covering block, which comes from combining the right half of block A and left half of block B. After getting block C, it checks and makes sure what kind of block this block C is (smooth, strong texture, or edge, for example). Then, it uses the correlation between the neighboring blocks (block A and block B) to reduce the discontinuity of the pixels across the boundaries. That is, it modifies the DC and some AC coefficients of block C according to the DC value and some AC coefficients of block A and block B. Ying and Rabab believed that the relationship of AC coefficients among blocks A, B and C is linear relationship. Therefore, they have designed their scheme linearly.

In fact, the relationship of AC coefficients among blocks A, B and C is very complex. In order to simulate and take advantage of the relationship, in this paper, we shall offer a new method with an artificial neural network. As we know, artificial neural networks have been used in a wide variety of fields to solve complex problems such as image coding [2, 14]. The artificial neural network can gain knowledge through learning, and the knowledge is stored within inter-neuron connections known as synaptic weights. Through the learning process, we can build up the approximate relationship and keep it in those synaptic weights. Finally, with the synaptic weights, our method can eliminate blocking effects successfully. According to our experimental results, the new method can indeed get good image quality.
The rest of this paper is organized as follows. To begin with, we shall review some related works in Section 2, including a brief introduction to the method proposed by Ying and Rabab [12]. Then, we will continue to present our method in Section 3, where the artificial neural network design is described first, followed by the way to apply the artificial neural network to eliminate blocking effects. In Section 4, we shall offer our experimental results to demonstrate the effectiveness of our new method. Finally, the conclusions will be given in Section 5.

2. Related Works

First, let’s have a look at Ying and Rabab’s method [12]. This method is a DCT-based method for reducing blocking effects. The method uses the correlation information between neighboring blocks to avoid blocking effects. That is, the method finds out the relationship of the DCT coefficient values between two neighboring blocks. According to the relationship, the continuing pixels across the boundary can be recovered. To be more precise, we describe the details of Ying and Rabab’s method [12] in the following paragraphs.

Following the DCT transforming standard, we set the block size to 8x8 pixels. The decompressed image is divided into non-overlapping blocks. For each pair of neighboring blocks, we combine the right half of the left block and the left half of the
right block to form a covering block with $8 \times 8$ pixels. The physical positions between two horizontal adjacent blocks and their corresponding covering block are illustrated in Fig. 1. In Fig. 1, block A and block B are two non-overlapping neighboring DCT blocks, and block C is the covering block, which is the combination of the right half of block A and left half of block B.

Because block A and block B go through DCT transforming, the block effects occur in the boundary between blocks A and block B after the DCT-encoded blocks A and B get decompressed. When we draw out block C, the boundary pixels are drawn out at the same time. That is, the blocking effects exist in block C. In order to eliminate the blocking effects in block C, Ying and Rabab’s method tries to modify the DCT coefficients of block C.

Before going further, we define some terms for future use. For each block $X$, after going through DCT, the pixels in block $X$ are transformed into DCT coefficients. Let $F_X(u, v)$ represent the DCT coefficient at position $(u, v)$ of this $8 \times 8$ block $X$, where $u$ and $v$ are in the range from 0 to 7. The DCT coefficients are shown in Fig 2. For example, $F_X(0, 0)$ is the top-left coefficient of the encoded block $X$, and it is also called a DC coefficient. The rest of the coefficients are called AC coefficients.
In [12], it is revealed that the vertical boundary line in block C is related to some coefficients in the first row of the DCT coefficients. To be more precise, when the vertical boundary line exists, the values of $F_C(u, v)$ become nonzero values, where $u = 0$ and $v = 0, 1, 3, 5, 7$. No matter whether the vertical boundary line is a true line or a line caused by blocking effects, the line does exist. According to description above, the blocking effects can be eliminated by modifying these coefficients.

Since Ying and Rabab supposed that vertical boundary line has to do with some certain coefficients, they gave three constraints to judge whether the vertical boundary line is a true line or a line caused by blocking. For each two horizontally adjacent blocks, the left block is treated as block A and the right block as block B. The three
constraints, which are used to judge whether the vertical boundary line is a true line or
a line caused by blocking, is shown below [12].

\[ | F_A(0,0) - F_B(0,0) | < T_1, \]
\[ | F_A(0,1) - F_B(0,1) | < T_2, \]
\[ | F_C(3,3) | < T_3, \]

where the thresholds T_1, T_2, and T_3 are all predetermined, fixed values. In
constraint (1), if the absolute value of difference between \( F_A(0,0) \) and \( F_B(0,0) \) is less
than threshold T_1, it means that the mean pixel value of block A is similar to that of
block B. If blocks A and B satisfy constraints (1) and (2), it means blocks A and B
have similar horizontal frequency property. As for constraint (3), it indicates that the
boundary between blocks A and B belongs to a relatively smooth region.

