Interdisciplinary Information Sciences, Vol. 9, No. 2, pp. 201–208 (2003)

Color Image Compression Algorithm Using Self-Organizing Feature Map

Kazuyuki TANAKA*, Norihiro HOSHI and Tsuyoshi HORIGUCHI

Department of Computer and Mathematical Sciences, Graduate School of Information Sciences, Tohoku University, Aramaki-aza-aoba 04, Aoba-ku, Sendai 980-8579, Japan
E-mail: kazu@statp.is.tohoku.ac.jp

Received July 2, 2002; final version accepted April 3, 2003

A new color image compression algorithm using Kohonen's self-organizing feature map is proposed. Our algorithm is an extension of color image compression algorithm proposed by Pei and Lo [*IEEE Trans. Circuits Syst. Video Technol.*, **8**: 191–205 (1998)]. N neurons are introduced in order to reduce a given full color image with 2^{24} colors to an indexed color image with N colors. There are control parameters for the competitive learning among neurons in the self-organizing feature map algorithm. In our proposed algorithm, some of the control parameters, which are included in a neighboring function defined for neurons, are updated by taking relationship among neighboring neurons into account, though all control parameters are updated so as to decrease monotonically and exponentially with respect to each iteration step in Pei and Lo's algorithm. The color palette obtained by the proposed algorithm is more robust as for control parameters than that by Pei and Lo's algorithm.

KEYWORDS: image compression, feature detection, color palette, self-organizing feature map, vector quantization

1. Introduction

Among image processing techniques, compression of color images is important for practical applications. A full color image has 2^{24} kinds of colors at each pixel. Hence, for example, the data size of a full color image with 512×512 pixels is about 768 Kbytes. In one of methods for the compression of the full color image, a color palette is constructed from a given original image [1, 2]. The construction of the color palette is based on the feature detection from the given original image. The feature direction was often done by using a competitive learning in the neural network [3–6]. The competitive learning algorithm was generalized as a form of self-organizing feature map (SOFM) by Kohonen [7–10]. The SOFM is applicable to not only feature detection from a data but also clustering of a data. Some authors applied the SOFM to the image processing of gray-level monochrome image [11–13].

Pei and Lo [14]. investigated compression of color images and showed that the color palette is obtained by means of the SOFM. Pei and Lo used N neurons and they reduced a given full color image with 2^{24} colors to an indexed color image with N colors. The SOFM has usually a learning rate and a neghboring function for position vectors of neurons. In the SOFM, it is important how to update the learning rate of the competitive learning and some parameters included in the neighboring function. Pei and Lo [14]. updated them as decreasing monotonically in their algorithm. It is doubtful if the iterative algorithm could avoid local minimum states by Pei and Lo's algorithm. Hence it is interesting to investigate how the performance in color image compression is improved by applying the competitive learning to the update of some control parameters in the neighboring function.

In the present paper, we propose a new algorithm, as an extension of Pei and Lo's algorithm, for color image compression and compare it with the original Pei and Lo algorithm by some numerical experiments. A major point of the extension is that a relationship between neighboring neurons is introduced according to distances between neurons and given data. Some of control parameters in the neighboring function are updated not by decreasing monotonically but by using a competitive learning in terms of the relationship between neighboring neurons.

In Sect. 2, we propose a new algorithm for color image compression. In Sect. 3, we give some numerical experiments. Concluding remarks are given in Sect. 4.

2. Limited Color Palette Design by Self-Organizing Feature Map Algorithm

In this section, we propose a new algorithm for the limited color palette design by taking a relationship among neighboring neurons into account in updating some control parameters in the competitive learning of neurons. In order to obtain a compressed color image from the original color image by means of the obtained color palette, we adopt the fast encoding algorithm and the vector quantization training algorithm proposed by Pei and Lo [14].

We consider a full color image with 2^{24} colors on a finite square lattice with $2^L \times 2^L$ pixels. In full color images with

* Corresponding author.

2^{24} colors, we assume that each pixel has three components, which are red, green and blue. The color on an lth pixel is denoted by $\boldsymbol{X}_l$. The color $\boldsymbol{X}_l$ is a three-dimensional vector whose components are the grades for red, green and blue on the lth pixel. The original color image is represented by $\{\boldsymbol{X}_l \mid l = 1, 2, \ldots, 2^{2L}\}$. The intensity of each component is represented in terms of 2^8 values, respectively. Each pixel takes a color in 2^{24} kinds of colors. We reduce such a full color image with 2^{24} colors to an indexed color image with N colors, where we assume $N \ll 2^{24}$.

We divide the $2^L \times 2^L$ indexed color image $\{\boldsymbol{X}_l\}$ to $2^{L-B} \times 2^{L-B}$ blocks. Each block has $2^B \times 2^B$ pixels. The labels $1, 2, \ldots, 2^{2(L-B)}$ are assigned to the $2^{L-B} \times 2^{L-B}$ blocks. We assign a random number selected from the set $\{1, 2, \ldots, 2^{2B}\}$ without overlapping them each others as a label at each pixel in a block. The corresponding pixel in each block has the same label. This list is called "Sweep List." One sweep sequence is represented by "$t = 1, 2, \ldots, 2^{2(L-B)}$." All the pixels of the image are sampled by means of 2^{2B} sweep sequences through all blocks. In one sweep sequence for a fixed value of s ($s \in \{1, 2, \ldots, 2^B\}$), the pixel assigned by the same number is picked up from each block in the image. This sweep sequence is called "a butterfly-jumping sequence" [14]. The butterfly-jumping sequence is adopted in order to conserve the independence on the sampling data among pixels. In the butterfly-jumping sequence, we introduce a three-dimensional vector $\boldsymbol{X}_{s,t}$ ($s = 1, 2, \ldots, 2^B$, $t = 1, 2, \ldots, 2^{2(L-B)}$). This means that the original color image $\{\boldsymbol{X}_l \mid l = 1, 2, \ldots, 2^{2L}\}$ is represented in the form of $\{\boldsymbol{X}_{s,t} \mid s = 1, 2, \ldots, 2^{2B},\ t = 1, 2, \ldots, 2^{2(L-B)}\}$.

