Journal of Electronic Imaging 12(3), 442–447 (July 2003).

Contextual algorithm for color quantization

M. P. Yu
K. C. Lo
The Hong Kong Polytechnic University
Department of Electronic and Information Engineering
Kowloon, Hong Kong
E-mail: mpyu@eie.polyu.edu.hk

Abstract. *We propose a heuristic approach to color quantize images with contextual information taken into consideration. The idea is to locate the regions of an image having the greatest need for colors, and allocate more quantization levels to them. We achieve this by scanning the elements of the input image in a way determined by their local intensity and selecting the color representatives that comprise the color map according to their local popularity. The overall performance of the color quantization algorithm is evaluated on a representative set of artificial and real images. The experimental results indicate good performance of our proposed algorithm with the capability to focus on the regions of an image having important color information.* © *2003 SPIE and IS&T.* [DOI: 10.1117/1.1586285]

1 Introduction

Full-color displays typically use 24 bits to represent the color of each pixel on a screen. With 8 bits for each of the primary components, red, green, and blue, approximately 16.7 million possible colors can be generated in a 24-bit full-RGB-color image. In many imaging applications, it is often desirable to represent images with as few colors as possible, while at the same time retaining the optimal perceived quality of the images. This is generally achieved by color quantization, in which the number of colors in an image is reduced to a maximum of 256 simultaneously displayable colors and followed by subsequent halftoning to create the illusion of a continuous tone image. Some well-known techniques to create an image color map or palette are the popularity algorithm,[1,2] the center-cut,[3] the octree quantization,[4] the Orchard-Bouman splitting algorithm,[5] and Wu's algorithm by principal analysis.[6] Important developments in digital halftoning include analog screening, noise encoding, ordered dithering, error diffusion and stochastic screening.[7] Among the various methods, ordered dithering and error diffusion are the two main classes of conventional digital halftoning techniques. While clustered dots are often preferred for printing, error diffusion is perhaps the most widely used approach for displays.[8] Error diffusion is an adaptive algorithm that operates by spreading or diffusing the quantization error of a current pixel to neighboring pixels. This method was first proposed by Floyd and Steinberg[9] and was originally used for gray-scale images. For color images, error diffusion is either performed separately in each independent channel (scalar error diffusion) or simultaneously in a three-dimensional (3-D) color space[10] (vector error diffusion). For color displays with an image-dependent palette, vector error diffusion is often used.

All the quantization techniques just mentioned are operated in accordance with the statistical distribution of the colors in the input image and share a common initial condition attempting to allocate more quantization levels to regions of the color space with a larger number of pixels. These algorithms, though image dependent, do not account for the spatial and contextual information of the image. As a result, instead of focusing on the area of high interest in an image containing highly saturated primaries, the grays and low saturated colors are often overrepresented, resulting in the appearance of contouring artifacts, especially in regions with smooth color transitions. Aiming at this problem, we propose a heuristic approach to color quantize images with contextual information taken into account.

In our approach, we identify the regions of an image having the greatest need for colors, allocate more quantization levels to them, and then move to another part of the image in a deterministic manner. We achieve this by scanning the elements of the input image in a way determined by their local intensity and select the color representatives that comprise the color map according to their local popularity. The overall performance of the color quantization algorithm is evaluated on a representative set of artificial and real images. The results show a significant image quality improvement compared to some of the other color quantization schemes.

2 Contextual Algorithm

Consider a case to color quantize an image I containing millions of colors to an output image Q containing a maximum of 256 simultaneously displayable colors. From I, we generate two pseudo-images, $PI1$ and $PI2$, as the inputs to our algorithm.

The first pseudo-image, $PI1$, is obtained by summing up the RGB values of I at each pixel and is used to locate the areas of I containing important color information. The second pseudo-image, $PI2$, is composed of indexing values representing the exact colors of every pixel in the original

Paper 02019 received Feb. 20, 2002; revised manuscript received Nov. 13, 2002; accepted for publication Jan. 17, 2003.
1017-9909/2003/$15.00 © 2003 SPIE and IS&T.

