CONFERM: Connectivity Features with Randomized Masks and Their Applications to Image Indexing

Arindam Biswas, Partha Bhowmick
Computer Science & Technology Department
Bengal Engineering College, India
{abiswas, partha}@becs.ac.in

Bhargab B. Bhattacharya
Center for Soft Computing Research
Indian Statistical Institute, Kolkata, India
bhargab@isical.ac.in

Abstract

A greedy algorithm for constructing a variable length, spatial feature vector of a binary image, depending on the size and diversity of the image database, is proposed. The novelty of the algorithm lies in iterative construction of the elements of the feature vector for some resultant images obtained by **xor***-ing the given image with a few pseudorandom synthetic masks. The classical Euler number and the two basic connectivity features from which it is derived, namely, the number of connected components and the number of holes, are used to finally generate a unique index (feature vector) for each image in the database. The computation process, being purely in the integer and Boolean domain, is very fast and easily implementable. It is found to converge within 3 iterations for a logo image benchmark database consisting of 1034 images. A kd-tree data structure has been used for efficient storage and retrieval of the images based on the proposed feature vector. The method is particularly suitable for object type of images.*

1. Introduction

Topological properties of a digital image typically represent the geometric shape of the image. These properties are resistant to the changes made to the image, such as stretching, rotation, scaling, translation, and other rubber-sheet transformations. Euler number of a binary image is an important topological feature, defined as the number of connected components minus the number of holes [5, 7]. It can be used as one of the major features for image classification and image indexing.

Euler number has numerous applications in medical diagnosis, viz. for detection of malaria infected cells. Recently, it is also used as a discriminating feature to locate many cervical disorders [6]. It also finds usage in optical character recognition, document image processing [10], shadow detection [9], and in constructing feature vectors [11]. The strength and elegance of Euler number lies in its simplicity, ability to capture the overall structural property of a binary image, and easy implementability. Furthermore, availability of low-cost on-chip computation of Euler number [3] makes it a very convenient feature for image indexing. Recently, several attempts have been made to exploit the strength and usefulness of Euler number for handling gray-scale images [4] with the help of "Euler vector", which is constructed from the Euler numbers corresponding to the significant bit-planes of a gray-scale image.

Although Euler number has certain discriminating ability, it alone cannot serve the purpose of uniquely identifying or indexing all the images in even a small or moderately-sized database. Hence, a few additional features, if judiciously selected, along with the Euler number, may perform the desired task of discriminating all the images in a database.

In this work, we use Euler number, as well as the two basic spatial features from which it is derived, namely, the number of connected components, C, and the number of holes, H, along with a masking technique for classifying the images with higher efficiency. The striking feature of this simple yet novel approach lies in extracting these dual characteristics (i.e., C and H) iteratively from a few derived images obtained by Boolean **xor**-ing the given image with synthetic pseudorandom masks. The ultimate objective of the procedure is to generate a unique index (feature vector) for each image in a binary image database. The proposed algorithm is easily implementable and usually converges within a small number of iterations. The algorithm has been tested on two different image databases, and the results are found to be very encouraging. For example, when applied on a logo database of 1034 binary images, the algorithm converges within 3 iterations, thereby requiring only 3 randomized masks to uniquely characterize all of them. For this database, the feature vector consists of at most 9 integers, and on the average, 5 integers. Importantly, there is no floating point computation involved in the entire process unlike indexing with features like center of mass, moments, etc. [8].

2. Proposed work

Given a database consisting of N binary images, the final feature vector must have N distinct values so that there is always a one-to-one correspondence between each image and its representative vector. We here show that, simply the Euler number along with the number of connected components and the number of holes of the original image, and those computed iteratively for the derived images, can be used to generate the desired feature vector of the original image.

2.1. Euler Number: the primary feature

An efficient and easily implementable algorithm for computing the Euler number of a binary image is based on the exhaustive study of the local patterns [7], which form the following set of 2×2 bit patterns, called *bit quad*:

$$Q_1 = \left\{ \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \right\}$$

$$Q_2 = \left\{ \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \right\}$$

$$Q_3 = \left\{ \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right\}$$

If q_1, q_2, q_3 be the respective number of patterns from the sets Q_1, Q_2, Q_3 in a binary image $\mathcal{I}$, then the Euler number for $\mathcal{I}$ is given by Eqn. 1:

$$E(\mathcal{I}) = \begin{cases} \frac{1}{4}(q_1 - q_2 + 2q_3) \text{ in 4-neighborhood} \\ \frac{1}{4}(q_1 - q_2 - 2q_3) \text{ in 8-neighborhood} \end{cases} \quad (1)$$

It may be noted that, the bit quads in the sets Q_1, Q_2, and Q_3 are distinct. Hence, each of the 10 bit quads, when enumerated in row major order, yields a unique 4-bit number, thereby enabling the formation of a Look Up Table (LUT) containing the 10 unique elements of the following modified sets:

$Q_1 = \{1, 2, 4, 8\}$

$Q_2 = \{14, 13, 11, 7\}$

$Q_3 = \{6, 9\}$

We have implemented the algorithm [7] for computing the Euler number in 8-neighborhood using the above LUT.

