In this paper, a robust color image retrieval algorithm is proposed based on the hybrid graph representation, i.e., a dual graph which consists of the Modified Color Adjacency Graph (MCAG) and Spatial Variance Graph (SVG). The MCAG, which is similar to the Color Adjacency Graph (CAG) [6], is proposed to enhance the indexing ability and the database capacity, by increasing the feature dimension. In addition, the SVG is introduced, in order to utilize the geometric statistics of the chromatic segment in the spatial domain. In the matching process, we expand the histogram intersection [2] into the graph intersection, in which graph matching is performed using simple matrix operations. Intensive discussions and experimental results are provided to evaluate the performance of the proposed algorithm. Experiments are carried out on the Swain’s test images and the Virage images, demonstrating that the proposed algorithm yields high retrieval performance with tolerable computational complexity. It is also shown that the proposed algorithm works well, even if the query image is corrupted, e.g., a large part of pixels is missing.

1. Introduction

Image retrieval from a large database is an important problem as the application of multimedia technology increases [1]. In the traditional database system, a context-based query and retrieval scheme, based on the textual keyword or file name, is usually adopted. However, a visual database is usually very large, so that such an approach requires complicated preclassification and, furthermore, the same image might be described in different ways by different people. In this context, a content-based image retrieval technique is very attractive. Among several contents of image, the color histogram is known to provide useful clues for measuring the similarity of two images, since it is robust to object distortion, including deformation, translation, rotation, occlusion, and scaling of the object. Moreover, image retrieval based on a histogram is very fast, making a real-time implementation possible [2]. Therefore, many studies have reported on histogram-based color image retrieval techniques [2–4,8,9].

Swain and Ballard [2] proposed a color histogram based on the so-called color indexing algorithm to identify color images. The 3D histograms are generated for the input and model images in the database. Then attempts are made to match two images, employing the histogram intersection method. The technique in Ref. [2] is very simple to implement, while providing good performance. Punt and Finlayson [3] proposed the color constant color indexing algorithm to take into account the effect of different illumination. It is shown [3] that the differentiation of the logarithm of the image is independent of the illumination conditions, making illumination-invariant indexing possible.

In addition to the color histogram itself, color adjacency information is also used for indexing [5,6,8,16]. Enesser and Medioni [5] developed a local histogram method to locate an object in the color image, in which the co-occurrence histogram is employed to increase the dimension of the feature space. Matas et al. [6] proposed the Color Adjacency Graph (CAG), in which salient chromatic information is carried by each node, while the reflectance ratio of the adjacent color components is employed for the attributes of the each edge. Note that Ref. [6] utilizes graph-matching, so that an object could be labeled in the whole image. It is interesting to note that several attempts have been made recently to implement the image search engine on the world wide web (WWW) [12–14].
The major issues which should be considered in image retrieval are successful retrieval rate, matching speed, and the capacity of the database. The retrieval rate is significantly affected by the deformation, rotation and translation of the object, noise addition, illumination changes, and variation of viewpoint. Matching speed is determined by the histogram quantization, database size, image size, and resolution. On the other hand, since the visual database is usually very large, the capacity of the indexing algorithm is of special interest [10]. This provokes the demand for increasing the dimension of the feature space. This is the main motivation of the proposed algorithm in this paper.

In our approach, the relationship between two distinct nodes is properly modeled in both chromatic and spatial domains. In the Modified Color Adjacency Graph (MCAG), the pixel adjacency of two chromatic regions is also taken into account, in addition to the histogram itself. Before constructing the MCAG, an image needs to be processed in order to remove the noisy channel which lies on the boundary between two adjacent chromatic regions. In our approach, the majority filtering technique [15] is employed to remove the noise channel. It is found that the chromatic edges between the regions are enhanced significantly after applying the majority filter. In addition to the MCAG, the Spatial Variance Graph (SVG) is also proposed to improve the performance, in which self and relational variance of chromatic regions are utilized. The proposed SVG differs from other previous work [7] in that moment features are constructed using the distribution of color pixels in the spatial domain. In the matching process, the cost is defined by the weighted sum of the graph intersection results. Actually, the graph intersection is the generalization of the histogram intersection [2]. Finally, each graph is relevantly interpreted in terms of the matrix form, and then the graph intersection is done by introducing a few matrix operations.