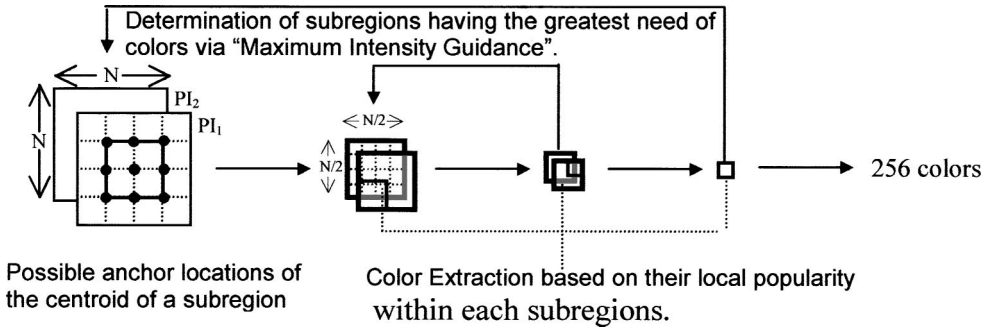


Fig. 1 Schematic diagram of the contextual algorithm.

image *I*, each of which constitutes the RGB components of a specific color such that the popularity of a color can be determined by counting the frequency of occurrence of the corresponding index value.

Our approach is to identify the regions of the image we want to emphasize from $PI1$, and then from $PI2$ in the same regions, we extract the color representatives that comprise the color map according to their local popularity. Once a color is extracted, we can identify the locations of the image having the same or visually similar colors. We then keep the locations and update the input images by assigning 0 to these locations to prevent the same color being selected in the next iteration of color extraction. Figure 1 shows a schematic diagram of the proposed algorithm.

2.1 *Determining the Regions of High Interest*

Figure 2(a) shows a 24-bits/pixel artificial image containing wide range of colors that has been uniformly quantized to contain 256 colors only, and Fig. 2(b) is the corresponding pseudo image ($PI1$) generated by summing up the RGB values of the original image at each pixel. As indicated by these figures, the regions where contouring is most likely to occur after color quantization are equivalent to the regions in the pseudo-image showing excessive brightness, i.e., regions having large sums of RGB values.

Therefore, by locating the areas of $PI1$ having the largest value, and thus the greatest need for colors, we can easily identify the parts of the image we want to emphasize and assign more colors to them. We achieve this by scanning the elements of $PI1$ in a deterministic way via "maximum intensity guidance."[11,12] The basic idea is to iteratively search the brightest region of $PI1$ by partitioning the image into subregions, always choosing the subregion containing the largest sum of elements for assigning colors as well as further division.

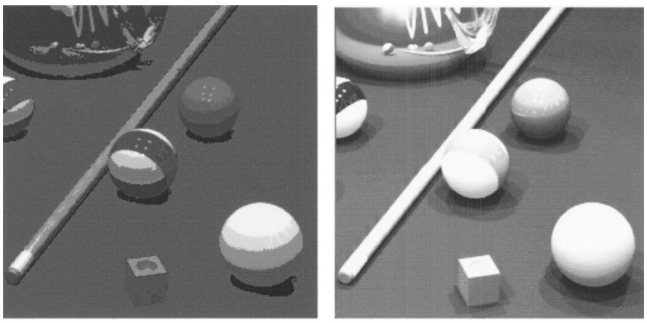


Fig. 2 (a) Artificial image that is uniformly quantized to contain 256 colors and (b) the corresponding pseudo-image generated by summing the RGB values of the original true color image at each pixel.

2.2 *Image Subregion Representation*

Let X be the input image array to our algorithm of size $K \times L$. If we consider a square image of a binary size, i.e., $K=L=N=2^r$, the dimensions of which are divisible by 2, when each side of the image is divided by 4, 16 equal subregions are obtained, as shown in Fig. 3(a). Let a side of a subregion be $w=N/4$.

With a simple scheme that picks all subregions with the largest sum of all elements for assigning colors, it is likely that the resulting color map will contain many entries of similar colors, neglecting other less popular ones. To prevent this, we consider a neighborhood or subset of the image:

$$X_k(l_k,m_k)=\{X_{k-1}(l_k+x,m_k+y)|x,y\in 0,1,\ldots,2w-1\} \tag{1}$$

where $k=1,2,\ldots,w=2^{r-k-1}$, and l_k and m_k each takes the values of 0, w, and $2w$.