2.2. Number of connected components and number of holes: the secondary features

For a binary image database $\mathcal{D} = \{\mathcal{I}_k\}_{k=1}^{N}$, there may exist several images having identical values of Euler number. Let there are η distinct Euler numbers, $\epsilon_1, \epsilon_2, \ldots, \epsilon_\eta$, for all the images in $\mathcal{D}$. Let $\mathcal{E}_t$ be the subset of $\mathcal{D}$, which contains images with same Euler number ϵ_t. Since ϵ_t can not classify the images in $\mathcal{E}_t$ any more, so the number of connected components, C, and the number of holes, H, where, $C - H = E$, can be chosen as the features to (partially) segregate the images in each subset $\mathcal{E}_t$ of $\mathcal{D}$. We have adopted the algorithm given in [5] for finding out the number of connected components, $C(\mathcal{I}_k)$, of each image $\mathcal{I}_k$. Now, with the help of the tuple of two features $\langle C(\mathcal{I}_k), H(\mathcal{I}_k)\rangle$, we have devised a scheme that can be employed to distinctly index all N images in the database $\mathcal{D}$.

2.3. Iterative XOR-ing with randomized masks

In order to differentiate two or more binary images with identical $\langle C, H\rangle$, an artificial binary mask is generated pseudorandomly. When this mask is Boolean **xor**-ed pixel by pixel with an image $\mathcal{I}_k$ having the tuple $\langle C(\mathcal{I}_k), H(\mathcal{I}_k)\rangle$, a new image $\mathcal{J}_k$ will be produced with the tuple $\langle C(\mathcal{J}_k), H(\mathcal{J}_k)\rangle$, which is possibly different from $\langle C(\mathcal{I}_k), H(\mathcal{I}_k)\rangle$. In spite of its random characterization, a particular mask, however, may not be able to segregate all images in a particular set $\{\mathcal{I}_k\}$ having identical $\langle C, H\rangle$, and therefore, more randomized masks may be required to resolve the conflict. Hence, generation of the randomized masks are done over several iterations in our algorithm, so that, after the final iteration, each of the N images in the database $\mathcal{D}$ would be distinctly indexed.

It may be observed that images in the database $\mathcal{D}$ may have nonuniform sizes. Hence, in order to have the compatibility of **xor**-operation between any image of $\mathcal{D}$ and a mask of predefined size, all images in $\mathcal{D}$ are normalized to the size of the mask. Further, at each iteration, a single mask is produced for all images in $\mathcal{D}$, and the masks over all iterations have identical predefined size, $\mu \times \mu$. In our experiments, we have considered $\mu = 256$ pixels.

Let $\mathcal{A}_\alpha$ be a square matrix of size $\alpha \times \alpha$ (such that α divides μ), where, $\mathcal{A}_\alpha(x, y) \in \{0, 1\}$ for $1 \leq x, y \leq \alpha$. The matrix $\mathcal{A}_\alpha$ is constructed based on an eight-bit positive integer r, which is generated randomly each time for deciding each element of $\mathcal{A}_\alpha$. Let $r(x, y)$ be the random number that decides the entry $\mathcal{A}_\alpha(x, y)$. Then the decision for $\mathcal{A}_\alpha(x, y)$ is taken as follows:

$$\mathcal{A}_\alpha(x, y) = \begin{cases} 1 \text{ if } r(x, y) > \rho \\ 0 \text{ otherwise} \end{cases} \quad (2)$$

In Eqn. 2, the parameter ρ, which is an integer, plays a crucial role in determining the ratio of the number of 0s to that of 1s in $\mathcal{A}_\alpha$. It can be shown that, based on uniform distribution of r in the integer range [0,255], the probability that the number of 0s is same (or, differs by at most unity, if α is odd) as that of 1s in $\mathcal{A}_\alpha$, would be maximum for $\rho = \rho_0 (= 127)$. Further, departure in the value of ρ from ρ_0 will reduce the probability that the number of 0s is same as that of 1s in $\mathcal{A}_\alpha$. The generation of the synthetic mask depends on the generation of the corresponding $\mathcal{A}_\alpha$ in the concerned iteration, and therefore, changing the value of ρ will play a significant role in designing the masking scheme, if desired.

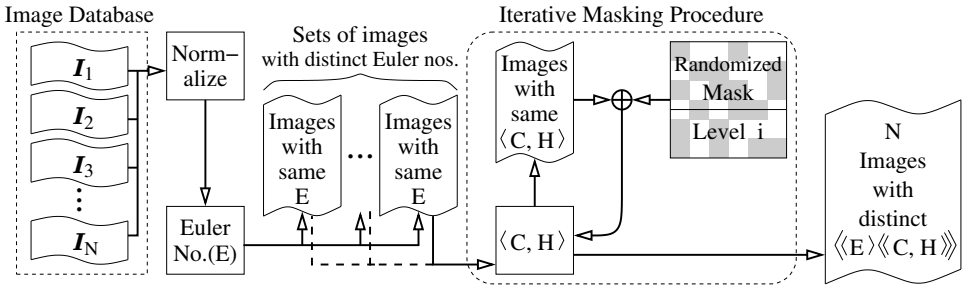


Figure 1. A schematic layout of the proposed method.