We denote the number of neurons by N and introduce the set $\boldsymbol{N} \equiv \{1, 2, \ldots, N\}$. The four kinds of winner neurons, $\gamma_1(s,t)$, $\gamma_2(s,t)$, $\gamma_3(s,t)$ and $\gamma_4(s,t)$, in each step t in an sth butterfly-jumping sequence are selected by the following equation:

$$\gamma_1(s,t) = \arg\min_{i \in \boldsymbol{N}} \|\boldsymbol{X}_{s,t} - \boldsymbol{w}_i(s,t)\|, \tag{1}$$

$$\gamma_2(s,t) = \arg\min_{i \in \boldsymbol{N} \setminus \{\gamma_1(s,t)\}} \|\boldsymbol{X}_{s,t} - \boldsymbol{w}_i(s,t)\|, \tag{2}$$

$$\gamma_3(s,t) = \arg\min_{i \in \boldsymbol{N} \setminus \{\gamma_1(s,t), \gamma_2(s,t)\}} \|\boldsymbol{X}_{s,t} - \boldsymbol{w}_i(s,t)\|, \tag{3}$$

$$\gamma_4(s,t) = \arg\min_{i \in \boldsymbol{N} \setminus \{\gamma_1(s,t), \gamma_2(s,t), \gamma_3(s,t)\}} \|\boldsymbol{X}_{s,t} - \boldsymbol{w}_i(s,t)\|. \tag{4}$$

Here, we adopt the Euclidean distance as the norm $\|\cdots\|$. After determining the winner neurons $\gamma_1(s,t)$, $\gamma_2(s,t)$, $\gamma_3(s,t)$ and $\gamma_4(s,t)$, the weight vector is updated as follows:

$$\boldsymbol{w}_i(s,t+1) = \boldsymbol{w}_i(s,t) + \alpha(s) h_{\gamma_1(s,t),i}(s,t)\big(\boldsymbol{X}_{s,t} - \boldsymbol{w}_i(s,t)\big) \qquad (i = 1, 2, \ldots, N). \tag{5}$$

We introduce a position vector of neuron i in the two different ways; one of them is a one-dimensional position vector:

$$\boldsymbol{r}_i = i, \tag{6}$$

and another one is a two-dimensional position vector:

$$\boldsymbol{r}_i = \left(\left\lfloor \frac{i-1}{\sqrt{N}} \right\rfloor + 1, \big((i-1) \bmod \sqrt{N}\big) + 1 \right). \tag{7}$$

Hereafter, we refer each case of Eq. (6) and Eq. (7) as "1D neuron" and "2D neuron," respectively. The neighboring function $h_{\cdot,i}(s,t)$ is defined by

$$h_{\gamma_1(s,t),i}(s,t) \equiv \exp\left(-\frac{\|\boldsymbol{r}_{\gamma_1(s,t)} - \boldsymbol{r}_i\|^2}{2\sigma_i(s,t)^2} \right). \tag{8}$$

The parameters $\{\sigma_i(s,t) \mid i = 1, 2, \ldots, N\}$ and $\alpha(s)$, included in $\sigma_{\gamma_1(s,t)}(s,t+1)$ given below, are assumed to decrease with the sweep step s and the iteration step r as follows:

$$\alpha(s) = 0.99 \times {k_1}^{sr}, \tag{9}$$

$$\sigma_{\gamma_1(s,t)}(s,t+1) = \min\left\{ \sigma_{\gamma_1(s,t)}(s,t)(1 + \alpha(s)\Delta(s,t)), \sigma_{\mathrm{UB}} \right\} \qquad (t = 1, 2, \ldots, 2^{2(L-B)},\ s = 1, 2, \ldots, 2^{2B}), \tag{10}$$

$$\sigma_i(s,1) = \sigma_{\mathrm{UB}} \times {k_2}^{sr} \qquad (i = 1, 2, \ldots, N,\ \text{1D neuron and 2D neuron}), \tag{11}$$

$$\sigma_{\mathrm{UB}} = 10 \qquad (\text{1D neuron}), \tag{12}$$

$$\sigma_{\mathrm{UB}} = 5 \qquad (\text{2D neuron}). \tag{13}$$

where k_1 and k_2 are control parameters. In each case of the 1D and the 2D neurons, $\Delta(s,t)$ is defined as follows:

$$\Delta(s,t) \equiv \frac{1}{2(N-1)} \left\{ \|\boldsymbol{r}_{\gamma_1(s,t)} - \boldsymbol{r}_{\gamma_2(s,t)}\| \left(1 - \delta_{\|\boldsymbol{r}_{\gamma_1(s,t)} - \boldsymbol{r}_{\gamma_2(s,t)}\|,1}\right) + \|\boldsymbol{r}_{\gamma_1(s,t)} - \boldsymbol{r}_{\gamma_3(s,t)}\| \left(1 - \delta_{\|\boldsymbol{r}_{\gamma_1(s,t)} - \boldsymbol{r}_{\gamma_3(s,t)}\|,1}\right) \right\} \qquad (\text{1D neuron}), \tag{14}$$

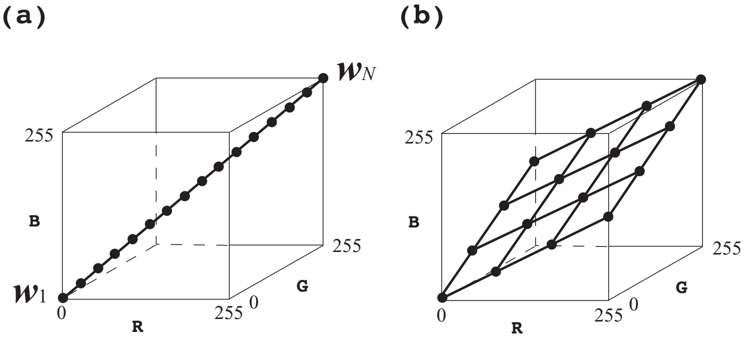

Fig. 1. Initial weight vector $\boldsymbol{w}_i(0)$ ($N = 16$). (a) 1D neuron. (b) 2D neuron.