In this way, pixels in an area (or segment) of $2w\times 2w$ are grouped together, as shown in Fig. 3(b). With this arrangement, nine overlapping segments are naturally formed

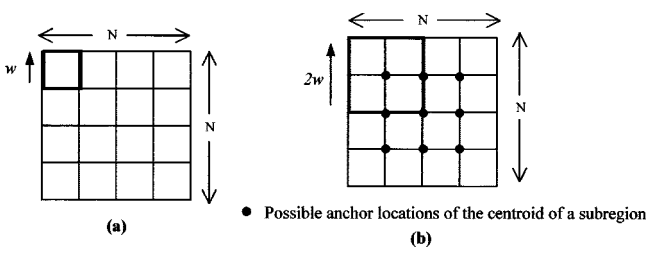


Fig. 3 (a) Scheme of dividing the image into 16 subregions, where the size of each subregion is $w\times w$ and (b) scheme of dividing the image into 9 subregions, with the size of each subregion is $2w \times 2w$.

within the whole image with each segment overlapping half the area of an adjacent segment. The elements associated with a segment E_k are defined as

$$E_k(l_k,m_k)=\sum_{x=0}^{(2w-1)}\sum_{y=0}^{(2w-1)}X_k(l_k+x,m_k+y). \quad (2)$$

The highest E_k is selected and the corresponding X_k is subdivided according to Eq. (1). Before proceeding to the next subsegment of the highest local intensity for further subdivision, colors are extracted as the color map entries from each of the nine subsegments sequentially based on their local popularity. In Eq. (1), a higher k refers to a smaller subregion such that X_0 represents the original input, and X_r is the finest level containing a single element corresponding to the actual pixel of the X array. Note that the process of subdividing X_k for color extraction within the algorithm will not continue up to $k=r-1$. This is because the color of a single pixel element is not likely to be an effective representation of any important color information in an image. Instead, the process will terminate at a particular level when X_k has reached a certain size of $n\times n$. We assume to stop extracting colors at a level that is sufficient to include less popular colors that may be essential in contributing to the final image quality, while further color extraction will not contribute much to the final image quality.

2.3 *Extracting Colors for the Color Map*

From $PI1$, we identify the regions of the image to emphasize, and then extract the color representatives that comprise the color map based on their local popularity. The popularity is obtained by counting the frequency of occurrence of each color point by point accordingly from $PI2$ at the same locations. The colors of the highest frequency counts are selected for construction of color map. To ensure that the dominant colors will be extracted in case the region contains a large part of background colors, the colors among the first three highest frequency counts are selected, so that more than one color will be extracted at one time.

Let $ID1=(id1_1,id1_2,\ldots,id1_n)$ be the list of colors extracted from a particular image segment. Starting with the first value $(id1_1)$ from the list of $ID1$, a second list of index values, $ID2=(id2_1,id2_2,\ldots,id2_n)$, can be generated consequently to include all possible colors within a distance D from the extracted color in the RGB color space. The colors among the list of $ID2$ are regarded as visually the same as $id1_1$ and satisfying results are obtained with the distance D equals to 5. After that, we identify any locations in $PI2$ having the same value as $id1_1$ or as those among the list of $ID2$, and assign 0 to the input image arrays at these locations to avoid the same color being selected in the next iteration of color extraction. This is followed by an update of $ID1$ by neglecting $id1_1$ as well as any subsequent values that are identical to those in the list of $ID2$.

2.4 *Algorithm*

Step 1. From the original image I of size $N\times N$, generate two pseudo-images, $PI1$ and $PI2$.

Step 2. Initialize $PI1$ as the image array X_0 for division. Set $k=1$.

Step 3. Divide the input image X_{k-1} into nine overlapping segments according to Eq. (1). Find the sum of all elements E_k associated with each segment X_k according to Eq. (2). Select the segment X_k^m with the largest E_k to be the new region of interest.

Step 4. Increment k. Set X_k^m as X_{k-1}, the input image array, further divide the input segment X_{k-1} into nine subsegments as in step 3. Label the nine subsegments as X_k^j, $j=1,2,\ldots,9$. Set $j=1$.

Step 5. From the subsegment X_k^j, identify the most frequently occurring colors within the confines of the subsegment according to $PI2$ at the same location. Select the colors with the three highest frequency counts and store them in a list $ID1$.

Step 6. Select $id1_1$, the first entry in $ID1$ as the color map entry. Generate a list of colors $ID2$ that is considered as visually the same as $id1_1$. Identify the locations of the image having the same values as those among the list of $ID2$ from $PI2$. Update both $PI1$ and $PI2$ as well as all the subsequent subsegments by assigning 0 to the locations to avoid the same color being selected in the next iteration of color extraction.