This paper is organized as follows. In Section 2, the proposed algorithms, including the MCAG and the SVG representations of an image, and matching technique are presented in detail. Section 3 presents the experimental results of the proposed algorithm and comparison with other algorithms [2]. Thereafter, we give concluding remarks in Section 4.

2. The proposed algorithms

Content-based retrieval is a new paradigm in database management systems. When it comes to the ‘content’ of an image, we mean the generic image properties, such as color, texture, shape and composition. Among these properties, color provides direct and quick access to image, since the color is easy to process with a tolerable computational complexity. In this context, the proposed graph representation is based on the color distribution, i.e., a color histogram of the image. It is worth noting that the graph-based approach could be easily extended to other features.

Subsequently, we shall describe the proposed algorithm in detail. The flow chart for the proposed algorithm is shown in Fig. 1.

2.1. Preprocessing with majority filter

It is found that there exists a noisy channel near the boundary of adjacent chromatic regions. This stems from the blending effect of light and the CCD input characteristics. Thus, in order to acquire more reliable and noiseless graph representation, the color image needs to be processed to yield a clear boundary between the regions. In our approach, the majority filtering technique [15] is employed, since all minor noise pixels can be merged with the major pixels in the filtering window.

Let \( W \) be the filtering window, in which there exists \( k \) different chromatic components, \( C_1, C_2, \ldots, C_k \). Then the majority operator is defined as

\[
M(P(x)) = \begin{cases} 
C_i, & N_i > N_j, \ 1 \leq j \leq k, \ j \neq i \\
\vdots
\end{cases}
\]

where \( N_i \) is the pixel count of \( C_i \) in \( W \).

An example of the majority filtering is shown in Fig. 2, in which upper-right part of Fig. 2a is illustrated in Fig. 2b. As

---

![Fig. 1. The flowchart for the proposed algorithm.](image1)

![Fig. 2. Majority filtering. (a) Real image; (b) upper-right region of (a) magnified; (c) filtering result of (b).](image2)
shown, a thin noisy channel lies between the blue and orange regions. However, after applying the majority operator, the noisy channel is significantly removed, as depicted in Fig. 2c. This improves the chromatic adjacency between the regions, making the resultant graph represent the image more clearly.

2.2. Modified color adjacency graph (MCAG)

To represent a color image in the feature space, we introduce the Modified Color Adjacency Graph (MCAG) representation. Since each quantized histogram bin is mapped into a node of the MCAG, there exist as many nodes as the number of effective histogram bins. In the MCAG, the node attribute encodes the pixel count of the RGB chromatic component, while the edge attribute denotes the spatial adjacency of two color features. In the implementation, we consider the color adjacency based on eight-connectivity. A 3 × 3 window is applied to every pixel in each region, in which the pixel count of the neighboring region is added to the corresponding edge label. A simple image and its resultant MCAG are shown in Fig. 3.

Usually there are many nodes in the real image, so that the size of a graph also becomes very large. For instance, if we quantize each R, G, B axis into eight levels, the number of histogram bins would be 512. Thus, there are 512 nodes and \( \binom{512}{2} \) edges in the graph, requiring tremendous computational burden to match all of the nodes and edges. Fortunately, however, the majority of the nodes and edges which can be excluded before matching are found to be null. In our approach, a node is classified into an effective one when the count of pixels in the node exceeds the given threshold. The threshold \( \tau \) is determined to be the maximum value, satisfying the need that the sum of node attributes above

