ELSEVIER

Pattern Recognition Letters 21 (2000) 899–906

Pattern Recognition Letters

www.elsevier.nl/locate/patrec

Fast mapping algorithm for histogram to binary set conversion

Tao Chen [a], Li-Hui Chen [b,*]

[a] *School of Computer Science and Software Engineering, Monash University, Clayton Campus, VIC 3168, Australia*

[b] *School of Electrical and Electronic Engineering, Nanyang Technological University, Nanyang Avenue, Singapore 639798, Singapore*

Received 16 March 1999; received in revised form 8 June 2000

Abstract

In this paper, a *fast binary set mapping* (FBSM) algorithm is proposed for expediting the conversion from histograms to *binary color sets*. In comparison with the original mapping scheme, significant reduction in the computation complexity can be achieved. Such an efficient mapping algorithm justifies the practical usage of the prefiltering technique in the application to histogram-based image retrieval systems, especially to searching large image databases. © 2000 Published by Elsevier Science B.V. All rights reserved.

Keywords: Computation complexity; Binary color set; Histogram-based image retrieval

1. Introduction

Due to the increasingly widespread dissemination of visual data, especially digital images, efficient and effective management of image databases is becoming more and more crucial for making full use of the information. Content-based image retrieval (CBIR) systems (e.g., Flickner et al., 1995; Pass and Zabih, 1996; Wan and Kuo, 1996) recently have gained prominence over traditional keyword-based searching engines. Among these visual features, e.g., color, texture, and shape, for representing images, color histogram is the simplest and most commonly used one. In addition to being easy-to-compute, histograms are invariant to translation and rotation about the viewing axis (Swain and Ballard, 1991). Therefore, they turn to be efficient and robust in applications to image query based on the similarity comparison.

In general, the color histogram of an image is usually described as an M-dimensional vector, $\boldsymbol{h} = \{h[0], h[1], \ldots, h[M-1]\}$, where M is the number of color bins and $h[m]$ $(m = 0, 1, \ldots, M-1)$ denotes the number of pixels with color m in the image. The similarity measure for comparing two histograms can be simply defined in squared Euclidean distance as follows:

$$\mathscr{D}_{\mathrm{e}}(q,t) = (\boldsymbol{h}_q - \boldsymbol{h}_t)^{\mathrm{T}}(\boldsymbol{h}_q - \boldsymbol{h}_t) = \sum_{m=0}^{M-1} \left(h_q[m] - h_t[m]\right)^2, \tag{1}$$

* Corresponding author. Tel.: +65-7904484; fax: +65-7912687.
E-mail address: elhchen@ntu.edu.sg (L.-H. Chen).

PII: S0167-8655(00)00056-8

where $\boldsymbol{h}_q$ and $\boldsymbol{h}_t$ represent the normalized histograms (corresponding to the image sizes) of the query and target images, i.e., q and t, respectively. In order to compare images of varying sizes, histograms are usually explored via the following normalization:

$$\tilde{\boldsymbol{h}}[m] = \frac{\boldsymbol{h}[m]}{\sum_{j=0}^{M-1} \boldsymbol{h}[j]} \quad (m = 0, 1, \ldots, M-1).$$

For the sake of convenience, hereafter we refer to histogram $\boldsymbol{h}$ as the normalized one. In other words, $0 \leqslant \boldsymbol{h}[m] \leqslant 1$ and $\sum_{m=0}^{M-1} \boldsymbol{h}[m] = 1$.

It was shown by Smith (1997) that the squared Euclidean metric is a favourable candidate for use in image indexing and retrieval applications, because of its simplicity and its consistently good performance as compared with that of the quadratic counterpart introduced by Flickner et al. (1995). However, the high-dimensional histogram vectors (e.g., values of M range from 64 to 8125 as suggested by Swain and Ballard (1991)) usually make the traditional indexing techniques, such as the R-tree proposed by Guttman (1984), unsuitable for image query applications. In terms of the computation complexity, for query-to-target image comparison, the squared Euclidean metric $\mathscr{D}_\mathrm{e}$ requires $O(NM)$ floating-point multiplications in order to calculate the histogram dissimilarity, where N is the size of the database. Admittedly, as image databases are growing larger, it is necessary for CBIR systems to address efficiency issues in addition to the problem of retrieval effectiveness.