Step 7. Update $ID1$ by deleting $id1_1$ as well as any subsequent values that are identical to those among the list of $ID2$. Set the next color that remains in the list of $ID1$ as $id1_1$ and repeat step 6 until $ID1$ contains no color.

Step 8. Repeat steps 5 to 7 for $j=2,3,\ldots,9$. Select the subsegment X_k^j with the largest E_k to be the new region of interest as in step 3.

Step 9. Repeat steps 3 to 8 until the input image array X_k has reached a size $n\times n$. It is shown later that $n=8$ is optimal.

Step 10. Repeat steps 2 to 9 until the whole color map is filled by 256 colors.

3 Experiment and Discussion

A representative set of images of a size 256×256 was chosen to include both areas of smooth gradation (low frequency) and fine details (high frequency) for evaluation of our proposed quantization algorithm. The images are shown in Fig. 4. To start with, we test our algorithm by subdividing X_k until $k=5$, at which level the process of division and color extraction will terminate as X_k reaches a size 8×8. At this size we show later that, if the process of division stops before this level such that colors are extracted from larger areas, some of the less popular colors that may be essential in contributing to the final image quality will be neglected. On the other hand, further subdivision after this level to extract colors from smaller subregions will be of no significance in contributing to the final image quality.

To compare our algorithm with other color quantization schemes, we use the peak signal-to-noise ratio (PSNR)[13,14] as a gauge to measure the effectiveness of the resulting palettes. Given the original image I having dimensions of $N\times N$ and a quantized image Q, the PSNR in decibels (dB) is computed as follows:

Fig. 4 Test images: (a) "Pool," (b) "Woman," (c) "Shop," and (d) "Blythe."

$$\mathrm{PSNR}=10\log_{10}\left\{\frac{N^2\times 255^2}{\Sigma_{i=0}^{N-1}\Sigma_{j=0}^{N-1}[I(i,j)-Q(i,j)]^2}\right\}. \qquad (3)$$

Median cut and octree are the two color quantization schemes that we used for comparison. The basic idea of median cut[1] is to use each of the colors in the color map to represent an equal number of pixels in the original image. The algorithm repeatedly divides the 3-D color space into subspaces, with the median point being the boundary for cutting, such that approximately equal numbers of points will fall at each side of the cutting plane. Subdivision is stopped until K required subspaces are generated, and the color representatives that comprise the color map are obtained by averaging the colors in each subspace. For octree,[4] the concept behind which is to build a tree structure always containing a maximum of K different colors. If a further color is to be added to the tree structure, the color value will be merged with the most likely one that is already in the tree and both values will be substituted by their mean. Both algorithms are statistical color quantization schemes that, unlike our proposed algorithm, do not account for the spatial or contextual information of the input image.

The PSNR values of the various test images quantized by our algorithm and by the other two algorithms are given in Table 1. According to the table, the best results are obtained by terminating the process of division and color extraction within the algorithm after the final input image array X_k has reached a size of 8×8. All the images produced under this terminating criterion give high values in PSNR, indicating a good performance of our proposed algorithm compared to the other color quantization schemes. Implied by the reduced PSNR, terminating the division process at other sizes such as 16×16 or 4×4 will not result in any improvements to the overall performance of the algorithm, as we assumed.

Apart from altering the size of the final subsegment to terminate the division process, we test our algorithm by changing the number of colors we selected at a time from each subsegment for color extraction. To start with, the number of colors to be selected is arbitrarily set to include those among the first three highest frequency counts. The reason for this is to ensure the selection of dominant colors, especially when the subsegment contains a large portion of analogous background colors. Since colors are selected based on their local popularity, decreasing the number of colors to be extracted will bring the focus on the background colors, despite the fact that the less popular colors are actually more important. While changing the selection criteria to less than three will degrade the PSNR, increasing the number of colors to be selected will not improve the performance of our algorithm either, as shown in Table 2. Our algorithm extracts colors from subsegment to subsegment sequentially after identifying the region of interest. Allocating more quantization levels to each subsegment and hence the whole region will have the possibility of selecting too many colors from a particular region at the beginning, thus giving the subsequent regions insufficient prominence and neglecting some of the important colors in these regions.