However, we have not changed ρ in our experiments and have considered $\rho = \rho_0$ over all iterations for all cases.

Let $\mathcal{A}_\alpha^{(i)}$ be the matrix that is generated at the (i)th iteration, such that, the Hamming distance $L\left(\mathcal{A}_\alpha^{(i)}, \mathcal{A}_\alpha^{(i-1)}\right)$ between $\mathcal{A}_\alpha^{(i)}$ and $\mathcal{A}_\alpha^{(i-1)}$, $i \geq 1$, $\mathcal{A}_\alpha^{(0)} = \mathbf{0}$, satisfies the criterion stated in Eqn. 3.

$$\begin{aligned} L\left(\mathcal{A}_\alpha^{(i)}, \mathcal{A}_\alpha^{(i-1)}\right) &= \sum_{x=1}^{\alpha}\sum_{y=1}^{\alpha} \mathcal{A}_\alpha^{(i)}(x,y) \sim \mathcal{A}_\alpha^{(i-1)}(x,y) \\ &\geq \tau \end{aligned} \tag{3}$$

where, $\mathcal{A}_\alpha^{(i)}(x,y) \sim \mathcal{A}_\alpha^{(i-1)}(x,y)$
$= 0$, if $\mathcal{A}_\alpha^{(i)}(x,y) = \mathcal{A}_\alpha^{(i-1)}(x,y)$;
$= 1$, otherwise .

Let $\mathcal{M}_\alpha^{(i)}$ be the randomized synthetic binary mask of size $\mu \times \mu$ generated in the (i)th iteration. $\mathcal{M}_\alpha^{(i)}$ is generated from $\mathcal{A}_\alpha^{(i)}$ as follows. Let $\beta(= \mu/\alpha)$ be the length of the square block that acts as the "building block" of $\mathcal{M}_\alpha^{(i)}$. That is, $\mathcal{M}_\alpha^{(i)}$ consists of α^2 building blocks, where, each building block is made up of β^2 pixels (all of which are either 0 or 1 for a particular building block). Then $\mathcal{M}_\alpha^{(i)}$ is constructed as shown in the following equation:

$$\begin{aligned} \mathcal{M}_\alpha^{(i)}(x,y) &= \mathcal{A}_\alpha^{(i)}(p,q), \\ &\text{where, } p = \left\lceil \frac{x}{\beta} \right\rceil, q = \left\lceil \frac{y}{\beta} \right\rceil, \\ &\forall x, \forall y, 1 \leq x \leq \mu, 1 \leq y \leq \mu, \end{aligned} \tag{4}$$

such that, the mask $\mathcal{M}_\alpha^{(i)}$ is different from the one in the preceding iteration, $\mathcal{M}_\alpha^{(i-1)}$, $i \geq 1$, by at least $\tau\beta^2$ pixels, in conformity with Eqn. 3.

Let $\mathcal{J}_k^{(i)}$ be the image obtained when **xor**-operation (represented by '$\oplus$') is performed between $\mathcal{I}_k$ and the mask $\mathcal{M}_\alpha^{(i)}$ in the (i)th iteration. That is, $\mathcal{J}_k^{(i)} = \mathcal{I}_k \oplus \mathcal{M}_\alpha^{(i)}$. An example of **xor**-operation between a sample image with a randomized mask is shown in Fig. 3.

Let $\{\mathcal{I}_k\}$ be a subset of images with all of its **xor**-ed images $\{\mathcal{J}_k^{(i-1)}\}$ having identical $\langle C, H\rangle$ in $(i-1)$th iteration.

Figure 3. A randomized mask (mask $\mathcal{M}_8^{(1)}$ shown in Fig. 4) overlaid on a sample image to obtain the **xor**-ed result.