<table>
<thead>
<tr>
<th>Index</th>
<th>0</th>
<th>147</th>
<th>148</th>
<th>149</th>
<th>163</th>
<th>166</th>
<th>171</th>
<th>193</th>
<th>194</th>
<th>228</th>
<th>235</th>
<th>243</th>
<th>300</th>
<th>301</th>
<th>365</th>
<th>373</th>
<th>437</th>
<th>438</th>
<th>446</th>
<th>506</th>
<th>507</th>
<th>510</th>
<th>511</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>82</td>
<td>64</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>147</td>
<td>98</td>
<td>10</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>9</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>148</td>
<td>128</td>
<td>56</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>10</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>149</td>
<td>180</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>163</td>
<td>80</td>
<td>26</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>18</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>166</td>
<td>74</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>171</td>
<td>104</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>40</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>193</td>
<td>48</td>
<td>22</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>194</td>
<td>82</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>228</td>
<td>264</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>117</td>
<td>34</td>
<td>27</td>
<td>8</td>
<td>8</td>
<td>18</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>235</td>
<td>388</td>
<td>41</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>243</td>
<td>118</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>300</td>
<td>216</td>
<td>77</td>
<td>129</td>
<td>14</td>
<td>14</td>
<td>13</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>301</td>
<td>80</td>
<td>112</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>16</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>365</td>
<td>782</td>
<td>155</td>
<td>61</td>
<td>46</td>
<td>13</td>
<td>0</td>
<td>0</td>
<td>8</td>
<td>13</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>373</td>
<td>186</td>
<td>57</td>
<td>62</td>
<td>8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>437</td>
<td>174</td>
<td>114</td>
<td>21</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>29</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>438</td>
<td>396</td>
<td>89</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>64</td>
<td>68</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>446</td>
<td>102</td>
<td>0</td>
<td>0</td>
<td>88</td>
<td>37</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>506</td>
<td>716</td>
<td>143</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>7</td>
<td>322</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>507</td>
<td>268</td>
<td>205</td>
<td>3888</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
respectively. Then, the probability of class occurrence is given by

\[
\tau = \max_k \left( \frac{\sum_{i} N_i}{N_k} > N_t f \right)
\]

where \( N_i \) and \( N_k \) denote the \( i \)th node attribute and the total pixel count, respectively.

Several advantages of the MCAG representation are explained as follows. First, since the MCAG is based on a color histogram, the advantages of the histogram indexing technique are also applicable. Next, the geometrical information can be taken into account, since the adjacency of spatial regions is considered in the graph. In addition, since the graph representation can describe an image in a compact way, an effective matching procedure can be established.

Table 1 shows the matrix representation of the MCAG for a real image shown in Fig. 4. As is shown, the matrix size is quite small; there are only 23 nodes. In this case, the amount of image data is reduced to one-sixtieth of the original. Notice that the compression of the data is one of the crucial factors when image retrieval is considered on the network.

### 2.3. Spatial variance graph (SVG)

Although the MCAG provides a representative model in both histogram space and spatial domain, it is not sufficient to obtain the geometric statistics of each chromatic component. Since the spatial distribution of single color can be considered as another meaningful attribute of the node, a dual graph representation is developed in our approach in addition to the MCAG.

Fig. 5 illustrates the usefulness of the SVG, in which the histogram intersection method [2] regards all the images as the same one, since the pixel count of each region is fixed from image to image. On the other hand, the MCAG cannot distinguish (a) and (b) from (c) using color adjacency. However, we shall show that the employment of the statistical characteristics of the chromatic regions in the spatial domain makes it possible to obtain a unique graph for each image.

First, let us consider two nodes \( k \) and \( l \) in the MCAG with their attributes \( N_k \) and \( N_l \), respectively, forming two classes of pixels: \( C_k \) and \( C_l \). In the spatial domain, let us denote the image coordinates of the pixel in \( C_k \) and \( C_l \) by \( p_i \) and \( p_j \), respectively. Then, the probability of class occurrence is given by

\[
\omega_k = Pr(C_k) = \frac{N_k}{N_l + N_k},
\]

\[
\omega_l = Pr(C_l) = \frac{N_l}{N_l + N_k},
\]