In the case where blocks A and B satisfy the three constraints above, the vertical
boundary line between them is judged as a block effect. The next step here is to try
to eliminate this block effect by modifying the coefficients in block C. Ying and
Rabab used some formulas below [12] to modify the DCT coefficients in block C to
achieve the goal.

\[ F_C(0, v) = \alpha_0 \times F_C(0, v) + \beta_0 \times [ F_A(0, v) + F_B(0, v) ], \]

where \( v = 0, 1 \).

\[ F_C(0, v) = \alpha_1 \times F_C(0, v) + \beta_1 \times [ F_A(0, v) + F_B(0, v) ], \]
where \( v = 3, 5, 7 \). \hspace{1cm} (5)

\[
F_C(u, v) = F_C(u, v),
\]

for \( v \neq 0, 1, 3, 5, 7 \). \hspace{1cm} (6)

Ying and Rabab’s method sets \( \alpha_0 = 0.6, \beta_0 = 0.2, \alpha_1 = 0.5 \) and \( \beta_1 = 0.25 \). Those values come from experimental results. After calculating the new coefficients and storing them back, they obtain the modified block \( C' \) by using inverse DCT.

Ying and Rabab claimed that the block effect in block \( C \) can be eliminated after going through the procedure above. That is, there will not be any vertical boundary line in the modified block \( C' \).

The same thing applies to vertically adjacent blocks, too. Horizontal boundary lines may exist between vertically adjacent blocks. The top block of vertically adjacent blocks is denoted as block \( A \), and the bottom block as block \( B \). Block \( C \) comes from combining the bottom half of block \( A \) and the top half of block \( B \). The new three constraints for the horizontal boundary line are as follows.

\[
| F_A(0,0) - F_B(0,0) | < T_1, \hspace{1cm} (7)
\]

\[
| F_A(1,0) - F_B(1,0) | < T_2, \hspace{1cm} (8)
\]

\[
| F_C(3,3) | < T_3. \hspace{1cm} (9)
\]

The formulas to eliminate the horizontal boundary line by modifying the DCT coefficients are shown below.
\[ F_C(u, 0) = \alpha_0 \times F_C(u, 0) + \beta_0 \times [F_A(u, 0) + F_B(u, 0)], \]

where \( u = 0, 1 \). \hspace{1cm} (10)

\[ F_C(u, 0) = \alpha_1 \times F_C(u, 0) + \beta_1 \times [F_A(u, 0) + F_B(u, 0)], \]

where \( u = 3, 5, 7 \). \hspace{1cm} (11)

\[ F_C(u, v) = F_C(u, v), \]

for all \( u \neq 0, 1, 3, 5, 7 \). \hspace{1cm} (12)

After calculating the new coefficients and storing them back, they get the modified block \( C' \) by using inverse DCT. This way, the horizontal boundary line in block \( C \) can be eliminated.

3. The Proposed Method

In this section, we shall present our method. First of all, the concept of artificial neural network is introduced. Then, we continue to apply the artificial neural network to eliminate blocking effects.

The artificial neural network is designed to solve complex problems of all kinds. If the relationship between inputs and the desired outputs is not linear, an artificial neural network can help build up an approximate relationship between them. This way, once new inputs are fed into the artificial neural network, the outputs can be roughly predicted.

Many models have been created and used in the artificial neural network. The
multilayer perceptron (MLP) is among the most widely accepted ones. An MLP model adds one or more hidden layers between inputs and outputs to extract high-order statistics. The connection lines between neurons in adjacent layers are full connection. Each connection line has its weight so as to represent the relative strength between two neurons. After building up the MLP model, error back-propagation algorithm [1, 8] is applied on the model to achieve our goal.