When each image $\mathcal{I}_k$ in this set is **xor**-ed with $\mathcal{M}_\alpha^{(i)}$ in (i)th iteration to get a set of images $\{\mathcal{J}_k^{(i)}\} = \{\mathcal{I}_k \oplus \mathcal{M}_\alpha^{(i)}\}$, there would be possibly different $\langle C, H\rangle$ tuples in the set $\{\mathcal{J}_k^{(i)}\}$, thereby offering further classification of the images in (i)th iteration of the algorithm. The iterations are carried on until all images with same Euler number have distinct ordered set of tuples, $\left\langle\langle C, H\rangle\right\rangle$, considered over all required number of iterations. In effect, all N images in the database $\mathcal{D}$ have N distinct feature vectors (of possibly unequal sizes), each of which is of the form $\left\langle \langle E_k\rangle \left\langle \langle C_k^{(i)}, H_k^{(i)}\rangle \right\rangle_{i=0}^{i\leq j} \right\rangle$ corresponding to a unique image $\mathcal{I}_k \in \mathcal{D}$, where, j is the total number of iterations required for $\mathcal{D}$, and $\langle C_k^{(0)}, H_k^{(0)}\rangle$ is $\langle C, H\rangle$ for $\mathcal{I}_k$ without any **xor**-ing. A schematic representation of the entire algorithm is shown in Fig. 1. Fig. 2 demonstrates the iterative **xor**-ing procedure and generation of requisite feature vectors on a small representative set of 8 images. The **xor**-ing procedure converges just in 2 iterations, which indicates the speed and efficiency of the proposed method. The randomized masks used in the two iterations are shown in Fig. 4. It may be noted that, out of the 8 images comprising the exemplary set in Fig. 2, the length of the feature vector for image $\mathcal{I}_{260}$ ($\langle C, H\rangle$ not required) is just 1, that for $\mathcal{I}_{418}$ (no **xor**-ing required) is $1 + 2 = 3$, whereas, the same for images $\mathcal{I}_{244}$ and $\mathcal{I}_{300}$ (**xor**-ed up to 2nd iteration) is $1 + 2 \times 3 = 7$ each, and for the remaining 4 images (**xor**-ed in 1st iteration only), length of the feature vector for each becomes $1+2\times2 = 5$. Thus, the length of the feature vector in the proposed algorithm may vary from image to image.