In the image coordinates, first- and second-order statistics of two classes are given by

\[
\hat{\mu}_k = \frac{1}{N_k} \sum_{i=1}^{N_k} p_i^T
\]

\[
\hat{\mu}_l = \frac{1}{N_l} \sum_{i=1}^{N_l} p_i^T
\]

\[
\hat{\mu}_T = \omega_k \hat{\mu}_k + \omega_l \hat{\mu}_l
\]

\[
\sigma_k^2 = \frac{1}{N_k} \sum_{i=1}^{N_k} (p_i^T - \hat{\mu}_k)^T (p_i^T - \hat{\mu}_k)
\]

\[
\sigma_l^2 = \frac{1}{N_l} \sum_{i=1}^{N_l} (p_i^T - \hat{\mu}_l)^T (p_i^T - \hat{\mu}_l)
\]

\[
\sigma_T^2 = \frac{1}{N_l + N_k} \left\{ \sum_{i=1}^{N_k} (p_i^T - \hat{\mu}_T)^T (p_i^T - \hat{\mu}_T) \right. \]

\[
+ \sum_{i=1}^{N_l} (p_i^T - \hat{\mu}_T)^T (p_i^T - \hat{\mu}_T) \right\}
\]

where \( \sigma_k^2 \), \( \sigma_l^2 \), and \( \sigma_T^2 \) denote the within-class variance, the between-class variance, and the total variance of two classes, respectively. In fact, the following relationship holds:

\[
\sigma_k^2 + \sigma_l^2 = \sigma_T^2
\]

In the SVG, each node takes the self variance as its attribute,
and one of the relational variances serves as the edge attribute. This provides a statistical model for the color distribution in the spatial domain. It is experimentally found that the best matching performance is achieved by using the within-class variance, although the difference is not significantly distinguished. The SVG is invariant in translation and rotation, if the geometric relationship of each region remains unchanged, since the variance does not depend on pixel coordinates. However, when the input is scaled with respect to the model image, the information contained in the SVG could be incorrect, due to the scaling. To compensate the discrepancy, the SVG of the input should be normalized, simply by multiplying the ratio of the model image size to the input image size.

2.4. Similarity metric

Since every model image in the database can be represented by the MCAG and the SVG in off-line processing, the generation of the graph for database images does not affect the matching complexity. In the implementation, the MCAG and the SVG are represented in the matrix form. The adjacency matrix $M_{MCAG}$ and $M_{SVG}$ are considered, in which the diagonal and off-diagonal elements denote the node and edge attributes, respectively. In addition, a selection matrix $S$ is considered to choose the effective nodes. $S$ is a diagonal binary matrix, in which $S_{i,i}$ is 1, if the $i$th node is the effective one. Then, by multiplying $S$ to $M_{MCAG}$ and $M_{SVG}$, we obtain the resultant matrix $\overline{M}_{MCAG}$ and $\overline{M}_{SVG}$, as in Eq. (14), in which nonzero elements denote self or relational attributes of the effective nodes and edges.

$$\overline{M}_{MCAG} = SM_{MCAG}S$$
$$\overline{M}_{SVG} = SM_{SVG}S$$  \hspace{1cm} (14)

The similarity metric for $\overline{M}$ (either $\overline{M}_{MCAG}$ or $\overline{M}_{SVG}$) is, in fact, the generalization of Swain’s; the histogram intersection is expanded into the graph intersection. In other words, the similarity between the two graphs is measured by the intersection of the graphs. Fig. 6 shows the procedure for the similarity computation. Two graphs to be compared are shown in Fig. 6a, in which dashed circles and lines denote the null nodes and edges, respectively. The matrix representation, $\overline{M}$, of each graph is shown in Fig. 6b. Note that the common nodes and edges are distinguished by rectangles.