The appearance of contouring artefacts is one of the major problems encountered by color quantization, especially in regions exhibiting smooth color transition, such as the background in "Woman" and the billiard balls in "Pool." This problem can be solved with spatial error diffusion by using the Floyd and Steinberg error diffusion filter[9] during pixel mapping. To study the overall performance of our proposed algorithm with spatial error diffusion, we measure the color reproduction errors by using the S-CIELAB color

Table 1 PSNR (in decibels) of pictures reproduced after color quantization by the contextual algorithm, median cut, and octree.

Contextual algorithm	"Pool"	"Woman"	"Shop"	"Blythe"
*$n=16$	30.8733	29.5448	25.8755	25.4181
*$n=8$	32.8724	29.5448	27.9399	28.5348
*$n=4$	30.9708	29.4979	25.8437	25.5266
Median cut	32.6382	29.9220	28.8440	28.3143
Octree	33.7274	31.2653	28.4794	29.0955

*The division process within the algorithm terminates when the size of the final input image array has reached the size of $n\times n$.

Table 2 PSNR (in decibels) of pictures reproduced after color quantization by the contextual algorithm. The division process within the algorithm terminates when the size of the input image array has reached 8×8.

	"Pool"	"Woman"	"Shop"	"Blythe"
†$f=2$	30.0157	25.5445	25.0955	28.4015
†$f=4$	31.6772	28.8182	25.4263	27.9048

†The colors selected from each subsegment are those among the first two highest frequency counts ($f=2$) and the first four highest frequency counts ($f=4$).

Table 3 Statistical parameters of S-CIELAB ΔE of the contextual algorithm followed by spatial error diffusion. Median cut (MC) and octree (OC) with error diffusion are included for comparison.

	Image								
	"Pool"			"Woman"			"Shop"		
Quantization scheme	Median	Mode	Pixels $>3\Delta E$ (%)	Median	Mode	Pixels $>3\Delta E$ (%)	Median	Mode	Pixels $>3\Delta E$ (%)
Contextual algorithm	0.360159	0.02	2.69	0.983365	0.46	8.99	0.956319	0.570	14.49
Median Cut	0.787249	1.090	3.63	1.083886	0.61	10.91	1.133692	0.410	13.69
Octree	0.975271	3.190	18.12	1.024349	0.41	7.82	1.704467	1.35	19.37

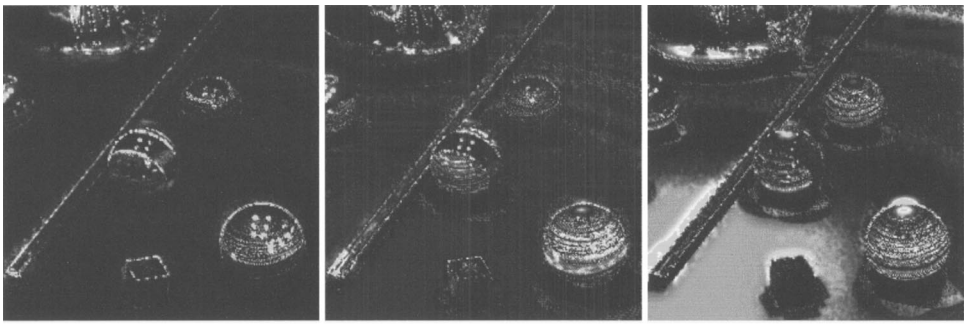
difference metric.[15] The S-CIELAB calculation is a spatial extension of CIELAB in which factors related to the pattern-color sensitivities of the human eye are incorporated in measuring color differences in digital images. The result is an error map with one ΔE value per pixel, indicating the difference between the S-CIELAB representation of the original image and the quantized image. Table 3 tabulates some statistics of S-CIELAB ΔE of three test images. At a first glance, our proposed algorithm performs more or less the same as either one of the two algorithms for different images. Obviously, with "Pool" and "Shop," our algorithm performs much better than octree, while for "Woman," our algorithm resembles octree and outperforms median cut.