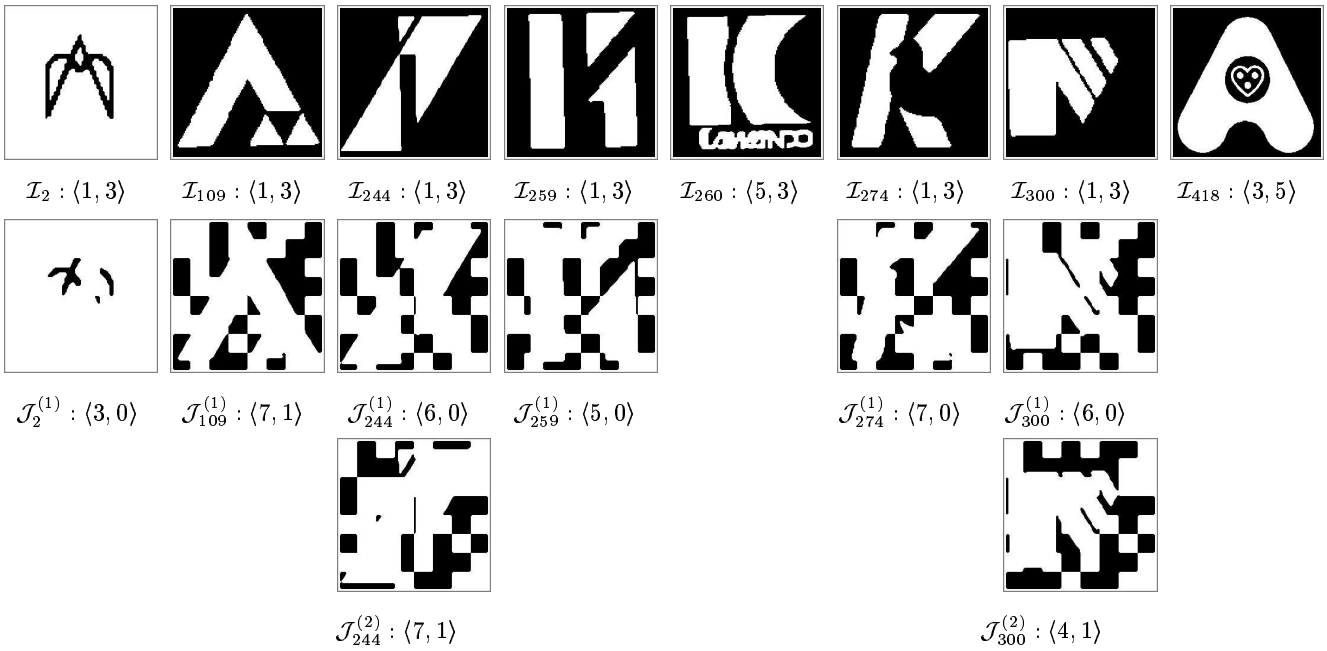


Figure 2. Demonstration of the proposed algorithm on a representative set of 8 images for generation of feature vectors of varying sizes by iterative **xor**-ing with randomized masks. For the images in the 1st row, their $\langle C^{(0)}, H^{(0)}\rangle$ are given in the figure, where, 7 images have same Euler number (E=-2), and one ($\mathcal{I}_{260}$) has a different Euler number (E=2). Further, among 7 images with E=-2, excepting the image $\mathcal{I}_{418}$ with $\langle C^{(0)}, H^{(0)}\rangle = \langle 3, 5\rangle$, each of the 6 images has $\langle C^{(0)}, H^{(0)}\rangle = \langle 1, 3\rangle$. Hence, these six images with $\langle E\rangle\langle C^{(0)}, H^{(0)}\rangle = \langle -2\rangle\langle 1, 3\rangle$ are selected for **xor**-ing in iteration (1). The 2nd row shows the **xor**-ed images in iteration (1) and the resultant tuples, $\langle C^{(1)}, H^{(1)}\rangle$. In the 2nd row, excepting the two images, $\mathcal{J}_{244}^{(1)}$ and $\mathcal{J}_{300}^{(1)}$, all other images have distinct $\langle C^{(1)}, H^{(1)}\rangle$, and the formation of their feature vectors, therefore, ends here. Since the images $\mathcal{J}_{244}^{(1)}$ and $\mathcal{J}_{300}^{(1)}$ have identical $\langle C^{(1)}, H^{(1)}\rangle$, their parent images, $\mathcal{I}_{244}$ and $\mathcal{I}_{300}$, are considered in the 2nd iteration, where they get **xor**-ed with the corresponding randomized mask, and their $\langle C^{(2)}, H^{(2)}\rangle$ become different, as shown in 3rd row of the figure. Thus the process terminates after iteration (2).

Similar images are more likely to have longer feature vectors, since they cannot be distinguished among themselves with a small number of features, whereas, an image, possessing lesser similarity with the rest of the images in the database, is likely to have a shorter feature vector.

2.4. Variable feature vector

As evident from the Sec. 2.3, the feature vectors obtained for various images are characterized by variable feature length, depending on their structural behavior apropos the iterative **xor**-ing with randomized masks, as used in our algorithm. In most of the conventional procedures, the length of a feature vector (sometimes the feature vector itself) is predefined. In contrast, adaptive construction of feature vectors, depending on the need and the topological properties of an image, is done here based on the output of each iteration of the randomized masking procedure. This resembles a greedy algorithm, and provides an effective and near-optimal choice of feature vectors. Furthermore, the dimensionality of the feature vector that has maximal length in the concerned space, would increase automatically with the execution of the algorithm on a database of larger size and diversity. The number of iterations in the randomized masking procedure increases with an appreciable increase in the size and diversity of the image database.

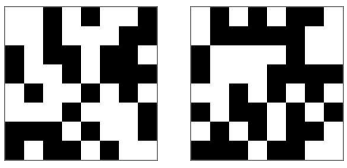


Figure 4. Randomized masks generated at iterations (1) and (2), and used for **xor**-operation with the images shown in Fig. 2.

2.5. Algorithm

1. Set parameters α, ρ, μ, τ (default: 8, 127, 256, 8).
2. Find the Euler number E_k of each normalized image

$\mathcal{I}_k \in \mathcal{D}, 1 \leq k \leq N$, N being the number of images in $\mathcal{D}$.

3. If there are η distinct Euler numbers obtained in step (2), then construct η subsets of $\mathcal{D}$, such that each subset $\mathcal{E}_t, 1 \leq t \leq \eta$, contains images with same Euler number.

4. For each image $\mathcal{I}_k \in \mathcal{E}_t, \forall t, 1 \leq t \leq \eta$, find the tuple $\langle C_k^{(0)}, H_k^{(0)} \rangle$, and make it the first entry of the ordered set of $\langle C, H \rangle$ tuples, to get $\left\langle \langle C_k^{(0)}, H_k^{(0)} \rangle \right\rangle$, for $\mathcal{I}_k$.

5. Construct the subsets of subset $\mathcal{E}_t$, $\forall t, 1 \leq t \leq \eta$, such that for each subset $\mathcal{E}_t$, each of its subsets contains images with identical $\langle C, H \rangle$. Let the (u)th subset of $\mathcal{E}_t$ be $\mathcal{S}_{t,u}$ that contains the images having identical $\langle C, H \rangle$ (and obviously, identical Euler number). If any subset $\mathcal{S}_{t,u}$ contains only one image, then for that subset the subsequent steps of the algorithm may be skipped, and final solution can be generated in step (14).

6. Initialize $i = 1$.

7. Generate a randomized mask $\mathcal{M}_\alpha^{(i)}$ conforming to Eqn. 5, such that the criterion in Eqn. 3 is satisfied. Put the corresponding $\mathcal{A}_\alpha^{(i)}$ in the final solution in step (15).

8. For each image $\mathcal{I}_k \in \mathcal{S}_{t,u}$, evaluate $\mathcal{J}_k^{(i)} = \mathcal{I}_k \oplus \mathcal{M}_\alpha^{(i)}$.

9. For the **xor**-ed image $\mathcal{J}_k^{(i)}$ corresponding to each image $\mathcal{I}_k \in \mathcal{S}_{t,u}$, find the number of connected components, $C_k^{(i)}$, and the number of holes, $H_k^{(i)}$. Append the tuple $\langle C_k^{(i)}, H_k^{(i)} \rangle$ in the ordered set of $\langle C, H \rangle$ tuples, in order to augment the previous ordered set, $\left\langle \langle C_k^{(j)}, H_k^{(j)} \rangle \right\rangle_{j=0}^{i-1}$, to get the updated ordered set, $\left\langle \langle C_k^{(j)}, H_k^{(j)} \rangle \right\rangle_{j=0}^{i}$, for $\mathcal{I}_k$.