As we have just described above, the MLP model adds one or more hidden layers between inputs and outputs. The physical MPL structure is shown in Fig. 3. The inputs are fed into the neurons in the input layer. The first hidden layer calculates values of neurons by multiplying the inputs in the input layer by the weights. For example, the value of $N_4$ in Fig. 3 comes from $N_1 \times W_{14} + N_2 \times W_{24} + N_3 \times W_{34}$. As for the second hidden layer, it takes the values of the neurons in first hidden layer as the inputs. Then, the neuron values in the second hidden layer are calculated by multiplying the values of the neurons in the first hidden layer by the weights. For other hidden layers, the same procedure is used to calculate neuron values from its front layer and pass the values to the layer behind. For the value of first output node, $N_8$, it comes from $N_4 \times W_{48} + N_5 \times W_{58} + N_6 \times W_{68} + N_7 \times W_{78}$. The values of neurons in the output layer are the final results. These final results are the output values.
The way to calculate the weights is through the back-propagation algorithm [1, 8]. There are two phases in the back-propagation algorithm. One is the forward phase, and the other is the backward phase. In the forward phase, the input values are multiplied by the weights to get the values and pass them layer by layer. The values of neurons in the output layer are the final outputs. In the backward phase, the errors between the final outputs and the desired outputs are used to adjust the weights so as to make the final outputs closer to the desired outputs. The basic way to adjust weights is to use the gradient steepest descent method to minimize the error.

In this paper, an MLP model with a back-propagation algorithm is performed to eliminate the blocking effects in the post-processing image. As we mentioned in Section 2, the vertical boundary line occurs between two horizontally adjacent blocks. Therefore, the DCT coefficients in block C must be modified in order to eliminate the
vertical boundary line. Ying and Rabab used linear combination among DCT coefficients in blocks A, B and C to decide the quantity of the modified DCT coefficients in block C. However, the relationships among DCT coefficients in blocks A, B and C are very complex. It directly influences the ability of eliminating blocking effects.

Because the relationships among DCT coefficients in blocks A, B and C are very complex, we prefer to build them up by way of artificial neural network rather than linear combination. Here, we take the coefficients in blocks A and B as inputs. The DCT coefficients of block C in the original test image are taken as the desired outputs. Since we have inputs and desired outputs, the MLP model can be applied to get weights of connection line between neurons.

Before going any further, let’s have a brief look at the design of our artificial neural network. The number of neurons in the input layer is eight, and the number of neurons in the output layer is four. Moreover, there is only one hidden layer in the structure, and the number of neurons in the hidden layer is six. Now, let’s go over our new method step by step.

There are two phases in our proposed method. The first one is the learning phase, and the other is the blocking effects removing phase. In the learning phase, we must calculate the weights in the connection lines. In this phase, three steps have
to be taken to accomplish the mission. To begin with, we come by the compressed image $O'$ from the original test image $O$ by way of DCT. Therefore, each block in $O'$ has 1 DC and 63 AC coefficients. For each pair of horizontally adjacent blocks in $O'$, we denote the left block as block $A$, and the right as block $B$.

As for block $C$, we get it from the original image $O$ instead of the compressed image $O'$. In Section 2, we mentioned that block $C$ can be constructed by combining the right half of block $A$ and left half of block $B$ in the decompressed image. However, in our method, block $C$ comes directly from the original image $O$. After drawing out the range block $C$ falls in, it goes through DCT to get coefficients. Here, these coefficients do not go through quantization, and therefore no quantization table is required.

What happens in the second step of the learning phase is that the neurons of the input layer get obtained from the AC coefficients of block $A$ and block $B$ in compressed image $O'$. Those inputs are $F_A(0, v)$ and $F_B(0, v)$, where $v = 1, 3, 5, 7$. The desired outputs, namely $F_C(0, v)$, where $v = 1, 3, 5, 7$, can be extracted from the AC coefficients in block $C$. For each pair of horizontally adjacent blocks, we can get their inputs and the corresponding desired outputs. After going through the whole test image, we can get a set of inputs and their corresponding desired outputs.

In the third step, the inputs and their corresponding desired outputs are
partitioned into several subsets according to their input numbers of non-zero AC coefficients among $F_A(0, v)$ and $F_B(0, v)$, where $v = 1, 3, 5, 7$. The inputs that have the same numbers of non-zero AC coefficients in block A and the same numbers of non-zero AC coefficients in block B are collected together and put in the same subset. As a result, the total number of subsets is $4 \times 4$, which is equal to 16.