In general, there is a tradeoff between the efficiency and effectiveness of searching algorithms, i.e., to yield better retrieval results requires more sophisticated methods which are computationally more expensive. Recently, Smith (1997) proposed a two-stage prefiltering method to improve the efficiency of indexing and searching, without compromising the effectiveness. In the first stage, the system retrieves a set of p candidates from the N images ($p \ll N$) in response to a query image by using an inexpensive and possibly crude distance measure, e.g., Hamming distance metric. Even though the ranking of these images could be unsatisfactory, it just needs to guarantee that relevant and useful images are contained in this set. In the second stage, by using a more sophisticated matching technique, e.g., squared Euclidean distance, the system compares the query image to these p candidates only (rather than to all the N images), and the most relevant images are likely to be highly ranked in the resultant list.

In the prefiltering framework, a mapping technique was employed to convert histograms to binary color sets. It is noted that, with further derivation, the original mapping algorithm for the conversion from histograms to binary color sets can be improved in terms of the computational efficiency. To start with, a brief description of binary color set is given in Section 2. The fast binary set mapping (FBSM) algorithm is presented in Section 3, where it is also compared with the original one regarding the computation complexity. Section 4 concludes the paper.

2. Binary color set

Since the use of histograms aims at representing the color information of images, we examine the characteristics of color histograms of typical images. Although the image content may be spread out into many colors, usually the histogram is defined by a few most significant colors within a given image. This observation indicates the following direction: to represent image colors, one could use compact feature sets, such as binary color sets (Smith, 1997), instead of histograms. Specifically, the binary color sets are obtained from the M colors as mentioned above.

Let $\mathscr{B}^M$ be the M-dimensional binary space such that each axis in $\mathscr{B}^M$ corresponds to one color, indexed by m ($m = 0, 1, \ldots, M-1$). A color set is a binary vector in space $\mathscr{B}^M$ and determines a set of colors $\{m\}$ in an image. A binary set is equivalent to a thresholded histogram, where each color bin is thresholded into two levels. For example, given threshold τ_m for color m, a binary set $\boldsymbol{b} = \{b[0], b[1], \ldots, b[M-1]\}$ is obtained by

$$b[m] = \begin{cases} 1 & \text{if } h[m] \geqslant \tau_m, \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

In other words, the binary set indicates only those colors that are found above the threshold levels. However, it works well for representing significant color information within the image. If color m is not well represented in an image, i.e., if $b[m]$ is below threshold τ_m, it is ignored.

The binary sets had been utilized effectively in a number of applications, such as the VisualSEEk (Smith and Chang, 1996a,b) and WebSEEk (Smith and Chang, 1997). In (Smith and Chang, 1996a,b), the authors stressed that using binary color sets can provide a compact alternative to color histograms for representing color information. In this way, color content is represented by the set of only the most prominent colors in the images. Furthermore, with such a feature indexing scheme, the computational cost in calculating the similarity between features can be greatly reduced. Hence it also substantially speeds up the query processing. In (Smith and Chang, 1997), the authors also described the significance of using binary sets as the color information for searching the Web for content. They reported that setting the "color significance" threshold adaptively to select only the 80–90% most significant colors in the query histogram decreases query time dramatically without degrading retrieval effectiveness.

3. Fast binary set mapping

This section describes an efficient algorithm, called the FBSM, to determine the *nearest binary set* for an arbitrary histogram. The objective is to find the binary set $\boldsymbol{b}$ that minimizes its distance to the histogram $\boldsymbol{h}$, i.e., $\epsilon_c(\boldsymbol{h}, \boldsymbol{b})$. The distance ϵ_c is defined to be the root mean-square-error as follows:

$$\epsilon_c(\boldsymbol{h}, \boldsymbol{b}) = \sqrt{\sum_{m=0}^{M-1} \left(h[m] - \frac{b[m]}{\|\boldsymbol{b}\|} \right)^2},$$

where $\|\boldsymbol{b}\|$ denotes the number of non-zero elements associated with binary set $\boldsymbol{b}$, i.e., $\|\boldsymbol{b}\| = \sum_{m=0}^{M-1} \boldsymbol{b}[m]$. The algorithm proceeds as follows:

1. The elements in histogram $\boldsymbol{h}$ are sorted in the descending order of their values: let $\mathscr{M}$ be the sorting process, then $\boldsymbol{g} = \mathscr{M}\boldsymbol{h}$ and $g[0] \geqslant g[1] \geqslant \cdots \geqslant g[M-1]$. Note that, $\boldsymbol{h}$ can be readily obtained from $\boldsymbol{g}$ by inverting the sorting process, i.e., $\boldsymbol{h} = \mathscr{M}^{-1}\boldsymbol{g}$.
2. Starting from the single largest element, it generates a binary set of *cardinality* $\|\boldsymbol{b}\| = 1$ that assigns a value of *one* to this element, and *zero* to the others. Then, the error, $\epsilon_c(\boldsymbol{h}, \boldsymbol{b})$, between the histogram and the resultant binary set is also calculated.
3. The process is repeated for the first two largest elements in the histogram, then the first three largest elements, and so forth, until the collection of binary sets with cardinalities $\|\boldsymbol{b}\| = 0, 1, \ldots, M-1$ is generated.
4. From this collection, the binary set that minimizes the error, $\epsilon_c(\boldsymbol{h}, \boldsymbol{b})$, is the nearest binary color set for the given histogram.

The mapping algorithm proposed by Smith (1997) for generating the nearest binary sets was based on two steps. First, given cardinality n, let $\boldsymbol{r}^n$ be a binary set such that $\|\boldsymbol{r}^n\| = n$ and

$$r^n[m] = \begin{cases} 1 & m < n, \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

Then, it is obtained that

$$\epsilon_c^n(\boldsymbol{g},\boldsymbol{r}^n)=\sqrt{\sum_{m=0}^{M-1}\left(g[m]-\frac{r^n[m]}{n}\right)^2}. \tag{4}$$

The nearest binary set with cardinality n to the sorted histogram $\boldsymbol{g}(=\mathscr{M}\boldsymbol{h})$ is given by $\boldsymbol{r}^n$. By using the inverse sorting process $\mathscr{M}^{-1}$, the nearest binary color set with cardinality n to histogram $\boldsymbol{h}$ is given by $\boldsymbol{b}^n=\mathscr{M}^{-1}\boldsymbol{r}^n$. Here, ϵ_c^n also corresponds to the distance between $\boldsymbol{b}^n$ and $\boldsymbol{h}$, since $\epsilon_c^n(\boldsymbol{g},\boldsymbol{r}^n)=\epsilon_c(\boldsymbol{h},\boldsymbol{b}^n)$. In the second step, the overall nearest binary set $\boldsymbol{b}$ to histogram $\boldsymbol{h}$ is found by determining the nearest size n binary color set $\boldsymbol{b}^n$ for $n=1,2,\ldots,M$. In other words, the one which provides the minimum ϵ_c^n is the nearest binary set for histogram $\boldsymbol{h}$.

In a word, the above algorithm requires $M\log M$ *comparisons* or *subtractions* to sort the histogram, M^2 *multiplications* and M^2 *additions* to calculate the mean-square-error for $n=1,\ldots,M$, and M comparisons to determine the overall nearest binary set. That is, it demands $\mathrm{O}(M\log M+M+M^2)$ subtractions or additions and $\mathrm{O}(M^2)$ multiplications in total for the histogram to binary set conversion. Although in practice, Smith's BSFM algorithm may terminate at an earlier point, such as when the color histograms have a large number of zero bins. This is because BSFM includes an initial sorting to identify histogram elements with zero value and would terminate once reaching a sorted histogram element with zero value. Unfortunately this is not a general case in the conversion. As we observed, for unconstrained (natural) images, the number of non-zero bins in an arbitrary color histogram varies largely. It is desired to find a faster conversion algorithm which is independent of the number of bins defined and value of histogram elements in applications.