10. Construct the subsets of $\mathcal{S}_{t,u}$, such that each subset contains images with identical $\langle C_k^{(i)}, H_k^{(i)} \rangle$. If any of these subsets contains exactly one image, then that subset is not considered in further iterations, and directly output to the final solution in step (15).

11. Execute steps (8) – (10), $\forall u$.

12. Execute steps (8) – (11), $\forall t$.

13. For each t, $1 \leq t \leq \eta$, remove all the subsets $\mathcal{S}_{t,u}$, which are used as input in steps (10) and (11) , and (re)name all the subsets obtained in steps (10) and (11) by $\mathcal{S}_{t,u}$.

14. If there is at least one $\mathcal{S}_{t,u}$ containing at least two images, then increment i by unity, and go to step (7); otherwise, assign $j = i$, and go to step (15).

15. Return the final solution, consisting of:

(i) Number of masks (iterations), j.

(ii) Ordered set of generator matrices, $\langle \mathcal{A}_\alpha^{(i)} \rangle_{i=1}^{i=j}$, needed to reconstruct the masks during image retrieval.

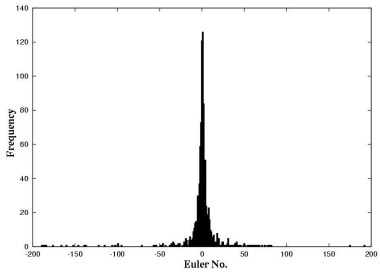
(iii) The derived feature vector for each image $\mathcal{I}_k \in \mathcal{D}$, which is an ordered set of its all relevant $\langle C, H \rangle$ tuples, preceded by its Euler number: $\langle E_k \rangle \left\langle \langle C_k^{(i)}, H_k^{(i)} \rangle \right\rangle_{i=0}^{i \leq j}, 1 \leq k \leq N$.

2.6. Data structure for storing images

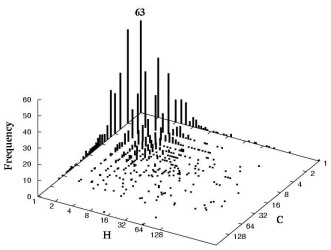
In order to store and retrieve the images using the proposed algorithm, a kd-tree [2] is used as the basic data structure. It may be noted that, the quadtree, which is commonly used in the existing image indexing methodologies [1], is not suitable for storing the feature vectors (the images, thereof) derived in this algorithm, since the quadtree would become very unbalanced because of the unbalanced distribution of $\langle C, H \rangle$ tuples in the CH plane (see Fig. 5). A kd-tree, on the other hand, would be always balanced and would ensure searching of an image in logarithmic time, since it splits a 2-dimensional plane alternately about the median w.r.t. x-coordinate and about the median w.r.t. y-coordinate, such that at each level of the kd-tree, the points stored at any two nodes differ in number at most by unity.

Let there be $n^{(i)}$ distinct tuples of $\langle C^{(i)}, H^{(i)} \rangle$ produced in the (i)th $(i = 0, 1, 2, \ldots, i)$ iteration of the algorithm for a database $\mathcal{D}$ containing N images. At (i)th iteration, the corresponding feature tuples, $\langle C^{(i)}, H^{(i)} \rangle$, on a 2-dimensional plane, namely the CH plane, give a planar set of points, which may be stored in a 2-dimensional kd-tree. Let $\mathcal{T}^{(i)}$ represent the kd-tree that is constructed at the (i)th iteration. Then $\mathcal{T}^{(i)}$ contains $n^{(i)}$ leaf nodes and has $\mathcal{O}(\log n^{(i)})$ height, where, each leaf node contains a unique $\langle C, H \rangle$ tuple that may be the $\langle C, H \rangle$ tuple for more than one image (after **xor**-ing with mask $\mathcal{M}_\alpha^{(i)}$, if $i \geq 1$, and original, if $i = 0$) in $\mathcal{D}$.

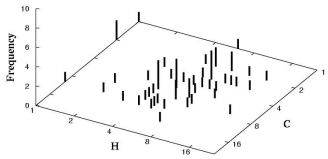
It is quite evident that, if at least one of the leaf nodes of $\mathcal{T}^{(i)}$ has more than one image of $\mathcal{D}$ associated with it, then only the algorithm proceeds for $(i + 1)$th iteration. For each such leaf node ν, having more than one associated image, in $\mathcal{T}^{(i)}$, another kd-tree $\mathcal{T}_\nu^{(i+1)}$ would be created in $(i + 1)$th iteration. The kd-tree $\mathcal{T}_\nu^{(i+1)}$ contains all the distinct $\langle C^{(j+1)}, H^{(j+1)} \rangle$ tuples of only those images which are being associated with the same node ν of $\mathcal{T}^{(i)}$. The process is repeated until each leaf node in the kd-tree has exactly one image associated with it, that is, until i reaches j, the total number of iterations as stated in step 15(i) of the algorithm in Sec. 2.5.



Frequency vs Euler no.