$$\begin{aligned}\Delta(s,t) \equiv \frac{1}{3\sqrt{2}(L-1)} \Big\{ &\|\boldsymbol{r}_{\gamma_1(s,t)} - \boldsymbol{r}_{\gamma_2(s,t)}\| \big(1 - \delta_{\|\boldsymbol{r}_{\gamma_1(s,t)} - \boldsymbol{r}_{\gamma_2(s,t)}\|,1}\big) \\ &+ \|\boldsymbol{r}_{\gamma_1(s,t)} - \boldsymbol{r}_{\gamma_3(s,t)}\| \big(1 - \delta_{\|\boldsymbol{r}_{\gamma_1(s,t)} - \boldsymbol{r}_{\gamma_3(s,t)}\|,1}\big) \\ &+ \|\boldsymbol{r}_{\gamma_1(s,t)} - \boldsymbol{r}_{\gamma_4(s,t)}\| \Big(1 - \delta_{\|\boldsymbol{r}_{\gamma_1(s,t)} - \boldsymbol{r}_{\gamma_4(s,t)}\|,\sqrt{2}}\Big) \Big\} \quad \text{(2D neuron)},\end{aligned} \tag{15}$$

respectively. The notation $\delta_{a,b}$ is Kronecker's delta. The initial state of the weight vector $\boldsymbol{w}_i(0)$ is set as

$$\boldsymbol{w}_i(0) = \left(\frac{255(i-1)}{N-1}, \frac{255(i-1)}{N-1}, \frac{255(i-1)}{N-1}\right) \quad (i = 1, 2, \ldots, N;\ \text{1D neuron}), \tag{16}$$

$$\begin{aligned}\boldsymbol{w}_i(0) = \Bigg(&\left\lfloor \frac{i-1}{\sqrt{N}} \right\rfloor \times \frac{255}{\sqrt{N}-1}, \Big((i-1) \bmod \sqrt{N}\Big) \times \frac{255}{\sqrt{N}-1}, \\ &\left\{\Big((i-1) \bmod \sqrt{N}\Big) + \left\lfloor \frac{i-1}{\sqrt{N}} \right\rfloor \right\} \times \frac{255}{2(\sqrt{N}-1)} \Bigg) \quad (i = 1, 2, \ldots, N;\ \text{2D neuron}).\end{aligned} \tag{17}$$

In the case of $N = 16$, the initial weight vector $\boldsymbol{w}_i(0)$ is given in Fig. 1. We repeat the procedure given by Eqs. (1)–(17) for $r = 1, 2, 3, \ldots$. When it is satisfied that $\alpha(s) < 0.001$, the above procedure is stopped and the obtained position vectors $\{\boldsymbol{r}_i(2^{2(L-B)}) \mid i = 1, 2, \ldots, N\}$ and their corresponding colors $\{\boldsymbol{w}_i(2^{2B}, 2^{2(L-B)}) \mid i = 1, 2, \ldots, N\}$ are substituted to $\{\boldsymbol{r}_i \mid i = 1, 2, \ldots, N\}$ and $\{\boldsymbol{w}_i \mid i = 1, 2, \ldots, N\}$, respectively. The set of vectors $\{\boldsymbol{r}_i \mid i = 1, 2, \ldots, N\}$ and $\{\boldsymbol{w}_i \mid i = 1, 2, \ldots, N\}$ is referred as a "color palette."

From the color $\boldsymbol{X}_l$ at each pixel in the given original image, the indexed color $\widehat{\boldsymbol{X}}_l$ at the corresponding pixel is given as follows:

$$\widehat{\boldsymbol{X}}_l = \boldsymbol{w}_{\nu_l}, \tag{18}$$

where

$$\nu_l = \arg \min_{i \in \{1,2,\ldots,N\}} \|\boldsymbol{X}_l - \boldsymbol{w}_i\|, \tag{19}$$

for $l = 1, 2, \ldots, 2^{2L}$. The notations $\widehat{R}_l$, $\widehat{G}_l$ and $\widehat{B}_l$ represent the grades of red, green and blue at the corresponding pixel in the indexed color image, respectively. We adopt the fast encoding algorithm proposed in [14] in order to search the indexed color $\widehat{\boldsymbol{X}}_l$ for all the pixels. We explain the fast encoding algorithm now. For each pixel l, we determine the index ν_l as follows:

$$\nu_l = \begin{cases} \arg \min\limits_{N_{\min} \le i \le N_{\max}} \|\boldsymbol{X}_l - \boldsymbol{w}_i\| & (\|\boldsymbol{X}_l - \boldsymbol{X}_{l-1}\| \le 30) \\ \arg \min\limits_{i=1,2,\ldots,N} \|\boldsymbol{X}_l - \boldsymbol{w}_i\| & (\|\boldsymbol{X}_l - \boldsymbol{X}_{l-1}\| > 30) \end{cases}, \tag{20}$$

where

$$N_{\min}(l) \equiv \max\left(1, \nu_{l-1} - \frac{N}{8}\right), \tag{21}$$

$$N_{\max}(l) \equiv \min\left(\nu_{l-1} + \frac{N}{8}, N\right). \tag{22}$$

From the obtained set $\{\nu_l \mid l = 1, 2, \ldots, 2^L\}$, the indexed color image $\{\widehat{\boldsymbol{X}}_l \mid l = 1, 2, \ldots, 2^{2L}\}$ is determined by Eq. (18). In the fast encoding algorithm in the present paper, we adopt the Euclidean distance as a norm $\|\cdots\|$. The obtained color indexed image $\{\widehat{\boldsymbol{X}}_l \mid l = 1, 2, \ldots, 2^{2L}\}$ is compressed by means of the vector quantization training algorithm. This algorithm was proposed in [14] and the detailed explanation is given in Appendix A. The recovered color indexed image in the compression algorithm by using the vector quantization is denoted by $\{\boldsymbol{Z}_l \mid l = 1, 2, \ldots, 2^{2L}\}$.

Fig. 2. Original full color image $\{\boldsymbol{X}_l\}$ $(L = 9)$.

In the algorithm of the SOFM for the limited color palette design proposed in [14], the parameters $\sigma(m,t)$ decrease with sweep step m as follows:

$$\sigma_i(s,t) = 20 \times {k_2}^{sr} \qquad (i = 1,2,\ldots,N,\ \text{1D neuron}), \tag{23}$$

$$\sigma_i(s,t) = 5 \times {k_2}^{sr} \qquad (i = 1,2,\ldots,N,\ \text{2D neuron}), \tag{24}$$

instead of Eqs. (10) and (11).

3. Numerical Experiments

In this section, we give some results by numerical experiments for the limited color palette design the color on image compression from some standard color image, "*Mandrill*," shown in Fig. 2. We compare the results by the proposed algorithm for the limited color palette design with those by Pei and Lo's algorithm in [14].