Based on the definition of the binary set of cardinality n, i.e., Eq. (3), we note that Eq. (4) can be rewritten as follows:

$$\left[\epsilon_c^n(\boldsymbol{g},\boldsymbol{r}^n)\right]^2=\sum_{m=0}^{M-1}\left(g[m]-\frac{r^n[m]}{n}\right)^2=\frac{1}{n^2}\sum_{m=0}^{M-1}(ng[m]-r^n[m])^2 \tag{5}$$

$$=\frac{1}{n^2}\left\{\sum_{m=0}^{n-1}(ng[m]-1)^2+n^2\sum_{m=n}^{M-1}g^2[m]\right\} \tag{6}$$

$$=\frac{1}{n^2}\left\{\sum_{m=0}^{n-1}\left(n^2g^2[m]-2ng[m]+1\right)+n^2\sum_{m=n}^{M-1}g^2[m]\right\}$$

$$=\frac{1}{n^2}\left\{n^2\sum_{m=0}^{M-1}g^2[m]-2n\sum_{m=0}^{n-1}g[m]+n\right\}$$

$$=\sum_{m=0}^{M-1}g^2[m]-\frac{2}{n}\sum_{m=0}^{n-1}g[m]+\frac{1}{n}$$

$$=\boldsymbol{A}-\frac{1}{n}\left(2\sum_{m=0}^{n-1}g[m]-1\right),$$

where

$$\boldsymbol{A}=\sum_{m=0}^{M-1}g^2[m]=\sum_{m=0}^{M-1}h^2[m]$$

can be treated as a constant for a given histogram $\boldsymbol{h}$. Note that the derivation from Eq. (5) to Eq. (6) is directed by the definition of Eq. (3). It is clear that minimizing $\epsilon_c^n(\boldsymbol{g}, \boldsymbol{r}^n)$ is equivalent to maximizing the following quantity:

$$\boldsymbol{B}(n) = \frac{1}{n}\left(2\sum_{m=0}^{n-1} g[m] - 1\right). \tag{7}$$

Furthermore, according to the condition that $g[m-1] \geqslant g[m] (m = 1, \ldots, M-1), \boldsymbol{B}(n)$ satisfies the following theorem.

Theorem 1. *If $\boldsymbol{B}(\hat{n}-1) \leqslant \boldsymbol{B}(\hat{n})$ and $\boldsymbol{B}(\hat{n}+1) \leqslant \boldsymbol{B}(\hat{n})$, then $\boldsymbol{B}(n-1) \leqslant \boldsymbol{B}(n)$ for $n < \hat{n}$ and $\boldsymbol{B}(n+1) \leqslant \boldsymbol{B}(n)$ for $n > \hat{n}$.*

Proof. First, we can obtain the following two inequalities straightforwardly, according to the prerequisites that $\boldsymbol{B}(\hat{n}-1) \leqslant \boldsymbol{B}(\hat{n})$ and $\boldsymbol{B}(\hat{n}+1) \leqslant \boldsymbol{B}(\hat{n})$.

$$\begin{aligned}\boldsymbol{B}(\hat{n}) - \boldsymbol{B}(\hat{n}-1) &= \frac{1}{\hat{n}}\left(2\sum_{m=0}^{\hat{n}-1} g[m] - 1\right) - \frac{1}{\hat{n}-1}\left(2\sum_{m=0}^{\hat{n}-2} g[m] - 1\right) \\ &= \frac{2}{\hat{n}} g[\hat{n}-1] - \frac{1}{\hat{n}(\hat{n}-1)}\left(2\sum_{m=0}^{\hat{n}-2} g[m] - 1\right) \geqslant 0 \\ &\Rightarrow 2(\hat{n}-1)g[\hat{n}-1] - \left(2\sum_{m=0}^{\hat{n}-2} g[m] - 1\right) \geqslant 0,\end{aligned} \tag{8}$$

$$\begin{aligned}\boldsymbol{B}(\hat{n}) - \boldsymbol{B}(\hat{n}+1) &= \frac{1}{\hat{n}}\left(2\sum_{m=0}^{\hat{n}-1} g[m] - 1\right) - \frac{1}{\hat{n}+1}\left(2\sum_{m=0}^{\hat{n}} g[m] - 1\right) \\ &= -\frac{2}{\hat{n}+1} g[\hat{n}] + \frac{1}{\hat{n}(\hat{n}+1)}\left(2\sum_{m=0}^{\hat{n}-1} g[m] - 1\right) \geqslant 0 \\ &\Rightarrow -2\hat{n}g[\hat{n}] + \left(2\sum_{m=0}^{\hat{n}-1} g[m] - 1\right) \geqslant 0.\end{aligned} \tag{9}$$