If there is any subset at all whose input numbers of non-zero AC coefficients in block A and block B are both smaller than 3, then the subset is discarded. They are called non-detail subsets. Otherwise, the subset is a detail subset. For each detail subset, we use a back-propagations algorithm to learn the weights of connection lines. This way, we can dynamically choose the weights of connection lines to get outputs according to the input numbers of non-zero AC coefficients in block A and block B.

Next, we enter the second phase called the blocking effects removing phase. In this phase, the blocking effects will be removed by using the artificial neural network. For each pair of horizontal adjacent blocks in the decompressed image, we first transform the two adjacent horizontal blocks into the DCT domain by using DCT transform. Then, we check the number of non-zero AC coefficients to decide whether the input belongs to the detail subset or non-detail subset. If the input belongs to the non-detail subset, we treat this block as a smooth block. As a result, the vertical boundary line is eliminated by using interpolation. The formula is as
follows.

\[ D_A(u, 7) = D_A(u, 6) + (D_B(u, 1) - D_A(u, 6))/3, \]  
\[ D_B(u, 0) = D_B(u, 1) - (D_B(u, 1) - D_A(u, 6))/3, \]  

where \( u \) ranges from 0 to 7. In the formula, \( D_A(u, v) \) and \( D_B(u, v) \) represent pixel values in the decompressed blocks. That is, \( D_A(u, v) \) represents the pixel values at position \((u, v)\) in the block A sized \(8 \times 8\), which goes through inverse DCT.

The meaning of \( D_A(u, v) \) and \( D_B(u, v) \) is shown in Fig. 3.

If the input belongs to the detail subset, we take this block as a detail block. That means the artificial neural network is involved to eliminate blocking effects. At the beginning, the subset where the inputs belong is decided according to the input numbers of non-zero AC coefficients in block \( A \) and block \( B \). After that, the inputs are fed into the artificial neural network to get the outputs if necessary. The outputs will be used to fill into the DCT coefficients in block \( C \).

We first draw out a block with \(8 \times 8\) pixels according to the description in Section 2 from the decompressed image. Then, we transform this block into the DCT domain. This way, we have one DC coefficient and 63 AC coefficients for block \( C \). The DC coefficient value in block \( C \) is modified as the mean value of the DC coefficients in block \( A \) and block \( B \).

\[ F_C(0, 0) = (F_A(0, 0) + F_B(0, 0))/2. \]
As for the AC coefficients, we fill the four outputs into $F_C(0,1)$, $F_C(0,3)$, $F_C(0,5)$ and $F_C(0,7)$ sequentially. Please note that there is still an important thing in the filling procedure. Because of the nature of the artificial neural network, the original values of $F_C(0,1)$, $F_C(0,3)$, $F_C(0,5)$ and $F_C(0,7)$ may be far away from the four outputs in certain cases. Therefore, it would be better not to modify $F_C(0,1)$, $F_C(0,3)$, $F_C(0,5)$ and $F_C(0,7)$ but keep their original values. Then, the modified block $C'$ can be obtained by using inverse DCT. After storing modified block $C'$ into its original position in the decompressed image, the block effects in block $C$ can be eliminated. By the same token, we can also eliminate horizontal boundary lines.

In the final step, like that presented in Ying and Rabab’s, we also use the Sigma filter [3] to improve the image quality even more. Because removing block effects might improve other encoding artifacts due to quantization such as ringing effects and uneven appearance in areas, we can use this edge-preserving smoothing spatial filter to smooth our resultant images.

4. Experimental Results

In this section, we shall show our experimental results and evaluate the performance of our method. To begin with, let’s have a look at our test images Lena, Baboon, and Plan, which are shown in Fig. 4. Because there are both detail areas
and non-detail areas in the Lena image, we take it as our training image. In the learning phase, we first use a JPEG compression method to compress the training image. The compression rate is 1.62 bpp (bit per pixel). With the original training image and compressed image, the artificial neural network can be applied to learn the weights of connection lines in MLP model.