In order to discuss the quality of indexed color images in our proposed algorithm for practical application, we introduce two kinds of quantities as follows. First we define a peak signal to noise ratio (PSNR) $R(\{\boldsymbol{X}_l\},\{\boldsymbol{Z}_l\})$ (dB) as follows:

$$R(\{\boldsymbol{X}_l\},\{\boldsymbol{Z}_l\}) \equiv -10\log_{10}\left(\frac{1}{(255\sqrt{3})^2 2^L}\sum_{l=1}^{2^L}\|\boldsymbol{X}_l - \boldsymbol{Z}_l\|^2\right) \text{ (dB)}. \tag{25}$$

Next, in order to discuss the similarity between the neighboring weight vectors in $\{\boldsymbol{w}_i \mid i = 1,2,\ldots,N\}$, we define the following quantity:

$$D(\{\boldsymbol{w}_i\}) \equiv \begin{cases} \dfrac{1}{N-1}\displaystyle\sum_{i=1}^{N-1}\|\boldsymbol{w}_i - \boldsymbol{w}_{i+1}\|^2 & \text{(1D neuron)}, \\ \dfrac{1}{\sqrt{N}(\sqrt{N}-1)}\displaystyle\sum_{i=1}^{N-1}\sum_{j\in D_i}\|\boldsymbol{w}_i - \boldsymbol{w}_j\|^2 & \text{(2D neuron)}, \end{cases} \tag{26}$$

where $D_i \equiv \{j \mid \|\boldsymbol{r}_i - \boldsymbol{r}_j\| = 1\}$. In our numerical experiments, the number of neuron, N, is set to 256. In Fig. 3, we see the robustness of control parameters k_1 and k_2 in the quantity $D(\{\boldsymbol{w}_i\})$. In Figs. 4 and 5, the robustness of control parameters k_1 and k_2 in the quantities $R(\{\boldsymbol{X}_l\},\{\boldsymbol{Z}_l\})$ (dB) is seen for the cases of $a = b = c = 2$ and $a = b = c = 4$, respectively. In Fig. 6, the color palette $\{\boldsymbol{w}_i\}$ and the recovered color indexed image $\{\boldsymbol{Z}_l\}$ are given for the case of 2D neuron.

We have found that, the similar results are obtained in more wide region for control parameters k_1 and k_2 in the proposed algorithm than in Pei and Lo's algorithm. We have also done the similar numerical experiments for the standard images "*Lena*" and "*Pepper*" and have obtained good results for both the standard images, too.

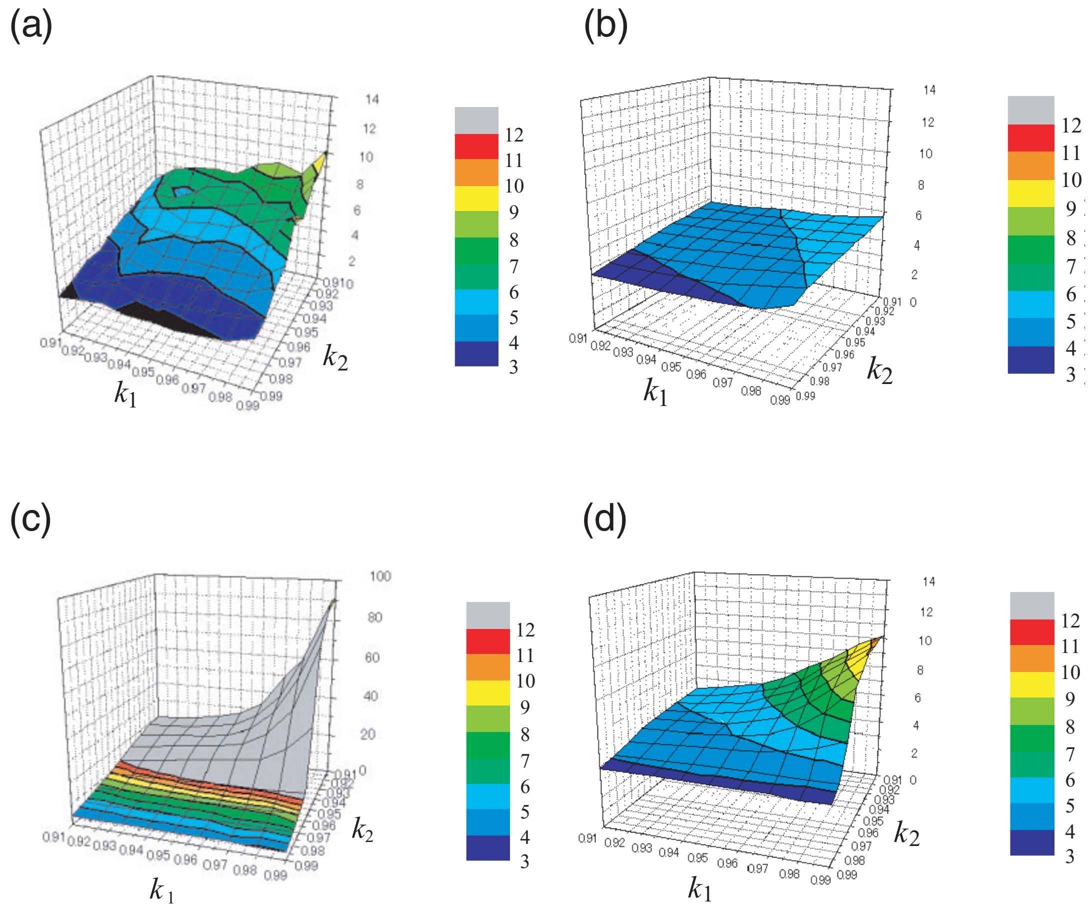

Fig. 3. Dependence of control parameters k_1 and k_2 of $D(\{\boldsymbol{w}_i\})$ ($N = 256$). (a) Proposed method by means of 1D neuron. (b) Proposed method by means of 2D neuron. (c) Pei and Lo's method by means of 1D neuron. (d) Pei and Lo's method by means of 2D neuron.

4. Concluding Remarks

In this paper, we have proposed a new color image compression algorithm using the SOFM. Our proposed algorithm is an extension of Pei and Lo's color image compression algorithm [14]. The extension is done by taking a relationship among neighboring neurons into account. We have found the more robustness in the obtained color palettes as for control parameters than in those by Pei and Lo's algorithm.