Frequency distribution of $\langle C^{(0)}, H^{(0)} \rangle$



Frequency distribution of $\langle C^{(1)}, H^{(1)} \rangle$ for 63 images in D1 with $\langle C^{(0)}, H^{(0)} \rangle = \langle 1, 1 \rangle$

Figure 5. Distribution of $\langle E \rangle$ and $\langle C, H \rangle$ for Database D1.

3. Results and discussions

We have used 2 sets of binary images for our experiments: (i) database D1 of 1034 logo images, received on request, from Prof. Anil K. Jain and Aditya Vailya of Michigan State Univ., USA, and, (ii) database D2 of 106 logo images collected from the Internet.

The proposed method is implemented in C on a Sun_Ultra 5_10, Sparc, 233 MHz, the OS being the SunOS Release 5.7 Generic. A tool called CONFERM (Connectivity Features with Randomized Masks) has been developed for this purpose. The average execution time for both the sets is given in Table 1. Table 2 shows in brief the results obtained on databases D1 and D2. That the output for D1 varies with different values of α, and these output again vary with the output for D2 for different values of α, reflects the randomized and adaptive nature of the proposed algorithm. In table 2, $\overline{\lambda}$ denotes the average length of feature vector for the database $\mathcal{D}$, which is given in Eqn. 5, where, λ_k denotes the feature vector length of the image $\mathcal{I}_k \in \mathcal{D}$.

$$\overline{\lambda} = \frac{1}{N} \sum_{k=1}^{k=N} \lambda_k \tag{5}$$

Table 1. Average CPU time in seconds per image for databases D1 and D2

	D1	D2
$E, \langle C^{(0)}, H^{(0)} \rangle$	0.476	0.508
$\langle C^{(i)}, H^{(i)} \rangle_{i=1}^{i=j}$	0.422	0.120
Total	0.898	0.628

To cite a few examples, four sample images from D1 and another four from D2 have been shown in Fig. 6 and Fig. 7 respectively. The images have been selected so as to represent the possible output vectors of the proposed algorithm. For instance, in Fig. 6, $\mathcal{I}_{761}$ is the only image in D1 with Euler number 38, and therefore, no other feature is required to uniquely index this image. The next image shown,

Table 2. Results for databases D1 and D2

	D1			D2		
α	8	16	32	8	16	32
$\langle E \rangle$	62	62	62	53	53	53
$\#\langle C, H \rangle^{(0)}$	247	247	247	79	79	79
$\#\langle C, H \rangle^{(1)}$	710	948	1007	106	106	106
$\#\langle C, H \rangle^{(2)}$	967	1034	1034	–	–	–
$\#\langle C, H \rangle^{(3)}$	1034	–	–	–	–	–
$\overline{\lambda}$	5.16	4.57	4.45	2.51	2.51	2.51

$\#\langle C, H \rangle^{(i)}$= No. of distinct vectors after (i)th iteration.
Each feature vector after (i)th iteration is given by:
$\langle E \rangle \langle C^{(0)}, H^{(0)} \rangle \langle C^{(1)}, H^{(1)} \rangle \cdots \langle C^{(i)}, H^{(i)} \rangle$.

on the other hand, needs the vector $\langle C^{(0)}, H^{(0)} \rangle = \langle 1, 12 \rangle$ along with its Euler number -11 since there are some other images in D1 having Euler number -11; there is, however, no other image in D1 with Euler number$= -11$ and $\langle C^{(0)}, H^{(0)} \rangle = \langle 1, 12 \rangle$. Hence, the vector $\langle -11 \rangle \langle \langle 1, 12 \rangle \rangle$ is a distinct vector with one-to-one correspondence with image $\mathcal{I}_{174}$, in the feature space of D1. Similar justifications hold for the vectors shown corresponding to the other two images in Fig. 6. The four images and their respective vectors, shown in Fig. 7, also obey the same one-to-one correspondence in the feature space of D2. In Fig. 5, the reduction in frequency of occurrences of identical $\langle C, H \rangle$ tuples with advancement of iterations in the algorithm exhibit the segregating power of the proposed indexing scheme.

4. Conclusion and future works

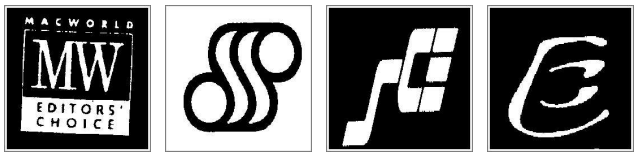
This work introduces a novel indexing technique for binary images by employing a greedy algorithm defined over a small set of traditional topological features of binary images. An efficient randomization is imparted to the algorithm to expedite the process and to make it adaptive to the database size and diversity. The iterative **xor**-ing procedure has an inherent property of capturing the topological features of an image in near-optimal number of iterations, which shows the elegance and strength of the algorithm.