With contextual information taken into account, the superiority of our algorithm lies in the capability to focus on the regions of an image having important color information, thus allocating more quantization levels to these regions. Let us consider an example. The "Pool" image was selected for its wide range of colors. Figure 5 is the S-CIELAB error images of three copies of "Pool" showing the spatial distribution of color reproduction errors resulting from different color quantization schemes. As indicated by the figures, it is apparent that our algorithm outperforms the other algorithms in providing a good selection of colors on the billiard balls exhibiting a smooth transition from dark to bright primary colors. Notice especially the deficiency in the yellow color in the images quantized by other methods. Due to the presence of large number of green pixels in the image, more quantization levels are allocated to the green colors in the background, despite the fact that the billiard balls require more colors, such that an image with optimized image quality can be reproduced. This is a fundamental problem of those statistical quantization algorithms that do not account for the spatial or contextual information of the input image. As more computation is involved in the analysis of the input image, our algorithm requires more execution time. Operating with a 600-MHz personal computer, the execution times for median cut range from 94 to 172 s, for Octree from 181 to 249 s, and for the contextual algorithm from 210 to 372 s.

We evaluated the contextual information of the input image and identify regions of interest by subdividing a pseudo-image into image segments. Another possible way to analyze an image is by color segmentation, which has long been used in many applications to identify regions of interest and objects in the scene. An argument is that instead of using fixed spatial image segments for operation, color segmentation should be conducted first and the operation applied to color segments.

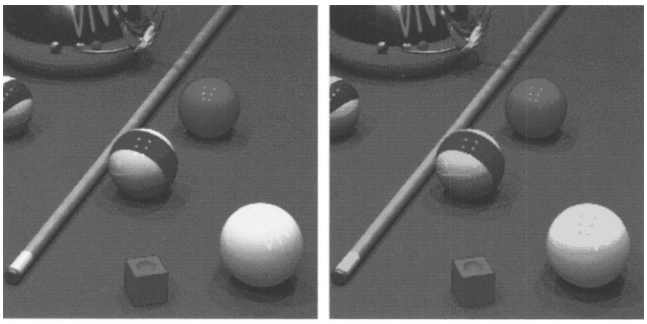
Suppose we conduct color segmentation first to separate the individual object in an image into color segments, and identify the region of interest by locating the color segment having the largest sum of elements for assigning colors. It is expected that more quantization levels will be allocated to the color segment that constitutes a larger area of the image and hence a larger sum of RGB values. Since colors having large coverage might not possibly be the same as colors that are important in contributing to the image quality, our aim to derive a mechanism to account for contextual information of an image might not be achieved.

Figure 6(b) is a quantized image "Pool" produced by performing color segmentation first with the technique based on the split and merge algorithm suggested by



(a) **(b)** **(c)**

Fig. 5 S-CIELAB error images of three copies of "Pool" reproduced from (a) the contextual algorithm, (b) median cut, and (c) octree. A higher intensity indicates a higher ΔE. Pixels with $\Delta E \geqslant 5$ are patched in green.



(a) (b)

Fig. 6 Color images quantized by the contextual algorithm using (a) fixed spatial segment for operations and (b) color segments for operations.

Horowitz and Pavlidis,[16] then applying the quantization operations of our proposed algorithm to the colors segments followed by spatial error diffusion using the Floyd and Steinberg error diffusion filter. Obviously, the image quantized using color segments for identifying regions of interest, similar to the other statistical quantization algorithms, has problem in allocating enough yellow to the billiard ball at the front due to the presence of large number of green pixels in the image. The result is an increase in color reproduction error, with the percentage of pixels having S-CIELAB $\Delta E > 3$ increasing form about 3% using fixed spatial segments to approximately 9.2% when color segments are used.

4 Conclusion

We proposed a new color quantization algorithm based on the contextual information of the input image. Experimental results indicated a good performance of the contextual algorithm with the capability to focus on the regions of an image having important color information. Similar to other color quantization methods, spatial error diffusion or dithering is performed when mapping pixel values. Comparing with two existing popular methods, our algorithm yields high PSNR before dithering (Table 1) and comparable or even better distributions of S-CIELAB errors after dithering (Fig. 5).

By the use of a pseudo-image, we provided a method for evaluating the contextual information of the input image, enabling us to easily locate the dominant colors in an image. We achieved this by dividing the image into nine overlapping segments, which is a consequence of using a binary number 4 as the divisor. If a scheme of 16 segments is to be devised and an overlap of half the area of two adjacent segments is to be maintained, an image will be divided into 25 squares, i.e., each side being divided by 5. Since each side has a dimension of 2^r, it is indivisible by an odd integer. Thus the resulting squares cannot be all equal. If such a scheme is nevertheless implemented, the amount of computations for checking the pixel values of an $N \times N$ image will be $(2N/5)^2 \times 16 = (64/25)N^2$, while computations for the case with nine segments is $(2N/4)^2 \times 9 = (9/4)N^2$. The ratio of computations is 1.14 to 1, i.e., slightly more computations are required for 16 segments.