It is interesting to investigate how the performance of the ordering in the color palette can be improved by taking more neurons as winner neurons in Eqs. (14) and (15). It is expected also that the robust region is expanded by improving the update criteria of the coefficient $\alpha(s)$ in Eq. (5). We notice that the present algorithm is expected to apply to the feature detections not only in color images but also in motion images and in sound processing. These are future problems.

Appendix: Vector Quantization Training Algorithm for Compression of Color Index Image

In this appendix, we explain the vector quantization training algorithm for the compression of the obtained indexed color image $\{\widehat{\boldsymbol{X}}_l \mid 1, 2, \ldots, 2^{2L}\}$. The algorithm was proposed in [14].

We divide the $2^L \times 2^L$ indexed color image $\{\widehat{\boldsymbol{X}}_l \mid 1, 2, \ldots, 2^{2L}\}$, which is obtained from the original full color image $\{\boldsymbol{X}_l \mid 1, 2, \ldots, 2^{2L}\}$ by the SOFM in Sect. 2, to $2^{L-a} \times 2^{L-a}$ blocks. Each block has $2^a \times 2^a$ pixels. The labels $1, 2, \ldots, 2^{2(L-a)}$ are assigned to the $2^{L-a} \times 2^{L-a}$ blocks. When the indexed color image $\{\widehat{\boldsymbol{X}}_l \mid l = 1, 2, \ldots, 2^{2L}\}$ is divided to the $2^{L-a} \times 2^{L-a}$ blocks in the above way, the indexed color image $\{\widehat{\boldsymbol{X}}_l \mid l = 1, 2, \ldots, 2^{2L}\}$ is represented in the form of $\{\widehat{\boldsymbol{X}}_{s,t} \mid s = 1, 2, \ldots, 2^{2a};\ t = 1, 2, \ldots, 2^{2(L-a)}\}$. From the indexed color image $\{\widehat{\boldsymbol{X}}_{s,t} \mid s = 1, 2, \ldots, 2^{2a};\ t = 1, 2, \ldots, 2^{2(L-a)}\}$, we produce a $2^{L-a} \times 2^{L-a}$ block mean indexed color image $\{\boldsymbol{F}_l \mid l = 1, 2, \ldots, 2^{2(L-a)}\}$; we assign the mean of colors $\{\widehat{\boldsymbol{X}}_{s,l} \mid s = 1, 2, \ldots, 2^{2a}\}$ in each lth block of the indexed color image to a color $\boldsymbol{F}_l$ of the lth pixel of the block mean indexed color image, where $l = 1, 2, \ldots, 2^{2(L-a)}$. We again divide the obtained $2^{L-a} \times 2^{L-a}$ block mean indexed color image $\{\boldsymbol{F}_l \mid l = 1, 2, \ldots, 2^{2(L-a)}\}$ to blocks with $2^b \times 2^b$ pixels. After the block mean indexed color image $\{\boldsymbol{F}_l \mid l = 1, 2, \ldots, 2^{2(L-a)}\}$ is divided to the $2^{L-a} \times 2^{L-a}$ blocks in the above way, we represent the indexed color image

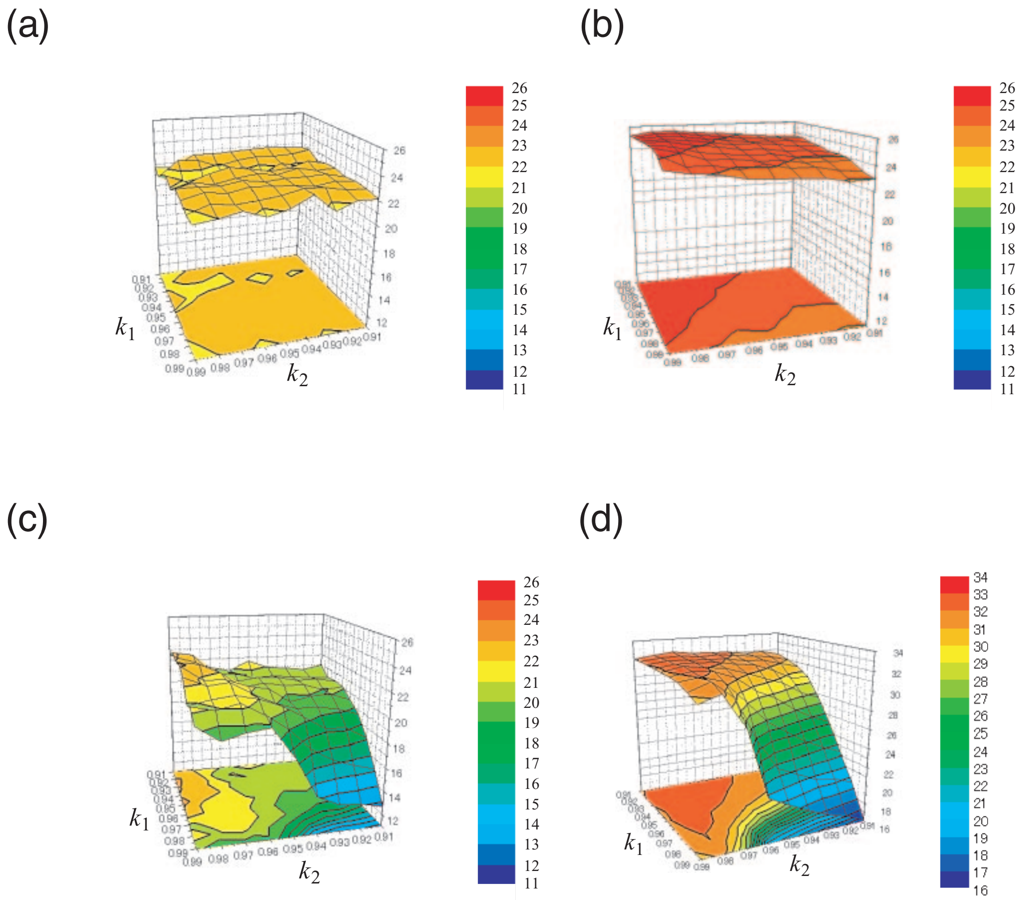

Fig. 4. Dependence of control parameters k_1 and k_2 of $R(\{\boldsymbol{X}_l\},\{\boldsymbol{Z}_l\})$ (dB) ($N = 256$, $a = b = c = 2$). (a) Proposed method by means of 1D neuron. (b) Proposed method by means of 2D neuron. (c) Pei and Lo's method by means of 1D neuron. (d) Pei and Lo's method by means of 2D neuron.