There lies further scope for improvement of the proposed

$\mathcal{I}_{761}$ $\mathcal{I}_{174}$ $\mathcal{I}_{162}$ $\mathcal{I}_{261}$

$\mathcal{I}_{761} : \langle 38 \rangle$
$\mathcal{I}_{174} : \langle -11 \rangle \big\langle \langle 1, 12 \rangle \big\rangle$
$\mathcal{I}_{162} : \langle -12 \rangle \big\langle \langle 1, 13 \rangle, \langle 7, 3 \rangle \big\rangle$
$\mathcal{I}_{261} : \langle -3 \rangle \big\langle \langle 1, 4 \rangle, \langle 5, 7 \rangle, \langle 5, 10 \rangle \big\rangle$

Figure 6. 4 sample images of database D1 with their $\langle E \rangle \big\langle \langle C, H \rangle \big\rangle$.



$\mathcal{I}_{69}$ $\mathcal{I}_{53}$ $\mathcal{I}_{26}$ $\mathcal{I}_{63}$

$\mathcal{I}_{69} : \langle -11 \rangle$
$\mathcal{I}_{53} : \langle -3 \rangle \big\langle \langle 1, 4 \rangle \big\rangle$
$\mathcal{I}_{26} : \langle -4 \rangle \big\langle \langle 1, 5 \rangle, \langle 1, 1 \rangle \big\rangle$
$\mathcal{I}_{63} : \langle -2 \rangle \big\langle \langle 1, 3 \rangle, \langle 3, 1 \rangle, \langle 1, 2 \rangle \big\rangle$

Figure 7. 4 sample images of database D2 with their $\langle E \rangle \big\langle \langle C, H \rangle \big\rangle$.

method. Depending on the mask that is randomly generated at a particular iteration, the feature vectors of the images are computed. The average feature vector length, $\overline{\lambda}$, therefore, may vary for a given image database, depending on nature of the randomized masks produced during the execution of the algorithm. The parameters α, ρ, μ, τ will have some effects on the generated set of feature vectors for all the images in the database, and on $\overline{\lambda}$ there of. This is evident from Table 2, where the results do vary with change in α. The optimal solution for the set of distinct feature vectors is, therefore, very difficult to find. Experimentation on the controlling parameters, α, ρ, μ, τ, would also produce some interesting results, which would be explored in a future work.

The proposed algorithm, with suitable modifications, can also be used to uniquely index a new image, when the new image has to be inserted in an existing database. It may also be noted that, the algorithm may produce two distinct feature vectors for two images that are obtained by different 2-dimensional transformations (linear or nonlinear) of the same image. This is due to the fact that, after the **xor**-operation of the two (different w.r.t. transformations) images with the same mask, the resultant images may not possess identical $\langle C, H \rangle$ tuples. Presently, we are working on designing a suitable masking scheme that would be invariant to image transformations.

The algorithm produces a very effective indexing scheme for binary logo image databases, as observed in our experiments on two different databases, and in particular for object type of images. It may not however produce good results for other databases like human face, natural scenes, etc. For gray scale images, proper adaptation of this technique, such as Euler Vector [4], may yield desired results, but this area needs further investigation.

References

[1] I. Ahmad and W. I. Grosky, Spatial Similarity Based Retrievals and Image Indexing by Hierarchical Decomposition, *Intl. Database Engg. & Applns. Symposium (IDEAS '97)*, pp. 269-278, 1997.

[2] M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf, *Computational Geometry - Algorithms and Applications*, Springer, Berlin, Germany, 2000.

[3] A. Bishnu, B. B. Bhattacharya, M. K. Kundu, C. A. Murthy, and T. Acharya, On-Chip Computation of Euler Number of a Binary Image for Efficient Database Search, *Proc. ICIP*, vol. 3, pp. 310-313, Greece, 2001.

[4] A. Bishnu, B. B. Bhattacharya, M. K. Kundu, C. A. Murthy, and T. Acharya, Euler Vector: A Combinatorial Signature for Gray-Tone Images, *Proc. 3rd Intl. Conf. on Information Technology: Coding and Computing (ITCC)*, pp. 121-126, Las Vegas, 2002.

[5] R. Gonzalez and R. Woods, Digital Image Processing, *Addison-Wesley Publishing Company*, 1992.

[6] B. W. Pogue, M. A. Mycek, and D. Harper, Image Analysis for Discrimination of Cervical Neoplasia, *J. Biomedical Optics*, vol. 5(1), pp. 72-82, 2000.

[7] W. K. Pratt, Digital Image Processing, *John Wiley & Sons*, 1978.

[8] R. J. Prokop and A. P. Reeves, A Survey of Moment-based Techniques for Unoccluded Object Representation and Recognition, *CVGIP: Graphical Models and Image Processing*, vol. 54 (5), pp. 438-460, 1992.

[9] P. L. Rosin and T. Ellis, Image Difference Threshold Strategies and Shadow Detection, *Proc. British Machine Vision Conference*, pp. 347-356, 1995.

[10] S. N. Srihari, Document Image Understanding, *Proc. ACM/IEEE Joint Fall Computer Conference*, 1986.

[11] A. Stavrianopoulou and V. Anastassopoulos, The Euler Feature Vector, *Proc. ICPR*, vol. 3, pp. 1022-1024, Barcelona, 2000.