Based upon the property that $g[m-1] \geqslant g[m] \ (m = 1, \ldots, M-1)$, the following inequalities can be derived:

$$\begin{aligned}\boldsymbol{B}(\hat{n}-1) - \boldsymbol{B}(\hat{n}-2) &= \frac{1}{(\hat{n}-1)(\hat{n}-2)}\left\{2(\hat{n}-2)g[\hat{n}-2] - \left(2\sum_{m=0}^{\hat{n}-3} g[m] - 1\right)\right\} \\ &= \frac{1}{(\hat{n}-1)(\hat{n}-2)}\left\{2(\hat{n}-1)g[\hat{n}-2] - \left(2\sum_{m=0}^{\hat{n}-2} g[m] - 1\right)\right\} \\ &\geqslant \frac{1}{(\hat{n}-1)(\hat{n}-2)}\left\{2(\hat{n}-1)g[\hat{n}-1] - \left(2\sum_{m=0}^{\hat{n}-2} g[m] - 1\right)\right\} \geqslant 0,\end{aligned}$$

$$
\begin{aligned}
\boldsymbol{B}(\hat{n}+1)-\boldsymbol{B}(\hat{n}+2) &= \frac{1}{(\hat{n}+1)(\hat{n}+2)}\left\{-2(\hat{n}+1)g[\hat{n}+1]+\left(2\sum_{m=0}^{\hat{n}}g[m]-1\right)\right\}\\
&= \frac{1}{(\hat{n}+1)(\hat{n}+2)}\left\{-2(\hat{n}+1)g[\hat{n}+1]+2g[\hat{n}]+\left(2\sum_{m=0}^{\hat{n}-1}g[m]-1\right)\right\}\\
&\geqslant \frac{1}{(\hat{n}+1)(\hat{n}+2)}\left\{-2(\hat{n}+1)g[\hat{n}]+2g[\hat{n}]+\left(2\sum_{m=0}^{\hat{n}-1}g[m]-1\right)\right\}\\
&= \frac{1}{(\hat{n}+1)(\hat{n}+2)}\left\{-2\hat{n}g[\hat{n}]+\left(2\sum_{m=0}^{\hat{n}-1}g[m]-1\right)\right\}\geqslant 0.
\end{aligned}
$$

For those $n < \hat{n}-1$ and $n > \hat{n}+1$, it can be shown in similar ways. Therefore, Theorem 1 gets proven. □

Therefore, it is apparent that $\boldsymbol{B}(\hat{n}) = \max_n\{\boldsymbol{B}(n)\}$ $(n = 1,\ldots,M)$. In other words, while searching for the maximum $\boldsymbol{B}(n)$, one can stop as long as conditions $\boldsymbol{B}(\hat{n}-1) \leqslant \boldsymbol{B}(\hat{n})$ and $\boldsymbol{B}(\hat{n}+1) \leqslant \boldsymbol{B}(\hat{n})$ are met. Normally, $\hat{n} \ll M$ in practical applications. As a result, a computationally more efficient mapping algorithm for converting a given histogram $\boldsymbol{h}$ to its nearest binary color set is obtained. In general, the FBSM algorithm is clearly shown to reduce the complexity for an arbitrary size of histograms, since it terminates once certain conditions are met in the mapping process, no matter how many histogram elements are zero valued. The FBSM algorithm can be recapitulated as follows:FBSM algorithm

Step 1. Convert the histogram $\boldsymbol{h}$ to $\boldsymbol{g}$ by using the sorting process $\mathscr{M}$, i.e., $\boldsymbol{g} = \mathscr{M}\boldsymbol{h}$.