$\{\boldsymbol{F}_l \mid l = 1, 2, \ldots, 2^{2(L-a)}\}$ in the form of $\{\boldsymbol{F}_{s,t} \mid s = 1, 2, \ldots, 2^{2b},\ t = 1, 2, \ldots, 2^{2(L-a-b)}\}$. From the indexed color image $\{\boldsymbol{F}_{s,t} \mid s = 1, 2, \ldots, 2^{2b},\ t = 1, 2, \ldots, 2^{2(L-a-b)}\}$, we produce a $2^{L-a-b} \times 2^{L-a-b}$ second block mean indexed color image $\{\boldsymbol{G}_l \mid l = 1, 2, \ldots, 2^{2(L-a-b)}\}$. We assign the mean of colors $\{\boldsymbol{F}_{s,l} \mid s = 1, 2, \ldots, 2^{2b}\}$ in each lth block of the block mean indexed color image to a color $\boldsymbol{G}_l$ of the lth pixel of the block mean indexed color image $\{\boldsymbol{G}_l \mid l = 1, 2, \ldots, 2^{2(L-a-b)}\}$, where $l = 1, 2, \ldots, 2^{2(L-a-b)}$. By subtracting a $2^{L-a} \times 2^{L-a}$ complemented image of $\{\boldsymbol{G}_l \mid l = 1, 2, \ldots, 2^{2(L-a-b)}\}$ from the $2^{L-a} \times 2^{L-a}$ block mean indexed color image $\{\boldsymbol{F}_l \mid l = 1, 2, \ldots, 2^{2(L-a)}\}$, a $2^{L-a} \times 2^{L-a}$ deference indexed color image, $\{\boldsymbol{Y}_l \mid l = 1, 2, \ldots, 2^{2(L-a)}\}$ is produced. From the data sets, we make N code book vectors by means of a vector quantization. The $2^{L-a} \times 2^{L-a}$ deference indexed color image $\{\boldsymbol{Y}_l \mid l = 1, 2, \ldots, 2^{2(L-a)}\}$ is represented by $\{\boldsymbol{Y}_{s,t} \mid s = 1, 2, \ldots, 2^{2b},\ t = 1, 2, \ldots, 2^{2(L-a-b)}\}$ and we construct a set of vectors $\{\boldsymbol{y}_t \mid t = 1, 2, \ldots, 2^{2(L-a-b)}\}$. The tth vector has 3×2^{2b} components which are $\{\boldsymbol{Y}_{s,t} \mid s = 1, 2, \ldots, 2^{2b}\}$. We remark that each vector $\boldsymbol{Y}_{s,t}$ has three components which denote the grades of red, green and blue. Then the code book vectors $\{\boldsymbol{u}_i \mid i = 1, 2, \ldots, N\}$ are produced as follows:

$$\phi_t = \arg \min_{i=1,2,\ldots,N} \|\boldsymbol{y}_t - \boldsymbol{u}_i(t)\|, \tag{A·1}$$

$$\boldsymbol{u}_i(t+1) = \boldsymbol{u}_i(t) + \beta(t)\big(\boldsymbol{y}_t - \boldsymbol{u}_i(t)\big)\delta_{\phi_t,i}, \tag{A·2}$$

$$\beta(t) = 0.99 \times 0.9999^{tr}, \tag{A·3}$$

for $t = 1, 2, \ldots, 2^{2(L-a-b)}$ and $r = 1, 2, 3, \ldots$. Here, the label r denotes an iteration step. When $\beta(t)$ becomes less than 0.0001, the above procedure given in Eqs. (A·1)–(A·3) is stopped and the obtained vectors $\{\boldsymbol{u}_i(t) \mid i = 1, 2, \ldots, N\}$ are substituted to $\{\boldsymbol{u}_i \mid i = 1, 2, \ldots, N\}$. From the code book vectors $\{\boldsymbol{u}_i \mid i = 1, 2, \ldots, N\}$, the first indexes $\{\eta_t \mid t = 1, 2, \ldots, 2^{2(L-a-b)}\}$ is obtained by

$$\eta_t = \arg \min_{i=1,2,\ldots,N} \|\boldsymbol{y}_t - \boldsymbol{u}_i(t)\|, \tag{A·4}$$

for $t = 1, 2, \ldots, 2^{2(L-a)}$. The decoded data of the first block mean indexed color image $\{\boldsymbol{F}_l \mid l = 1, 2, \ldots, 2^{2(L-a)}\}$ is given by the code book vector $\{\boldsymbol{u}_i \mid i = 1, 2, \ldots, N\}$, $\{\eta_t \mid t = 1, 2, \ldots, 2^{2(L-a-b)}\}$ and the second block mean indexed color image $\{\boldsymbol{G}_l \mid l = 1, 2, \ldots, 2^{2(L-a-b)}\}$. The $2^{L-a} \times 2^{L-a}$ recovered first block mean indexed color image $\{\widehat{\boldsymbol{F}}_l \mid l =$