![Fig. 4. The test images with 512×512 pixels](image)

The inputs come from the compressed training image, and the desired outputs come from the original training image. The whole learning procedure is in Section 3. After that, the weights of the connection lines can be obtained. In the blocking effects removing phase, it is the decompressed images that are processed. The experimental results are shown in Figs. 5 through 10. In Figs. 5 to 7, the test image is Lena. Here, detail areas and non-detail areas are shown in Fig. 5 and Fig.6, respectively. After that, the whole resultant images are presented in Fig. 7. Let’s go over the figures one by one.
In Fig. 5, we show the detail areas of the test image Lenna. All the figures here are partial images, the hair part to be more precise, of our training image. Fig 5(a) is the partial image of the original image, and Fig. 5(b) is the partial image of the decompressed image. The resultant partial images, which come from our method and Ying and Rabab’s method “without” going through the Sigma filter, are presented in Fig. 5(c) and Fig. 5(d), respectively. The resultant partial images, which come from our method and Ying and Rabab’s method “with” the Sigma filter, are presented in Fig. 5(e) and Fig. 5(f), respectively.
(c) Resultant partial image of our proposed method “without” Sigma filter
(d) Resultant partial image of Ying and Rabab’s method “without” Sigma filter
(e) Resultant partial image of our proposed method “with” Sigma filter
(f) Resultant partial image of Ying and Rabab’s method “with” Sigma filter

Fig. 5. The detail areas in resultant images

We can see that the experimental result in Fig 5(c) is much better than that in Fig 5(d). In the picture shown in Fig. 5(d), there are obvious blocking effects; however, the blocking effects in Fig. 5(c) are not so many as those in Fig 5(d). Although blocking effects in Fig. 5(f) are fewer after Fig. 5(d) goes through the Sigma filter, the partial image in Fig. 5(f) is still worse than the one processed by our method.

In the next figures, we show the non-detail (smooth) areas in the resultant images.
The partial images are the face part of the Lena image. It is obvious that Ying and Rabab’s method still leaves quite a number of blocking effects in the eyes, nose, and mouth. In contrast, our method has eliminated most of the blocking effects. Therefore, the quality of our method is much better than that of Ying and Rabab’s method. It seems that Ying and Rabab’s method works well on smooth areas, such as the facial skin. However, it does not work very well as far as detail areas are concerned. After going through the Sigma filter, we can see in Fig. 6(f) that the outline of the face still suffers from blocking effects. With our method, we have got good results in the outline of the face in Fig. 6(e). In Fig. 7, we show the whole images.
Fig. 6. The non-detail (smooth) areas in resultant images
(a) The original image
(b) The decompressed image

(c) Resultant image of our proposed method “without” Sigma filter
(d) Resultant image of Ying and Rabab’s method “without” Sigma filter

(e) Resultant image of our proposed method “with” Sigma filter
(f) Resultant image of Ying and Rabab’s method “with” Sigma filter
Fig. 7. The whole resultant images

Now, we shall demonstrate that the learning weighs of connection lines learned from the training image Lena can be used in other images. In this experiment, some images with complex or simple contents are taken as test images. They are Banboon and Airplane, which are shown in Fig. 4(b) and Fig. 4(c), respectively. In Fig. 8, the partial images are of the left part of the hair portion of the Baboon image. This part is the most complex part of the whole image.
In Fig. 9, the partial images are the top-left part of the Airplane image. This part is simple, but it is not the simplest part. If all the pixels in a certain part (simplest part) are almost the same, the interpolation in our method can remove the blocking effects easily. In order to show that our method can also work in areas with simple contents, the results of our experiment done to simple parts are shown in Fig 8. Fig. 10 then shows the final experimental results in the whole images.
Fig. 9. The non-detail (smooth) areas in resultant images
5. Conclusions

Many methods have been brought up and used to eliminate blocking effects. For example, Ying and Rabab’s method uses the existing correlation information between the existing blocks to eliminate block effects. In their method, they believe that the relationship between correlation information is linear. However, the
relationship between correlation information is far too complex to be linear. In order
to build up the relationship, in this paper, we have tried the artificial neural network,
and the results are quite good. After going through the learning phase, weights of
connection lines can be obtained. Then, in the block effects removing phase, the
weights can be used to eliminate the blocking effects.

According to the experimental results, our method can get high quality final
resultant images. This also demonstrates that our original assumption that the
relationship between correlation information is very complex is true. However, the
learning time may be a little too long. Fortunately, once we get the weights of
connection lines, they can be reused again and again when other images are to be
refined. Unlike the learning process, the blocking effects removing phase takes little
time. In a word, our new scheme as a whole is quite a practical method to eliminate
blocking effects.
References