Before introducing the similarity metric, let us first define a few terms and operations for $\overline{M}$ as follows:

1. Norm $\|\overline{M}\|$: the sum of all elements in upper triangular part of $\overline{M}$.
2. Intersection $\overline{M}_1 \cap \overline{M}_2$: reduced matrix with common nodes and edges of both $\overline{M}_1$ and $\overline{M}_2$, in which the ($i,j$) element is the smaller one of two corresponding labels. For example, the intersection of matrices in Fig. 6b is

$$\begin{pmatrix}
8 & 3 \\
3 & 5
\end{pmatrix}$$

3. Norm of union $\|\overline{M}_1 \cup \overline{M}_2\|$

Based on the above operations, the similarity metric is now defined, given by

$$S = \alpha \frac{\|\overline{M}_{INPUT} \cap \overline{M}_{MODEL}\|}{\|\overline{M}_{INPUT} \cup \overline{M}_{MODEL}\|} + \beta \frac{\|\overline{M}_{SVG} \cap \overline{M}_{MODEL}\|}{\|\overline{M}_{INPUT} \cup \overline{M}_{MODEL}\|}$$  \hspace{1cm} (16)

where the weights $\alpha$ and $\beta$ are usually chosen to be 1. Each metric in Eq. (16) increases from 0 to 1, as the images become more similar. For example, the computed similarity

---

Fig. 6. Computation of similarity. (a) Graph representations; (b) matrix representations of the graph in (a); (c) similarity computation.
of the graphs in Fig. 6 is 0.195, as shown in Fig. 6c. Notice that, in Eq. (16), the similarity $S$ is defined as the weighted sum of the metrics of the MCAG and SVG.

2.5. The complexity of the proposed algorithm

Now, let us briefly consider the complexity of the proposed graph matching. There are two computational complexity terms: graph generation time $t_g$ and graph matching time $t_m$. Assuming there are $M$ model images in the database, the approximated computational complexities for each term are approximately $t_g = O(N_I^2)$ and $t_m = O(M \cdot N_G^2)$, where $N_I$ and $N_G$ denote the image and graph size, respectively. Thus, total complexity is $t_g + t_m$, which is slightly more complex than Swain’s algorithm.

3. Experimental results

In this section, we explore several experiments to demonstrate the performance of the proposed algorithm. Two databases are tested: Swain’s and Virage image database [12], which are shown in Figs. 7–10. Of the two databases, the Virage images are believed to be more natural, since there is no black background in the Virage image.

In the implementation, we employ the RGB coordinates. Although the HSI color coordinate is preferred in several cases [8,13], we found that the indexing performance is almost comparable. In our experiments, each chromatic axis is quantized into eight levels, yielding 512 histogram bins. Thus, the maximum number of nodes in the graph could be 512. However, as discussed previously, majority of the nodes are found to be null. For instance, in the case of Swain’s database, the average number of effective nodes is observed only to be 108, even if we set $f = 1.0$ in Eq. (2), i.e., all the non-empty nodes are regarded as effective. The remaining others are all empty nodes, which are ignored when storing and matching the graph.

The indexing capability is measured, in terms of the rank, the successful matching rate (SMR), and the average match percentile (AMP). The SMR is defined as the rate of perfect
Table 2
Indexing result with Swain’s image (32 inputs, 66 models, \( f = 1.0 \))

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Rank 1</th>
<th>Rank 2</th>
<th>( \geq ) Rank 3</th>
<th>SMR</th>
<th>AMP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hybrid graph</td>
<td>32</td>
<td>0</td>
<td>0</td>
<td>1.000</td>
<td>1.000</td>
</tr>
<tr>
<td>MCAG only</td>
<td>31</td>
<td>1</td>
<td>0</td>
<td>0.969</td>
<td>1.000</td>
</tr>
<tr>
<td>SVG only</td>
<td>30</td>
<td>0</td>
<td>2</td>
<td>0.938</td>
<td>0.998</td>
</tr>
<tr>
<td>Swain’s</td>
<td>29</td>
<td>3</td>
<td>0</td>
<td>0.907</td>
<td>0.998</td>
</tr>
</tbody>
</table>

Table 3
Indexing result with the Virage images (50 inputs, 104 models, \( f = 1.0 \))