Step 2. Obtain the cardinality $\hat{n}$ such that $\hat{n} = \arg\max_n\{\boldsymbol{B}(n)\}$, using Eq. (7) and Theorem 1 as aforementioned.

Step 3. Construct the corresponding binary set for the sorted histogram $\boldsymbol{g}$

$$
r^{\hat{n}}[m] = \begin{cases} 1 & m < \hat{n}, \\ 0 & \text{otherwise.} \end{cases} \tag{10}
$$

Step 4. The nearest binary set for histogram $\boldsymbol{h}$ is finally generated by using the inverse sorting process, i.e., $\boldsymbol{b}^{\hat{n}} = \mathscr{M}^{-1}\boldsymbol{r}^{\hat{n}}$.

Concerning the computational cost, the FBSM algorithm demands $M\log M$ comparisons or subtractions to sort the histogram $\boldsymbol{h}$, $\hat{n}(\hat{n}-1)/2$ additions along with $\hat{n}$ *divisions* or multiplications to search for $\hat{n}$, i.e., for the maximum $\boldsymbol{B}(n)$. In other words, the FBSM algorithm totally requires $\mathrm{O}(M\log M + \hat{n}^2)$ subtractions or additions and $\mathrm{O}(\hat{n})$ multiplications. It is worth pointing out that $\hat{n} \ll M$ in practice, which is verified experimentally with results shown in Fig. 1 for a collection of 500 natural images. As compared with the computational requirement of the original mapping scheme, the proposed FBSM algorithm greatly cuts down on the number of multiplication operations, which comprise the major part of the computational load. Table 1 presents a comparison in computational complexity for different schemes, where the Best-BSFM refers to Smith's BSFM algorithm that terminates once all non-zero elements are considered. As a result, the FBSM algorithm requires a much lower computational burden and therefore gives a good reason for practical use of the prefiltering technique in applications to image indexing and retrieval.

We believe that Smith's technique has certain potential in practical applications of content-based image indexing and query. It is clear that the proposed FBSM helps to speed up the conversion from histograms to binary color sets. The impact of FBSM becomes more significant when the number of color bins used is larger, which is necessary under some circumstances in order to achieve a higher efficiency in query as reported by Wan and Kuo (1996). As multimedia information is emerging at a rapid speed over the Internet, the demand of techniques for searching and making use of them becomes unavoidable. We would