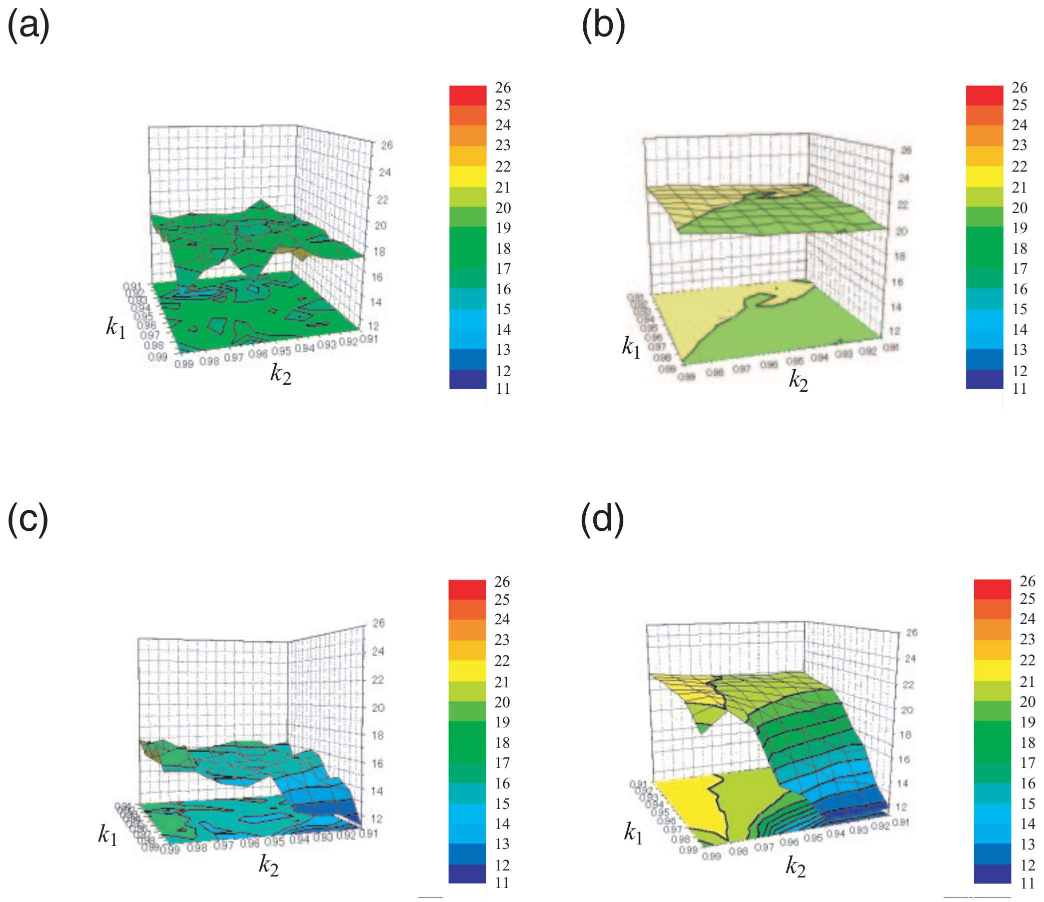

Fig. 5. Dependence of control parameters k_1 and k_2 of $R(\{\boldsymbol{X}_l\}, \{\boldsymbol{Z}_l\})$ (dB) ($N = 256$, $a = b = c = 4$). (a) Proposed method by means of 1D neuron. (b) Proposed method by means of 2D neuron. (c) Pei and Lo's method by means of 1D neuron. (d) Pei and Lo's method by means of 2D neuron.

$1, 2, \ldots, 2^{2(L-a)}\}$ is obtained by adding the $2^{L-a} \times 2^{L-a}$ complemented image of $\{\boldsymbol{G}_l \mid l = 1, 2, \ldots, 2^{2(L-a)}\}$ to a $2^{L-a} \times 2^{L-a}$ image which is produced by assigning the components of the code book vector $\boldsymbol{u}_{\eta(t)}$ to all pixels belonging to any tth block ($t = 1, 2, \ldots, 2^{2(L-a-b)}$). The $2^L \times 2^L$ recovered mean indexed color image $\{\bar{\boldsymbol{X}}_l \mid l = 1, 2, \ldots, 2^{2L}\}$ is obtained by complementing the $2^{L-a} \times 2^{L-a}$ recovered first block mean indexed color image $\{\widehat{\boldsymbol{F}}_l \mid l = 1, 2, \ldots, 2^{2(L-a)}\}$. A $2^L \times 2^L$ deference indexed color image $\{\boldsymbol{H}_l \mid l = 1, 2, \ldots, 2^{2L}\}$ is produced, by subtract the $2^L \times 2^L$ recovered mean indexed color image $\{\bar{\boldsymbol{X}}_l \mid l = 1, 2, \ldots, 2^{2L}\}$ from the $2^L \times 2^L$ indexed color image $\{\widehat{\boldsymbol{X}}_l \mid l = 1, 2, \ldots, 2^{2L}\}$. The $2^L \times 2^L$ deference indexed color image $\{\boldsymbol{H}_l \mid l = 1, 2, \ldots, 2^{2L}\}$ is represented by $\{\boldsymbol{H}_{s,t} \mid s = 1, 2, \ldots, 2^{2c},\ t = 1, 2, \ldots, 2^{2(L-c)}\}$ and we construct a set of vectors $\{\boldsymbol{z}_t \mid t = 1, 2, \ldots, 2^{2(L-c)}\}$. The tth vector has 3×2^{2c} components which are $\{\boldsymbol{H}_{s,t} \mid s = 1, 2, \ldots, 2^{2c}\}$. We remark that each vector $\boldsymbol{H}_{s,t}$ has three components which denote the grades of red, green and blue. From the data sets, we make N_2 code book vectors by means of a vector quantization. From the set of vector, $\{\boldsymbol{z}_t \mid t = 1, 2, \ldots, 2^{2(L-c)}\}$, the code book vectors, $\{\boldsymbol{v}_i \mid i = 1, 2, \ldots, N\}$, are produced as follows:

$$\psi_t = \arg \min_{i=1,2,\ldots,N} \|\boldsymbol{y}_t - \boldsymbol{v}_i(t)\|, \tag{A·5}$$

$$\boldsymbol{v}_i(t+1) = \boldsymbol{v}_i(t) + \beta(t)(\boldsymbol{z}(t) - \boldsymbol{v}_i(t))\delta_{\psi_t, i}, \tag{A·6}$$

$$\beta(t) = 0.99 \times 0.9999^{tr}, \tag{A·7}$$

for $t = 1, 2, \ldots, 2^{2(L-c)}$ and $r = 1, 2, 3, \ldots$. Here, the label r denotes an iteration step. When $\beta(t)$ becomes less than 0.0001, the above procedure given in Eqs. (A·5)–(A·7) is stopped and the obtained vectors $\{\boldsymbol{v}_i(t) \mid i = 1, 2, \ldots, N\}$ are substituted to $\{\boldsymbol{v}_i \mid i = 1, 2, \ldots, N\}$. From the code book vectors $\{\boldsymbol{v}_i \mid i = 1, 2, \ldots, N\}$, the second indexes $\{\zeta_t \mid t = 1, 2, \ldots, 2^{2(L-a)}\}$ is obtained by

$$\zeta_t = \arg \min_{i=1,2,\ldots,N} \|\boldsymbol{z}(t) - \boldsymbol{v}_i(t)\|, \tag{A·8}$$

for $t = 1, 2, \ldots, 2^{2(L-a)}$. The decoded data of the first block mean indexed color image $\{\boldsymbol{H}_l \mid l = 1, 2, \ldots, 2^{2(L-a)}\}$ is given by the code book vector $\{\boldsymbol{v}_i \mid i = 1, 2, \ldots, N\}$ and $\{\eta_t \mid t = 1, 2, \ldots, 2^{2(L-a-b)}\}$. The $2^L \times 2^L$ recovered indexed color image $\{\widehat{\boldsymbol{Z}}_l \mid l = 1, 2, \ldots, 2^{2L}\}$ is obtained by adding the $2^L \times 2^L$ recovered mean indexed image $\{\bar{\boldsymbol{X}}_l \mid l = 1, 2, \ldots, 2^{2L}\}$ to an $2^L \times 2^L$ recovered difference indexed color image $\{\widehat{\boldsymbol{H}}_l \mid l = 1, 2, \ldots, 2^{2L}\}$ which is produced by assigning the components of the code book vector $\boldsymbol{v}_{\zeta(t)}$ to all pixels belonging to any tth block ($t = 1, 2, \ldots, 2^{2(L-c)}$).