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Rank 1</th>
<th>Rank 2</th>
<th>( \geq ) Rank 3</th>
<th>SMR</th>
<th>AMP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hybrid graph</td>
<td>49</td>
<td>1</td>
<td>0</td>
<td>0.980</td>
<td>1.000</td>
</tr>
<tr>
<td>MCAG only</td>
<td>47</td>
<td>2</td>
<td>1</td>
<td>0.940</td>
<td>0.999</td>
</tr>
<tr>
<td>SVG only</td>
<td>47</td>
<td>1</td>
<td>2</td>
<td>0.940</td>
<td>0.999</td>
</tr>
</tbody>
</table>

Fig. 9. Virage image database [12].

Fig. 10. Example of the Virage input images (total 50 images) [12].
retrieval. In addition, in order to take the overall rank into account, the AMP, $h$, is also defined as

$$h = \frac{1}{M} \sum_{i=1}^{M} \frac{N - R_i}{N - 1}$$

(17)

where $N$ and $M$ denote the number of model and input images, respectively, and $R_i$ is the rank of the $i$th input query. If all the input queries are perfectly retrieved, then $h$ would be 1, while $h$ decreases to 0 as the count of the poorly retrieved queries increases.

Let us consider 32 images, which are shown in Fig. 8, for the input query to Swain’s database. Notice that the objects in the input query are distorted by deformation, occlusion, and scaling. The results are presented in Table 2, in which it is shown that the hybrid graph approach yields the perfect result. In addition, it is observed that either MCAG or SVG alone also provides a satisfactory result, which is even better than Swain’s.

The matching performance is also evaluated on the Virage database. There are 104 images in the database, and attempts are made to retrieve 50 input queries, which are shown in Fig. 10. In this case, the distortion is much more severe than the Swain’s images. The final result is provided in Table 3. It is observed that the result is perfect, with one exception: the query, retrieved, and the correct answer are also shown in Fig. 11. As shown in Fig. 11, there are many small objects scattered randomly, making each chromatic region very thin and noisy. Thus, it is believed that the majority operator probably fails in this case. As a result, the resultant MCAG and SVG provide incorrect information about the chromatic regions.

To examine the real environment, attempts are also made to retrieve similar images for a large database. The database contains more than 1000 images. The results are shown in Fig. 12. In Fig. 12, the leftmost image is the query and those following to the right are the retrieved images ordered by their similarity. Note that the computational complexity is also tolerable. It takes about 1 s for an image to be retrieved.

In the final experiment, we attempt to reduce the size of the graph, while maintaining the indexing performance. As
described in Section 2, a node is considered to be effective, if the node label is greater than the threshold $t$ given in Eq. (2). The average number of effective nodes is computed for Swain’s database as varying the $f$ from 1.0 to 0. As $f$ decreases, the number of effective nodes decreases dramatically, as illustrated in Fig. 13. For example, if $f = 0.7$, i.e., 70% of total pixels are used for generating the graph, we can see that only 17 nodes are sufficient for representing the image. Although the graph size is reduced, it is observed that there is only slight degradation in the indexing performance. Table 4 and Table 5 demonstrate the indexing results using several choices of $f$ with Swain’s and the Virage database, respectively.

4. Conclusions

In this paper, we presented a novel algorithm to retrieve color images using their contents. A hybrid graph representation—MCAG and SVG—is proposed to construct the features, making use of regional adjacency and the spatial distribution of each color region in the image. The proposed algorithm was validated in the experiments, by testing the both Swain’s and the Virage databases. It was observed that the proposed algorithm works fairly well for the poor input set and the large database. In addition, the proposed graph representation is attractive, since it reduces the image data significantly. Therefore, it could be relevant for utilization when searching and retrieving images in a remote digital library through the network.

Another area of future work will be object recognition problem using global and local graphs, as well as multiple graph representation, which can employ other features, such as texture and shape. In parallel with the feature construction, further investigation is required for problems relating to the capacity of the retrieval algorithm.

### References


![Fig. 13. Average number of effective nodes for Swain’s image when $f$ varies.](image-url)