Quantization was performed under the RGB color space, and color distance was used to identify visually similar colors. The RGB color space was, however, not perceptually uniform, which means a given change in any coordinates with the same Euclidian distance does not correspond to the same perceived color difference in all regions of the color space. In principle, a perceptually more uniform space such as CIELAB is more appropriate for the process.

In Table 1, we can see that ''Pool'' and ''Shop'' having colors extracted in RGB have PSNR of about 33 and 28 dB, respectively. Suppose during the course of color quantization, we transform the color after color extraction from RGB format to CIELAB format, identify any visually similar colors within the CIELAB color space, transform the resulting colors determined in form of L^*, a^*, and b^* back to RGB and continue to quantize the image in the RGB space. The PSNR for the resultant ''Pool'' and ''Shop'' image would drop to about 30 and 26 dB, respectively. The results imply that the conversion could degrade the PSNR by more than 2 dB, which indicates the overall performance with color extraction operating in a uniform color space with a subsequent conversion may not be better than a quantization completely in RGB. Since most common displays operate in the RGB mode, an advantage of performing the quantization wholly in RGB is an elimination of computational errors created by transforming processes between color spaces.

Acknowledgment

This work is supported by the Center for Multimedia Signal Processing, the Hong Kong Polytechnic University.

References

1. P. S. Heckbert, ''Color image quantization for frame buffer display,'' *Comput. Graph.* **6**, 293–370 (1982).
2. G. Braudaway, ''A procedure for optimum choice of a small number of colors from a large color palette for color imaging,'' in *Proc. Electronic Imaging'87*, pp. 75–79, San Francisco (1987).
3. G. Joy and Z. Xiang, ''Center-cut for color image quantization,'' *Visual Comput.* **9**(10), 62–66 (1993).
4. M. Gervautz and W. Purgathofer, ''A simple method for color quantization: Octree quantization,'' in *New Trends in Computer Graphics*, Thalmann and Thalmann, Eds., pp. 219–231, Springer, New York, (1988).
5. M. T. Orchard and C. A. Bouman, ''Color quantization of images,'' *IEEE Trans. Signal Process.* **39**, 2677–2690 (1991).
6. X. Wu, ''Color quantization by dynamic programming and principal analysis,'' *ACM Trans. Graphics* **11**, 348–372 (1992).
7. H. R. Kang, *Digital Color Halftoning*, Bollingham Press, WA (1999).
8. G. Sharma and H. J. Trussell, ''Digital color imaging,'' *IEEE Trans. Image Process.* **6**(7), 901–932 (1997).
9. R. Floyd and L. Steinberg, ''An adaptive algorithm for spatial grey scale,'' *Proc. Soc. Inf. Display* **17**(12), 75–77 (1976).
10. K. T. Knox, ''Evolution of error diffusion,'' *J. Electron. Imaging* **8**(4), 422–429 (1999).
11. I. Katsavounidis and C. C. J. Kuo, ''A multiscale error diffusion technique for digital halftoning,'' *IEEE Trans. Image Process.* **6**, 483–490 (1997).
12. Y. H. Chan, ''A modified multiscale error diffusion technique for digital halftoning,'' *IEEE Signal Process. Lett.* **5**, 277–280 (1998).
13. A. N. Netravali and B. G. Haskell, *Digital Pictures: Representation, Compression, and Standards*, 2nd ed., Plenum Press, New York (1995).
14. M. Rabbani and P. W. Jones, *Digital Image Compression Techniques* (TT7), SPIE Optical Engineering Press, Bellvue, WA (1991).
15. X. Zhang and B. A. Wandell, ''A spatial extension to CIELAB for digital color image reproduction,'' in *Soc. Inf. Disp. Symp. Technical Digest* (SID27), pp. 731–734 (1997).
16. S. L. Horowitz and T. Pavlidis, ''Picture Segmentation by a Tree Traversal Algorithm,'' *J. ACM* **23**(2), 368–388 (1976).

Biographies and photographs of the authors not available.