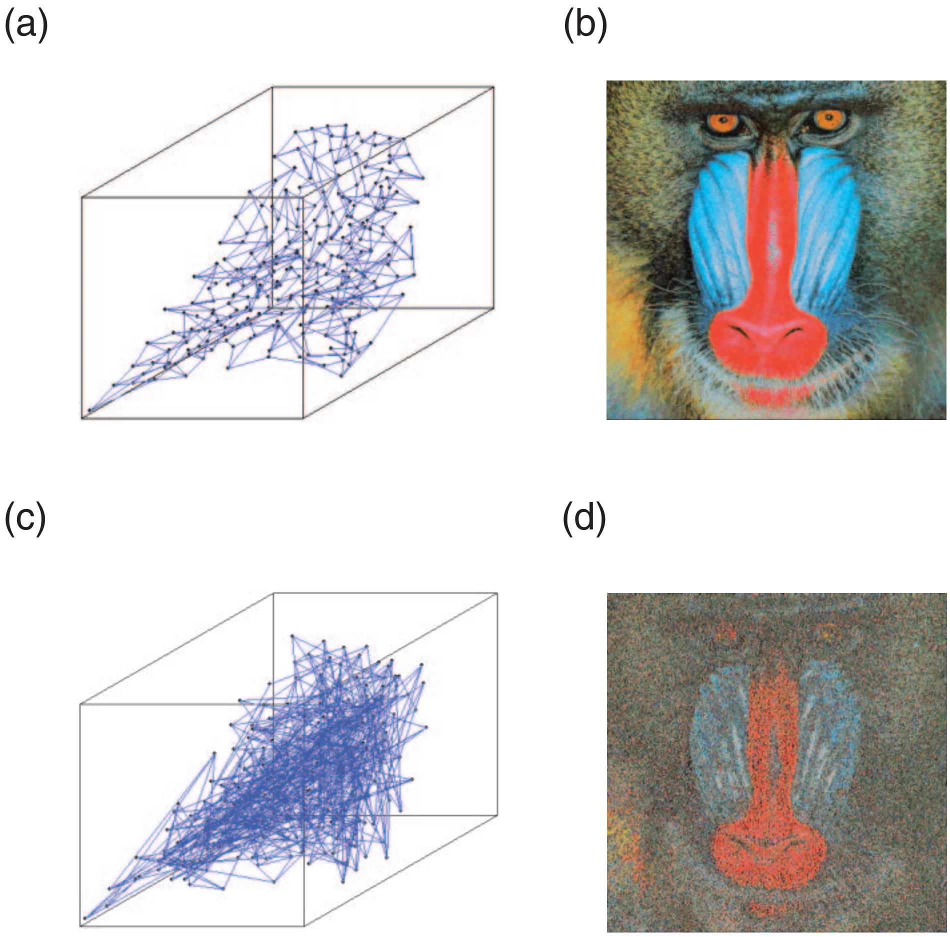

Fig. 6. Color palettes and recovered indexed color image ($k_1 = 0.99$, $k_2 = 0.92$, $a = b = c = 2$, $N = 256$). (a) Color palette $\{\boldsymbol{w}_i\}$ of the proposed method by means of 2D neuron. (b) Recovered indexed color image $\{\boldsymbol{Z}_l\}$ of the proposed method by means of 2D neuron. (c) Color palette $\{\boldsymbol{w}_i\}$ of Pei and Lo's method by means of 2D neuron. (d) Recovered indexed color image $\{\boldsymbol{Z}_l\}$ of Pei and Lo's method by means of 2D neuron.

REFERENCES

[1] Heckbert, P., "Color image quantization for frame buffer display," *Comput. Graphics*, **16**: 297–304 (1982).
[2] Zaccarin, A., and Liu, B., "A novel approach for coding color quantized images," *IEEE Trans. Image Process.*, **2**: 442–453 (1993).
[3] Durbin, R., Miall, C., and Mitchison G., The Computing Neuron, Addison-Wesley, Reading (1989).
[4] Hertz J., Krogh A., and Palmer, R. G., Introduction to the Theory of Neural Computation, Addison-Wesley, Reading (1991).
[5] Bose, N. K., and Liang, P., Neural Network Fudamentals with Graphs, Algorithms, and Applications, McGraw-Hill, Columbus (1996).
[6] Gurney, K., An Introduction to Neural Network, UCL Press, London (1997).
[7] Kohonen, T., Self-organization and Associative Memory, Springer-Verlag, New York (1989).
[8] Kohonen, T., "The self-organization map," *Proc. IEEE*, **78**: 1464–1480 (1990).
[9] Kohonen, T., "Physiological interpretation of the self-organizing map algorithm," *Neural Networks*, **6**: 895–905 (1993).
[10] Kohonen, T., Self-Organizing Maps, Springer-Verlag, New York (1997).
[11] Nasrabadi, N. M., and King, R. A., "Image coding using vector quantization: a review," *IEEE Trans. Commun.*, **36**: 957–971 (1988).
[12] Oehler, K. L., and Gray, R. M., "Combining image compression and classification using vector quantization," *IEEE Trans. Pattern Anal. Machine Intel.*, **17**: 461–473 (1995).
[13] Hang, H., and Haskell, B. G., "Interpolative vector quantization of color images," *IEEE Trans. Commun.*, **36**: 465–470 (1988).
[14] Pei, S. C., and Lo, Y. S., "Color image compression and limited display using self-organization Kohonen map," *IEEE Trans. Circuits Syst. Video Technol.*, **8**: 191–205 (1998).