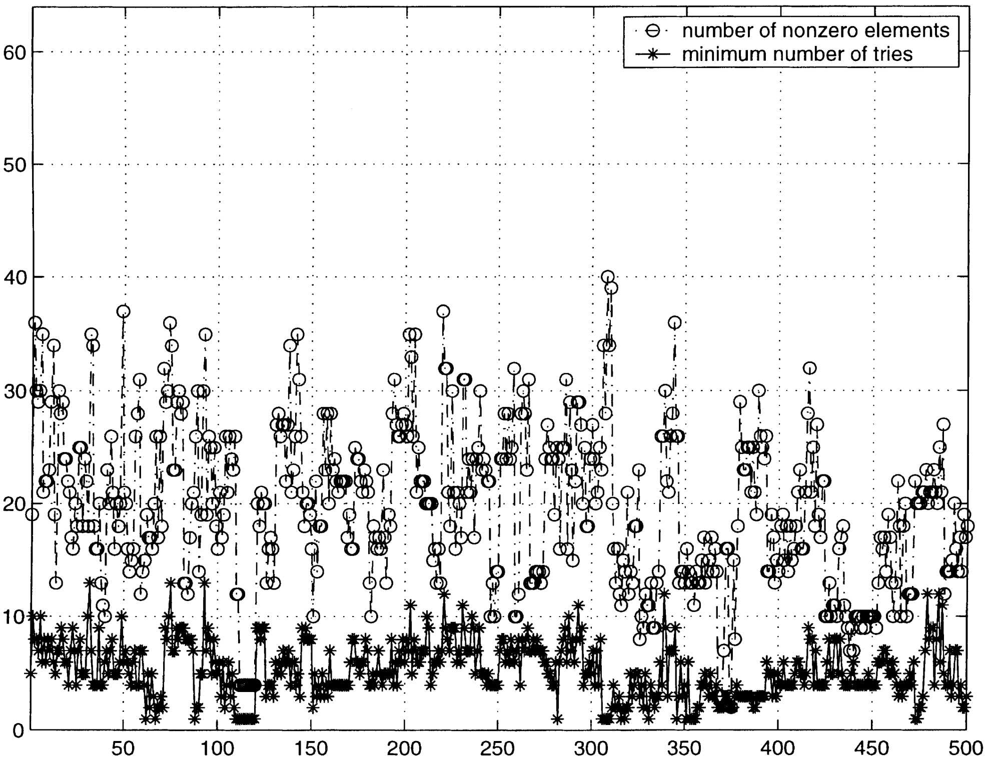

Fig. 1. Over 500 test images, it is obtained that the average *number of non-zero elements* of the histograms is $\tilde{n} = 19.6$. The *minimum number of tries* corresponds to $\hat{n}$ used in the FBSM algorithm and its average is 5.3. Notice that the number of color bins of the histograms is $M = 64$.

Table 1
Comparison in computational complexity, where $\tilde{n} = 19.6$ and $\hat{n} = 5.3$, respectively, on average over 500 images with $M = 64$

Algorithms	Operations	
	Additions	Multiplications
Smith's BSFM	$O(M \log M + M + M^2)$	$O(M^2)$
Best-BSFM	$O(M \log M + \tilde{n} + \tilde{n}^2)$	$O(\tilde{n}^2)$
Proposed FBSM	$O(M \log M + \hat{n}^2)$	$O(\hat{n})$

also like to add that the effectiveness is another important issue to practical applications of prefiltering techniques. Previously the efficacy of the prefiltering technique was discussed by Smith (1997), and it was concluded that false dismissals could be prevented by using the binary set bounding scheme. There may still exist room for further investigation on the effectiveness of such an approach in this aspect. It is hoped that the enhanced Smith's approach would hold more attraction from people to the application, improvement and extension of prefiltering schemes, especially for retrieving images from large databases.

4. Conclusion

This paper presents the efficiency consideration for histogram to binary set conversion proposed to prefilter images in histogram-based image retrieval systems. Based on the original conversion scheme, a

FBSM algorithm is derived to expedite the prefiltering process. The computational cost is significantly reduced in the proposed algorithm as compared to that of the original one. Although the binary set conversion is currently an off-line preprocessing operation, the reduction in computation complexity is always a desired and pursued task in practice, especially for indexing and retrieving a voluminous set of multimedia documents which is and will be an essential application in the future Internet and other application domain. This further justifies the potential use of the prefiltering technique.

References

Flickner, M., et al., 1995. Query by image and video content: the QBIC system. IEEE Comput. 28 (9), 23–32.

Guttman, A., 1984. R-trees: a dynamic index structure for spatial searching. In: Proc. ACM SIGMOD, Boston, MA.

Pass, G., Zabih, R., 1996. Histogram refinement for content-based image retrieval. In: Proc. IEEE Workshop on Application of Computer Vision, pp. 96–102.

Smith, J.R., 1997. Integrated spatial and feature image systems: retrieval, analysis and compression, Ph.D. thesis, Columbia University.

Smith, J.R., Chang, S.-F., 1996a. VisualSEEk: a fully automated content-based image query system. In: Proc. fourth ACM Internat. Conf. on Multimedia, pp. 87–98.

Smith, J.R., Chang, S.-F., 1996b. Local color and texture extraction and spatial query. In: Proc. IEEE International Conf. Image Processing, Vol. 3, pp. 1011–1014.

Smith, J.R., Chang, S.-F., 1997. Visually searching the web for content. IEEE Multimedia 43 (3), 12–20.

Swain, M.J., Ballard, D.H., 1991. Color indexing. Int. J. Comput. Vision 7 (1), 11–32.

Wan, X., Kuo, C.-C., 1996. Image retrieval based on multiresolution color space quantization. In: Proc. SPIE: Electronic Imaging and Multimedia Systems.